IN THE UNITED STATES DISTRICT COURT
DISTRICT OF DELAWARE

NOKIA TECHNOLOGIES OY, **Plaintiff,** **v.** **AMAZON.COM, INC., AMAZON.COM SERVICES LLC, AND TWITCH INTERACTIVE, INC.,** **Defendants.**	**Civil Action No.**

ORIGINAL COMPLAINT

Plaintiff Nokia Technologies Oy ("Nokia,") files this Original Complaint against Amazon.com, Inc., Amazon.com Services LLC, and Twitch Interactive, Inc. (collectively, "Amazon" or "Defendants"), and alleges as follows:

NATURE OF THE ACTION

1. This case is about Amazon's infringement of Nokia's Asserted Patents (defined below) through Amazon's video streaming products and services, including Prime Video, Amazon.com videos, and Twitch.tv.

2. Nokia is a leading innovator in video coding and streaming technology with one of the strongest video coding and streaming patent portfolios in the world. Nokia's patented inventions allow video to be transmitted and received over communications networks, such as Wi-Fi or cellular networks, with high quality and dramatically lower bandwidth requirements, and minimize the amount of data it takes to receive and store these videos on mobile devices, such as laptop computers and tablet computers. Nokia's patented inventions also allow streaming video content to be searched, filtered, and combined in ways that provide the most compelling and relevant content to users, including in a mobile environment.

Nokia Exhibit 2009, p. 1
Snap v. Nokia
IPR2025-01114

3. Nokia's Asserted Patents (defined below) include claims that are widely practiced by products and services that encode, decode, and stream video, such as those used in over-the-top (OTT) video streaming services and digital advertising, including, for example, video that is encoded into formats compliant with the H.264 Advanced Video Coding Standard ("H.264") and the H.265 High Efficiency Video Coding Standard ("H.265") promulgated by the International Telecommunications Union ("ITU"). Amazon's unlicensed streaming products and services, including without limitation Amazon streaming videos, movies, shows, trailers, and advertising, such as those on Amazon Prime Video (including Freevee), Amazon.com, and Twitch.tv, ("Accused Products"), infringe Nokia's Asserted Patents.

4. Amazon currently benefits from and has benefitted greatly from Nokia's innovations, which enable the Amazon Accused Products to stream, store, and transmit high quality video more efficiently and effectively.

5. Dozens of companies have taken a license to Nokia's video encoding and decoding patent claims. Yet, despite Nokia's good faith efforts, Amazon has not accepted any of Nokia's offers to take a license to Nokia's encoding and decoding patent claims. Amazon's failure to negotiate in good faith and refusal to license Nokia's video patents has forced Nokia to file this lawsuit.

6. This Complaint includes causes of action for patent infringement arising under 35 U.S.C. § 271, *et seq*., for the willful infringement of U.S. Patent Nos. 7,532,808 ("the '808 Patent"), 8,050,321 ("the '321 Patent"), 7,724,818 ("the '818 Patent"), 6,950,469 ("the '469 Patent"), 7,280,599 ("the '599 Patent"), 8,036,273 ("the '273 Patent"), 6,856,701 ("the '701 Patent"), 9,800,891 ("the '891 Patent"), 6,968,005 ("the '005 Patent"), 8,144,764 ("the '764 Patent"), 8,175,148 ("the '148 Patent"), 8,077,991 ("the '991 Patent"), 9,571,833 ("the '833

Patent"), 11,805,267 ("the '267 Patent"), and 9,390,137 ("the '137 Patent") (together, the "Asserted Patents").

PARTIES

7. Plaintiff Nokia Technologies Oy ("Nokia") is a Finnish corporation with its principal place of business at Karaportti 3, FIN-02610, Espoo, Finland.

8. Amazon.com, Inc. is a Delaware corporation with a principal place of business at 410 Terry Avenue North, Seattle, Washington 98109. On information and belief, Amazon.com, Inc. operates the e-commerce website "Amazon.com," which is one of the providers of the Accused Products.

9. Amazon.com Services LLC is a Delaware limited liability company with a principal place of business at 410 Terry Avenue North, Seattle, Washington 98109. It is a wholly owned subsidiary of Amazon.com, Inc. and on information and belief, operates Amazon Prime Video, which is one of the providers of the Accused Products.

10. Twitch Interactive, Inc. is a Delaware corporation with a principal place of business at 350 Bush Street, San Francisco, CA 94104. It is a wholly owned subsidiary of Amazon.com, Inc. and on information and belief, operates Twitch.tv, which is one of the providers of the Accused Products.

JURISDICTION AND VENUE

11. This Court has subject matter jurisdiction over the patent infringement claims asserted in this case under 28 U.S.C. §§ 1331 and 1338.

12. This Court has general personal jurisdiction over Amazon because each Defendant is incorporated in the State of Delaware. Amazon has appointed a registered agent for service of process, Corporation Service Company, 251 Little Falls Drive, Wilmington, Delaware 19808.

13. The Court also has specific personal jurisdiction over Amazon because Amazon has committed acts of infringement in this District.

14. Venue is proper in this District pursuant to 28 U.S.C. §§1391 and 1400(b). Venue is proper as to Amazon in this District because Amazon is incorporated in this District.

15. In addition, Amazon maintains a regular and established place of business within this District. For example, and without limitation, Amazon has maintained a regular and established place of business with offices and/or other facilities located at 1025 Boxwood Rd., Wilmington, DE 19804. At 3.8 million square feet, it is the largest Amazon fulfilment center in the United States. Exhibit 1, https://www.delawareonline.com/story/money/business/2021/09/21/amazon-opens-mega-warehouse-delaware/8347000002/. Amazon additionally maintains offices in this District including at 560 Merrimac Ave 1437, Middletown, Delaware 19709 and 820 Federal School Lane, New Castle, Delaware 19720.

THE ITU COMMON PATENT POLICY AND NOKIA'S RELEVANT DECLARATIONS

A. The International Telecommunications Union ("ITU") and the H.264 and H.265 Standardization Process

16. Certain claims of the Asserted Patents relate to the H.264 and H.265 Standards (defined below) developed by the International Telecommunication Union ("ITU").

17. The ITU and the International Standards Organization ("ISO") jointly published a standard referred to as "H.264," "MPEG-4 part 10," or "Advanced Video Coding" (the "H.264 Standard"). The H.264 Standard development process was initiated by VCEG and finalized by the Joint Video Team ("JVT"), which was a collaborative effort between VCEG and the Moving Picture Experts Group ("MPEG").

18. Following publication of the H.264 Standard, the JVT began work on the H.265 Standard. The H.265 Standard, which is also known as "MPEG-H Part 2" or "High Efficiency Video Coding," represents the next step for video quality and coding efficiency after the widely successful H.264 Standard.

19. The ITU was formed in 1865 at the International Telegraph Convention and, in 1947, it became a specialized agency of the United Nations, responsible for issues that concern information and communication technologies. The ITU handles a variety of matters and thus is organized into various sectors. One of the sectors is Telecommunication Standardization or "ITU-T." The mission of ITU-T is to ensure efficient and timely production of standards related to the field of telecommunications. The standards developed by ITU-T are referred to as "Recommendations."

20. The Guidelines define the term "Patent" to be "those claims contained in and identified by patents, utility models and other similar statutory rights based on inventions (including applications for any of these) solely to the extent that any such claims are essential to the implementation of a Recommendation | Deliverable. Essential patents are patents that would be required to implement a specific Recommendation | Deliverable." *See "Common Patent Policy for ITU-TIITU-RIISOIIEC,"* ITU (2022), https://www.itu.int/en/ITU-T/ipr/Pages/policy.aspx. The definition of "Patent" provided by the Guidelines is mirrored in the Patent Statement and Licensing Declaration Form that is completed by patent holders who may have patent claims essential to the H.264 or H.265 standards. The Patent Statement and Licensing Declaration Form states that identifying specific patents on the form is optional but not required. The ITU thus deems "essential" only patent claims that are essential or necessary for implementation of a specific Recommendation.

21. The H.264 Recommendation specifies the implementation of decoders and specifically defines the "decoding process" as "[t]he process specified in this Recommendation | International Standard that reads a bitstream and derives decoded pictures from it." Exhibit 2 at 6 [Recommendation ITU-T H.264]. It does not, however, specify the implementation of encoders. The H.264 Recommendation defines "encoding process" as "[a] process, not specified in this Recommendation | International Standard, that produces a bitstream conforming to this Recommendation | International Standard." *Id*.

22. Similarly, the H.265 Recommendation only specifies the implementation of decoders. *See* Exhibit 3 at 5 [Recommendation ITU-T H.265] (defining (i) "decoding process" as "[t]he process specified in this Specification that reads a bitstream and derives decoded pictures from it" and (ii) "encoding process" as "[a] process not specified in this Specification that produces a bitstream conforming to this Specification.").

B. Nokia's Compliance with the ITU Common Patent Policy and Nokia's Relevant Declarations

23. Consistent with the ITU Common Patent Policy, Nokia timely notified standard setting participants that it may obtain patents on its contributions, including by submitting Patent Statement and Licensing Declarations to the ITU declaring in good faith that Nokia is prepared to grant licenses to the essential claims of the relevant patents on RAND terms and conditions.

24. As explained above, the H.264 and H.265 Standards do not specify an encoding process. Therefore, any patent claims related to an encoder or encoding process are not essential to the H.264 and H.265 Standards and are not encumbered by any commitment to grant licenses to any such claims on RAND terms and conditions.

25. To the extent that Amazon alleges and proves that claims asserted in this case are actually essential to the ITU H.264 or H.265 Recommendations, the damages being sought on such claims will take into account RAND principles.

C. Nokia's Negotiations with Amazon

26. Nokia has negotiated with Amazon in good faith for a portfolio license to its patents related to H.264 and H.265. Nokia has made offers to Amazon for its end user devices and streaming services. The Asserted Patents would have been covered by these offers. Specifically, on October 25, 2023, Nokia sent Amazon an offer for a license to Nokia's patents for Amazon's streaming services, including Amazon Prime Video, Freevee, and Twitch. Nokia's offer is supported by, for example, Nokia's executed running royalty agreements covering Nokia's video patents.

27. Despite the fact that Nokia has made offers to Amazon for a license consistent with Nokia's obligations under the ITU Common Patent Policy and Nokia's declarations, Amazon has not accepted Nokia's offers.

28. Nokia has complied with all aspects of the relevant IPR policies of the relevant standard setting bodies and is entitled to seek the relief requested in this case. This Complaint is necessary to put an end to Amazon's infringing conduct.

NOKIA'S INVESTMENT IN VIDEO TECHNOLOGIES

29. The Asserted Patents arise from Nokia's long term work in the fields of wireless communication, video standards, and related technologies that enable many features that are commonplace and expected of today's consumer electronics.

30. By the mid-1990s, Nokia Corp. was developing its own proprietary video technologies, referred to as the MobiVideo Codec. In early 1998, the Video Coding Experts

Group ("VCEG") of the International Telecommunication Union-Telecommunication (ITU-T) issued a call for proposals on a project called H.26L, the "L" standing for "long term."

31. The development of H.26L eventually led to ITU-T Recommendation H.264 Advanced Video Coding for Generic Audiovisual Services ("the H.264 Standard"). Thereafter, work began on the successor to the H.264 Standard, which published as ITU-T Recommendation H.265 High Efficiency Video Coding ("the H.265 Standard"). Nokia, a video coding innovator, contributed numerous innovations to the development of both the H.264 and H.265 Standards. In addition, Nokia has developed many other video coding technologies.

32. Over the last few decades, internet traffic has evolved from simple, text-based interfaces to a plethora of media, including video. As technology has evolved, the importance and use of video has skyrocketed. Video coding technologies, including the H.264 and H.265 Standards, are crucial to the development and evolution of modern communication particularly as video traffic has become an increasingly outsized share of total consumer Internet traffic.

33. The H.264 and H.265 Standards enable efficient and reliable video decoding in millions of devices, including smartphones, computers, and tablets. The H.264 and H.265 Standards reduce the amount of data needed to decode digital video and are the two most prominent video decoding standards in the world. These advances in video coding technology were made possible by the work of Nokia and other video coding innovators.

34. The H.264 Standard, first released in 2003, was designed to decode high quality video using lower bit rates than previous standards. The H.264 Standard is flexible enough to implement across a variety of applications, networks, and systems and offers vastly improved performance over previous standards, such as MPEG-2 and MPEG-4 Part 2.

35. The H.265 Standard, first released in 2013, built on the H.264 Standard in several key respects. The H.265 Standard enables consumers to decode video with even less bandwidth than before and to decode higher quality video in higher resolutions.

36. The rise of these new video coding technologies has enabled people to consume news, sports, movies, shows, and other streaming entertainment on demand. Video was estimated to be 82% of global consumer internet traffic in 2022.

37. Amazon benefits greatly from Nokia's video coding inventions. For example, Nokia's video encoding inventions enable Amazon's Accused Products—such as Prime Video, Amazon.com video, and Twitch.tv—to offer reliable, high quality video to subscribers with far less bandwidth than would otherwise be required.

THE NOKIA PATENTS

A. U.S. Patent No. 7,532,808 ("the '808 Patent")

38. On May 12, 2009, the U.S. Patent and Trademark Office duly and legally issued U.S. Patent No. 7,532,808 ("the '808 Patent"), entitled "Method for Coding Motion in a Video Sequence," to inventor Jani Lainema. Nokia owns all rights to the '808 Patent necessary to bring this action. The '808 Patent issued from U.S. Patent Application No. 10/390,549, filed on March 14, 2003, and claims priority to U.S. Provisional Application No. 60/365,072, filed on March 15, 2002. A true and correct copy of the '808 Patent is attached hereto as Exhibit 4 and incorporated herein by reference.

39. The '808 Patent is not directed to merely an abstract idea or any patent-ineligible concept. Instead, the '808 Patent is directed to novel and unconventional improvements to motion-compensated prediction in the field of digital video coding. The '808 Patent provides improvements over prior motion compensated prediction and video compression techniques that result in substantial benefits to motion prediction, video compression, video quality, and video

playback. These substantial benefits are enjoyed by users of the Accused Products when, for example, watching video over the Internet.

40. A digital video sequence is a sequence of still images with "the illusion of motion being created by displaying the images one after the other at a relatively fast rate." '808 Patent at 1:15-19. These still images are referred to as frames. "Each frame of an uncompressed digital video sequence comprises an array of image pixels." *Id*. at 1:32-33. Frames in commonly used video formats may have millions of pixels.

41. The '808 Patent describes that video frames in a given digital video sequence may contain various forms of redundancy. *Id.* at 2:36-46. "Temporal redundancy" refers to the fact that "objects appearing in one frame of a sequence are likely to appear in subsequent frames." *Id*.

42. As the '808 Patent explains, "motion-compensated prediction" can take advantage of temporal redundancy to "predict" the image content of some frames from "one or more other frames in the sequence, known as 'reference frames.'" *Id.* at 3:15-18. Predictions can be achieved by tracking the motion of objects or regions of an image between a given frame and one or more reference frames. *Id.* at 3:18-23.

43. Prior to the '808 Patent, some motion-compensated prediction techniques involved assigning "coding modes" to "macroblocks" (a region of 16x16 image pixels in the original image). *See id*. at 1:64-2:6. One such coding mode was referred to as "SKIP" mode. SKIP mode was assigned to macroblocks that could be copied directly from a reference frame without using or having to take into account motion-compensated prediction. '808 Patent at 10:64-67. SKIP mode prior to the '808 Patent provided benefits in certain scenarios with macroblocks without motion from frame to frame.

44. As explained in the ’808 Patent, “it is necessary for a corresponding video decoder to be aware of that coding mode in order for it to correctly decode the received information relating to the macroblock in question.” ’808 Patent at 11:20-24. “Therefore, an indication of the coding mode assigned to each macroblock is provided in the video bit-stream transmitted from the video encoder to the video decoder.” ’808 Patent at 11:24-27. The indication is transmitted using a variable length codeword, where “the shortest codeword is used to represent the coding mode that is statistically most likely to occur.” ’808 Patent at 11:27-32. Prior systems assume that SKIP mode is statistically the most likely coding mode for a macroblock. ’808 Patent at 12:18-19.

45. However, SKIP mode could not effectively address problems with certain types of redundancy within video sequences—for example, global and regional motion, such as might occur when phenomena like panning or zooming are present in a video sequence. *Id*. at 12:41-47. For example, redundancies may occur in a video sequence when footage is captured by a video camera moving horizontally from a fixed position or when translational motion occurs, such as when a volleyball moves overhead across a court. Conventional SKIP mode is never actually used in these situations, and the assumption that SKIP mode is always the most probable ceases to be valid. *Id*. at 12:41-44. Therefore, prior motion-compensated prediction techniques could not efficiently or effectively handle these scenarios. For example, in the prior H.263+ video coding standard, this global motion scenario was addressed by using a highly complex global motion compensation technique that required additional information to be sent to the decoder. *Id*. at 12:48-13:30. This prior solution was computationally intensive and less efficient. *Id*.

46. The ’808 Patent overcame these technical challenges in the prior systems by inventing a novel and improved skip coding mode. The ’808 Patent’s improved skip coding mode

can address certain scenarios with motion (and/or without motion) without the need for additional motion data. For example, the ’808 Patent teaches that the skip coding mode is associated with either a zero (non-active) motion vector or a non-zero (active motion vector), where the decision is made by analyzing the motion of other macroblocks or sub-blocks in a region surrounding the macroblock to be coded. ’808 Patent at 14:23-32. Therefore, for example, “SKIP mode macroblocks can adapt to the motion in the region surrounding them, enabling global or regional motion to [be] taken account of in an efficient manner.” *Id*. at 14:48-51.

47. The assigned motion vector can then be used by the decoder, for example, to form a prediction for the given macroblock with respect to a reference frame. These unconventional solutions allow a decoder to, for example, reliably and efficiently decode video sequences with a drastically reduced amount of information. Because the ’808 Patent invention uses the surrounding macroblocks or sub-blocks to determine the assignment of the motion vector for the skip coding mode for an image segment, there is no need for the video encoder to transmit any additional information to the video decoder in order to model global or regional motion. *Id*. at 14:52-64.

48. The ’808 Patent therefore provides a specific technological improvement to the functionality and capabilities of video coding technology that “not only provides an improvement in coding efficiency in the presence of global motion . . . but also enables regional motion to be represented in an efficient manner.” *Id*. at 14:14-22.

49. Conventional technology prior to the ’808 Patent was not capable of using SKIP mode with motion-compensated prediction, including global and regional motion. Conventional technology prior to the ’808 Patent was also not capable of using the surrounding macroblocks or sub-blocks to determine the SKIP mode assignment.

50. The ’808 Patent recognizes and solves these specific technological problems with the conventional technology at the time. The ’808 Patent’s ability to assign either a zero motion vector or a predicted non-zero motion vector for the SKIP coding mode for the first segment based at least in part on the motion information of a second segment neighboring the first segment and ability to provide in an encoded bitstream an indication of the SKIP coding mode, wherein no further motion vector information for the first segment is coded in the encoded bitstream, was a significant advancement over existing technology.

51. The novel solutions of the ’808 Patent, including redefining skip coding mode to adapt to the motion of surrounding regions, was not well-understood, routine, or conventional, nor was it simply comprised of well-understood, routine, and conventional activities previously known to the industry. Furthermore, the ordered combination of elements, including assigning either a zero motion vector or a predicted non-zero motion vector for the SKIP coding mode for the first segment based at least in part on the motion information of a second segment neighboring the first segment and providing in an encoded bitstream an indication of the SKIP coding mode, wherein no further motion vector information for the first segment is coded in the encoded bitstream, was not well-understood, routine, or conventional.

B. U.S. Patent No. 8,050,321 (“the ’321 Patent”)

52. On November 1, 2011, the U.S. Patent and Trademark Office duly and legally issued U.S. Patent No. 8,050,321 (“the ’321 Patent”), entitled “Grouping of Image Frames In Video Coding,” to inventor Miska Hannuksela. Nokia owns all rights to the ’321 Patent necessary to bring this action. The ’321 Patent issued from U.S. Patent Application No. 11/338,934, filed on January 25, 2006, which is a continuation of U.S. Application No. 10/306,942, filed on November 29, 2002, which issued as U.S. Patent No. 7,894,521, and claims priority to Finnish

Application No. 20020127, filed on January 23, 2002. A true and correct copy of the ’321 Patent is attached hereto as Exhibit 5 and incorporated herein by reference.

53. The ’321 Patent is not directed to merely an abstract idea or any patent-ineligible concept. Instead, the ‘321 Patent is directed to novel and unconventional improvements to the process of video coding. The ’321 Patent provides improvements over prior video coding techniques that result in substantial benefits to video compression, video quality, and video playback. These substantial benefits are enjoyed by users of the Accused Products when, for example, watching video over the Internet.

54. Video sequences are comprised of still image frames, which are displayed rapidly in succession to create an impression of a moving image. ’321 Patent at 1:55-58. The image frames typically comprise a number of stationary background objects and few moving objects, such that the information in consecutively displayed image frames is typically largely similar. *Id*. at 1:58-65. Many video coding methods make use of this so-called “temporal redundancy” by using “motion-compensated temporal prediction,” in which the contents of an image frame are predicted from other frames. *Id*. at 2:16-23. Frames that use motion-compensated temporal prediction are also called INTER-frames. *Id*. at 2:27-29. Frames that do not use motion-compensated temporal prediction are also called INTRA-frames or I-frames. *Id*. at 2:23-26. INTRA-frames or I-frames therefore do not depend on any frames that come before them.

55. Both INTER-frames and INTRA-frames may be used in the motion-compensated prediction of another frame. However, if a frame that is used in the motion-compensated prediction of another frame is lost or corrupted, the frames dependent on it can no longer be correctly decoded. *Id*. at 2:32-33.

56. Prior to the ’321 Patent, one significant problem occurred when a user wanted to stream or browse a video from somewhere other than the beginning of the video (*e.g.*, the user wishes to start from a certain position such as the middle or where the user left off from a previous viewing). *Id*. at 3:62-4:4. Prior systems did not include a numbering scheme that allowed the decoder to recognize the first I-frame in a sequence of pictures. *Id*. at 11:11-21. Therefore, when streaming or browsing a video file from a point other than the beginning, the decoder would interpret this as an unintentional loss of image frames and unnecessarily try to reconstruct the image frames suspected as lost. *Id*. at 11:20-25.

57. The ’321 Patent overcame these technical challenges in the prior systems by inventing a novel independent sequence of image frames that includes an indication of a first picture in an independently decodable group of pictures. *Id*. at 4:16-35. The ’321 Patent employs the unconventional solution of indicating the first picture in an independently decodable group of pictures so that it is possible for the decoder to start decoding from that first picture and continue the decoding process without needing prediction from any image frame prior to that first picture. *Id*. at 4:16-38.

58. The ’321 Patent therefore provides a specific technological improvement to the functionality and capabilities of video coding technology that results in increased efficiency and improved video playback. For example, the encoder can now enable the decoder to begin decoding from a random point in a video stream without any prediction from any prior picture and without storing any pictures decoded prior to the first picture of the independent sequence in its memory. *Id*. at 4:48-58. For another example, the indication by an encoder of a first picture in an independently decodable group of pictures enables the decoder to identify a loss of that type

of picture, which is unlikely to allow satisfactory image quality without retransmission or picture refresh. *Id*. at 4:64-5:5.

59. Conventional technology prior to the ’321 Patent was not capable of identifying the first picture of an independently decodable group of pictures, encoding identifier values for the image frames according to a numbering scheme, and resetting the identifier value for the indicated first image frame of the independent sequence.

60. The ’321 Patent recognizes and solves these specific technological problems with the conventional technology at the time. The ’321 Patent’s ability to encode into the video sequence an indication of at least one image frame, which is the first image frame, in decoding order, of the independent sequence and to reset the identifier value for the indicated first image frame of the independent sequence was a significant advancement over existing technology.

61. The novel solution of the ’321 Patent, including encoding into the video sequence an indication of at least one image frame, which is the first image frame, in decoding order, of the independent sequence and resetting the identifier value for the indicated first image frame of the independent sequence, was not well-understood, routine, or conventional, nor was it simply comprised of well-understood, routine, and conventional activities previously known to the industry. Furthermore, the ordered combination of elements, including encoding into the video sequence an indication of at least one image frame, which is the first image frame, in decoding order, of the independent sequence and resetting the identifier value for the indicated first image frame of the independent sequence, was not well-understood, routine, or conventional.

C. U.S. Patent No. 7,724,818 (“the ’818 Patent”)

62. On May 25, 2010, the U.S. Patent and Trademark Office duly and legally issued U.S. Patent No. 7,724,818 (“the ’818 Patent”), entitled “Method for Coding Sequences of Pictures,” to inventors Miska Hannuksela and Ye-Kui Wang. Nokia owns all rights to the ’818

Patent necessary to bring this action. The ’818 Patent issued from U.S. Application No. 10/426,928, filed on April 30, 2003. A true and correct copy of the ’818 Patent is attached hereto as Exhibit 6 and incorporated herein by reference.

63. The ’818 Patent is not directed to merely an abstract idea or any patent-ineligible concept. Instead, the ’818 Patent provides improvements over conventional video coding techniques that result in substantial benefits to video compression, video quality, and video playback. These substantial benefits are enjoyed by users of the Accused Products when, for example, watching video over the Internet.

64. A coded picture consists of one or more slices, and a slice consists of macroblocks of pixel values. ’818 Patent at 1:51-55. Conventional video coding standards before the ’818 Patent specified a structure for a bitstream that consisted of several layers, including a sequence layer, picture layer, slice layer, and macroblock layer. *Id*. at 2:6-13. These conventional video coding standards also included the use of headers in the bitstream at the slice layer and below. *Id*. at 2:49-51. However, the data at the picture level and sequence level were included in a single parameter set structure instead of using headers. *Id*. at 2:51-56. Each instance of a parameter set included a unique identifier, and each slice header included a reference to a parameter set identifier so that the decoder could use the parameter values of the identified parameter set when decoding the slice. *Id*. at 2:56-59.

65. Prior to the ’818 Patent, one significant problem with the single parameter set structure was that many sequence-level parameters remain unchanged over parameter sets and are repeated. *Id*. at 6:54-57. In order to be able to change picture parameters (such as the picture size) without having to transmit an updated parameter set synchronously with the packet stream, encoders and decoders maintained multiple parameter sets either pre-defined or transmitted at the

beginning of a session. *Id*. at 3:5-21. Transmitting the multiple parameter sets at the beginning of a session with repetitive information was overburdening and caused latency and reliability issues. *Id*. at 3:17-24. Further, since the parameter sets were transmitted frequently to accommodate changes in users watching video, the redundant transmission was inefficient and very costly from a bit-rate point of view. *Id*. at 3:24-29.

66. The ’818 Patent overcame these technical challenges in the prior systems by splitting the parameter set structure and inventing multiple parameter set structures that could be nested according to the persistency and target of parameters, including a sequence parameter set and a picture parameter set. *Id*. at 4:3-5, 4:15-19. The ’818 Patent employs the unconventional solution of including parameter values that may change in every slice or will likely change in every picture in a slice header, including parameter values that will likely remain unchanged in multiple pictures in a picture parameter set, and including parameter values that are not allowed to change within a given coded video sequence in a sequence parameter set. *Id*. at 4:24-28.

67. The ’818 Patent therefore provides a specific technological improvement to the functionality and capabilities of video coding technology that results in increased compression efficiency and significant reduction in transmission bit-rate. *Id*. at 3:56-58, 6:50-59.

68. Conventional technology prior to the ’818 Patent did not recognize that video coding parameters may change at different rates and that some parameters may change much more frequently than others. Conventional technology prior to the ’818 Patent was not capable of transmitting parameter values using multiple parameter set structures according to the persistency and target of parameters, including a sequence parameter set and a picture parameter set.

69. The ’818 Patent recognizes and solves these specific technological problems with the conventional technology at the time. The ’818 Patent’s ability to define parameter values in a

sequence parameter set for a sequence of pictures, to define parameter values in a picture parameter set for a picture, and to define at least one picture parameter value in a slice header, the picture parameter value remaining unchanged at least in all slice headers of one picture, was a significant advancement over existing technology.

70. The novel solution of the ’818 Patent, including defining parameter values in a sequence parameter set for a sequence of pictures, defining parameter values in a picture parameter set for a picture, and defining at least one picture parameter value in a slice header, the picture parameter value remaining unchanged at least in all slice headers of one picture, was not well-understood, routine, or conventional, nor was it simply comprised of well-understood, routine, and conventional activities previously known to the industry. Furthermore, the ordered combination of elements, including defining parameter values in a sequence parameter set for a sequence of pictures, defining parameter values in a picture parameter set for a picture, and defining at least one picture parameter value in a slice header, the picture parameter value remaining unchanged at least in all slice headers of one picture, was not well-understood, routine, or conventional.

D. U.S. Patent No. 6,950,469 (“the ’469 Patent”)

71. On September 27, 2005, the U.S. Patent and Trademark Office duly and legally issued U.S. Patent No. 6,950,469 (“the ’469 Patent”), entitled “Method for Sub-Pixel Value Interpolation,” to inventors Marta Karczewicz and Antti Olli Hallapuro. Nokia owns all rights to the ’469 Patent necessary to bring this action. The ’469 Patent issued from U.S. Patent Application No. 09/954,608, filed on September 17, 2001. A true and correct copy of the ’469 Patent is attached hereto as Exhibit 7 and incorporated herein by reference.

72. The ’469 Patent is not directed to merely an abstract idea or any patent-ineligible concept. Instead, the ’469 Patent is directed to novel and unconventional improvements to

motion-compensated prediction in the field of digital video coding. The '469 Patent provides improvements over prior motion compensated prediction and video compression techniques that result in substantial benefits to motion prediction, video compression, video quality, and video playback. These substantial benefits are enjoyed by users of the Accused Products when, for example, watching video over the Internet.

73. A digital video sequence is a sequence of still images with "the illusion of motion being created by displaying consecutive images of the sequence one after the other at a relatively fast frame rate." '469 Patent at 1:10-15. Consecutive images (or frames) often tend to be only slightly different from one another because, for example, the background is stationary or only changes slowly. *Id*. at 1:18-25. Thus, there is often a "considerable amount of redundant information" between consecutive images. *Id*. at 1:15-18.

74. The '469 Patent describes one form of redundancy as "temporal redundancy," in which "the content of some (often many) frames in a video sequence is 'predicted' from other frames in the sequence by tracing the motion of objects or regions of an image between frames." '469 Patent at 2:61-65. A "prediction frame" is created by moving prediction pixels from a reference frame according to motion information, which describes the relationship between pixels in the current frame and their corresponding prediction pixels in the reference frame. *Id*. at 3:53-58. The difference between the current frame being coded and the predicted frame is referred to as the prediction error frame. *Id*. at 3:49-64.

75. A video encoder can therefore represent a current frame in a more compact way by representing the frame in terms of the motion information required to form its prediction frame and the prediction error frame. *Id*. at 3:64-4:3. Thus, the "operating principle of video coders

using motion compensation is to minimize the amount of information in a prediction error frame." *Id*. at 3:38-42.

76. However, as the '469 Patent explains, when using motion prediction, full pixel resolution is generally not sufficiently accurate to model real life motion, which has arbitrary precision. *Id*. at 6:24-35. Therefore, sub-pixel resolution is used. *Id*. at 6:35-41. Allowing motion vectors to have sub-pixel resolution adds to the complexity and burden of the encoding and decoding operations that must be performed, in part, because the sub-pixel values must be interpolated from full resolution pixels. *Id*. at 6:35-45, 7:23-26.

77. One problem with sub-pixel value interpolation is maintaining prediction accuracy while also limiting computational complexity and memory usage. For example, prior to the '469 Patent, conventional sub-pixel value interpolation methods used predetermined sets of particular nearby pixels and sub-pixels to interpolate other sub-pixel values, without providing any choice or flexibility and without sufficiently good balancing of interpolation accuracy and computational complexity. *Id*. at 11:15-21. These methods therefore required the unnecessary interpolation and storage of sub-pixel values that were needed only to interpolate other sub-pixels. *Id*. at 11:14-32. This increased computational complexity and memory requirements, as well as reduced the precision of the interpolated sub-pixel values (due to truncation when storing the values). *Id*. Other methods reduced unnecessary dependencies in the interpolation of certain sub-pixels, but those methods retained numerical precision which required high precision arithmetic and high memory requirements. *Id*. at 13:20-29.

78. The '469 Patent overcame these technical challenges in the prior systems by inventing a method of sub-pixel value interpolation that improves performance with respect to both computational complexity and memory requirements while maintaining prediction accuracy.

The ’469 Patent employs the unconventional solution of providing flexibility and a choice of which pixels and sub-pixels to use in the interpolation of other sub-pixels. ’469 Patent at 38:47-51. For example, the ’469 Patent is superior to prior art methods in that it does not require any ¼ resolution sub-pixels to depend on previously interpolated ¼ resolution sub-pixels. *Id*. at 37:36-41. According to the ’469 Patent, ¼ resolution sub-pixels that have ¼ resolution in both the horizontal and vertical directions are interpolated by a diagonally located pixel and sub-pixel or two diagonally located sub-pixels. *Id*. at 13:66-14:9. This not only reduces the number of calculations required as compared to conventional technology, it also increases the precision by eliminating truncation and clipping that occurs in the intermediate interpolation steps. *Id*. at 37:41-53. Similarly, the ’469 Patent provides the unconventional flexibility of a choice for the interpolation of the ½ resolution sub-pixels. *Id*. at 38:35-39. This also results in the minimization of operations required to perform sub-pixel interpolation. *Id*. at 38:39-43.

79. As another example, the ’469 Patent is superior to prior art methods in that it does not require high precision arithmetic to be used in the calculation of all sub-pixels. *Id*. at 37:66-38:5. The selective use of lower precision arithmetic decreases the computational complexity and increases the speed at which the calculations can be performed. *Id*. at 38:5-22.

80. The ’469 Patent therefore provides a specific technological improvement to the functionality and capabilities of video coding technology that results in increased efficiency and reduced computational complexity and memory requirements.

81. Conventional technology prior to the ’469 Patent was not capable of providing any flexibility or choice of dependencies on pixels and sub-pixels used in sub-pixel interpolation. Conventional technology prior to the ’469 Patent also could not avoid unnecessary interpolation and storage of sub-pixel values that were needed only to interpolate other sub-pixels.

82. The ’469 Patent recognizes and solves these specific technological problems with the conventional technology at the time. The ’469 Patent’s ability to use a choice of a first weighted sum of values for sub-pixels residing at $½^{N-1}$ unit horizontal and unit vertical locations and a second weighted sum of values for sub-pixels residing at unit horizontal and $½^{N-1}$ unit vertical locations and ability to take a weighted average of the value of a first sub-pixel or pixel situated at a $½^{N-m}$ unit horizontal and $½^{N-n}$ unit vertical location and the value of a second sub-pixel or pixel located at a $½^{N-p}$ unit horizontal and $½^{N-q}$ unit vertical location, variables m, n, p and q taking integer values in the range 1 to N such that the first and second sub-pixels or pixels are located diagonally with respect to the sub-pixel at $½^{N}$ unit horizontal and $½^{N}$ vertical location was a significant advancement over existing technology.

83. The novel solution of the ’469 Patent, including the flexibility of a choice of which pixels and sub-pixels to use in the interpolation of other sub-pixels, was not well-understood, routine, or conventional, nor was it simply comprised of well-understood, routine, and conventional activities previously known to the industry. Furthermore, the ordered combination of elements, including using a choice of a first weighted sum of values for sub-pixels residing at $½^{N-1}$ unit horizontal and unit vertical locations and a second weighted sum of values for sub-pixels residing at unit horizontal and $½^{N-1}$ unit vertical locations and taking a weighted average of the value of a first sub-pixel or pixel situated at a $½^{N-m}$ unit horizontal and $½^{N-n}$ unit vertical location and the value of a second sub-pixel or pixel located at a $½^{N-p}$ unit horizontal and $½^{N-q}$ unit vertical location, variables m, n, p and q taking integer values in the range 1 to N such that the first and second sub-pixels or pixels are located diagonally with respect to the sub-pixel at $½^{N}$ unit horizontal and $½^{N}$ vertical location, was not well-understood, routine, or conventional.

E. U.S. Patent No. 7,280,599 ("the '599 Patent")

84. On October 9, 2007, the U.S. Patent and Trademark Office duly and legally issued U.S. Patent No. 7,280,599 ("the '599 Patent"), entitled "Method for Sub-Pixel Value Interpolation," to inventors Marta Karczewicz and Antti Olli Hallapuro. Nokia owns all rights to the '599 Patent necessary to bring this action. The '599 Patent issued from U.S. Patent Application No. 11/090,717, filed on March 25, 2005, which is a continuation of U.S. Patent Application No. 09/954,608, filed on September 17, 2001, which issued as U.S. Patent No. 6,950,469. A true and correct copy of the '599 Patent is attached hereto as Exhibit 8 and incorporated herein by reference.

85. The '599 Patent is not directed to merely an abstract idea or any patent-ineligible concept. Instead, the '599 Patent is directed to novel and unconventional improvements to motion-compensated prediction in the field of digital video coding. The '599 Patent provides improvements over prior motion compensated prediction and video compression techniques that result in substantial benefits to motion prediction, video compression, video quality, and video playback. These substantial benefits are enjoyed by users of the Accused Products when, for example, watching video over the Internet.

86. A digital video sequence is a sequence of still images with "the illusion of motion being created by displaying consecutive images of the sequence one after the other at a relatively fast frame rate." '599 Patent at 1:18-22. Consecutive images (or frames) often tend to be only slightly different from one another because, for example, the background is stationary or only changes slowly. *Id*. at 1:22-33. Thus, there is often a "considerable amount of redundant information" between consecutive images. *Id*. at 1:22-25.

87. The '599 Patent describes one form of redundancy as "temporal redundancy," in which "the content of some (often many) frames in a video sequence is 'predicted' from other

frames in the sequence by tracing the motion of objects or regions of an image between frames." '599 Patent at 2:62-3:3. A "prediction frame" is created by moving prediction pixels from a reference frame according to motion information, which describes the relationship between pixels in the current frame and their corresponding prediction pixels in the reference frame. *Id*. at 3:49-62. The difference between the current frame being coded and the predicted frame is referred to as the prediction error frame. *Id*. at 3:53-4:1.

88. A video encoder can therefore represent a current frame in a more compact way by representing the frame in terms of the motion information required to form its prediction frame and the prediction error frame. *Id*. at 4:1-7. Thus, the "operating principle of video coders using motion compensation is to minimize the amount of information in a prediction error frame." *Id*. at 3:42-46.

89. However, as the '599 Patent explains, when using motion prediction, full pixel resolution is generally not sufficiently accurate to model real life motion, which has arbitrary precision. *Id*. at 6:29-40. Therefore, sub-pixel resolution is used. *Id*. at 6:40-46. Allowing motion vectors to have sub-pixel resolution adds to the complexity and burden of the encoding and decoding operations that must be performed, in part, because the sub-pixel values must be interpolated from full resolution pixels. *Id*. at 6:40-50, 7:28-30.

90. One problem with sub-pixel value interpolation is maintaining prediction accuracy while also limiting computational complexity and memory usage. For example, prior to the '599 Patent, conventional sub-pixel value interpolation methods used predetermined sets of particular nearby pixels and sub-pixels to interpolate other sub-pixel values, without providing any choice or flexibility and without sufficiently good balancing of interpolation accuracy and computational complexity. *Id*. at 11:25-32. These methods therefore required the unnecessary interpolation and

storage of sub-pixel values that were needed only to interpolate other sub-pixels. *Id*. at 11:25-43. This increased computational complexity and memory requirements, as well as reduced the precision of the interpolated sub-pixel values (due to truncation when storing the values). *Id*. Other methods reduced unnecessary dependencies in the interpolation of certain sub-pixels, but those methods retained numerical precision which required high precision arithmetic and high memory requirement. *Id*. at 13:35-44.

91. The ’599 Patent overcame these technical challenges in the prior systems by inventing a method of sub-pixel value interpolation that improves performance with respect to both computational complexity and memory requirements while maintaining prediction accuracy. The ’599 Patent employs the unconventional solution of providing flexibility and a choice of which pixels and sub-pixels to use in the interpolation of other sub-pixels. ’599 Patent at 38:63-67. For example, the ’599 Patent is superior to prior art methods in that it does not require any ¼ resolution sub-pixels to depend on previously interpolated ¼ resolution sub-pixels. *Id*. at 37:53-58. According to the ’599 Patent, ¼ resolution sub-pixels that have ¼ resolution in both the horizontal and vertical directions are interpolated by a diagonally located pixel and sub-pixel or two diagonally located sub-pixels, which avoids dependency on other ¼ resolution sub-pixels. *Id*. at 19:60-20:2, 14:14-24. This not only reduces the number of calculations required as compared to conventional technology, it also increases the precision by eliminating truncation and clipping that occurs in the intermediate interpolation steps. *Id*. at 37:59-38:4. Similarly, the ’599 Patent provides the unconventional flexibility of a choice for the interpolation of the ½ resolution sub-pixels. *Id*. at 38:51-55. This also results in the minimization of operations required to perform sub-pixel interpolation. *Id*. at 38:55-67.

92. As another example, the ’599 Patent is superior to prior art methods in that it does not require high precision arithmetic to be used in the calculation of all sub-pixels. *Id*. at 38:17-23. The selective use of lower precision arithmetic decreases the computational complexity and increases the speed at which the calculations can be performed. *Id*. at 38:23-41.

93. The ’599 Patent therefore provides a specific technological improvement to the functionality and capabilities of video coding technology that results in increased efficiency and reduced computational complexity and memory requirements.

94. Conventional technology prior to the ’599 Patent was not capable of providing any flexibility or choice of dependencies on pixels and sub-pixels used in sub-pixel interpolation. Conventional technology prior to the ’599 Patent also could not avoid unnecessary interpolation and storage of sub-pixel values that were needed only to interpolate other sub-pixels.

95. The ’599 Patent recognizes and solves these specific technological problems with the conventional technology at the time. The ’599 Patent’s ability to use a choice of either a weighted sum of the values of sub-pixels having co-ordinates $K/2^N$, $L/2^N$ with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the values of sub-pixels having co-ordinates $K/2^N$, $L/2^N$ with K equal to zero and L equal to an even value and the values of sub-pixels having corresponding co-ordinates in immediately adjacent bounded rectangular regions and ability to use a choice of a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates 1/2, 1/2, and a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of both K and L, including zero, situated within a quadrant of the rectangular bounded region defined

by corner pixels having co-ordinates 1/2, 1/2 and the nearest neighbouring pixel was a significant advancement over existing technology.

96. The novel solution of the ’599 Patent, including the flexibility of a choice of which pixels and sub-pixels to use in the interpolation of other sub-pixels, was not well-understood, routine, or conventional, nor was it simply comprised of well-understood, routine, and conventional activities previously known to the industry. Furthermore, the ordered combination of elements, including using a choice of either a weighted sum of the values of sub-pixels having co-ordinates $K/2^N$, $L/2^N$ with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the values of sub-pixels having co-ordinates $K/2^N$, $L/2^N$ with K equal to zero and L equal to an even value and the values of sub-pixels having corresponding co-ordinates in immediately adjacent bounded rectangular regions and using a choice of a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates 1/2, 1/2, and a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of both K and L, including zero, situated within a quadrant of the rectangular bounded region defined by corner pixels having co-ordinates 1/2, 1/2 and the nearest neighbouring pixel, was not well-understood, routine, or conventional.

F. U.S. Patent No. 8,036,273 (“the ’273 Patent”)

97. On October 11, 2011, the U.S. Patent and Trademark Office duly and legally issued U.S. Patent No. 8,036,273 (“the ’273 Patent”), entitled “Method for Sub-Pixel Value Interpolation,” to inventors Marta Karczewicz and Antti Olli Hallapuro. Nokia owns all rights to the ’273 Patent necessary to bring this action. The ’273 Patent issued from U.S. Patent Application No. 11/839,205, filed on August 15, 2007, which is a continuation of U.S. Patent Application No. 11/090,717, filed on March 25, 2005, which issued as U.S. Patent No. 7,280,599, which is a

continuation of U.S. Patent Application No. 09/954,608, filed on September 17, 2001, which issued as U.S. Patent No. 6,950,469. A true and correct copy of the ’273 Patent is attached hereto as Exhibit 9 and incorporated herein by reference.

98. The ’273 Patent is not directed to merely an abstract idea or any patent-ineligible concept. Instead, the ’273 Patent is directed to novel and unconventional improvements to motion-compensated prediction in the field of digital video coding. The ’273 Patent provides improvements over prior motion compensated prediction and video compression techniques that result in substantial benefits to motion prediction, video compression, video quality, and video playback. These substantial benefits are enjoyed by users of the Accused Products when, for example, watching video over the Internet.

99. A digital video sequence is a sequence of still images with “the illusion of motion being created by displaying consecutive images of the sequence one after the other at a relatively fast frame rate.” ’273 Patent at 1:21-25. Consecutive images (or frames) often tend to be only slightly different from one another because, for example, the background is stationary or only changes slowly. *Id*. at 1:25-35. Thus, there is often a “considerable amount of redundant information” between consecutive images. *Id*. at 1:25-28.

100. The ’273 Patent describes one form of redundancy as “temporal redundancy,” in which “the content of some (often many) frames in a video sequence is ‘predicted’ from other frames in the sequence by tracing the motion of objects or regions of an image between frames.” ’273 Patent at 2:62-3:2. A “prediction frame” is created by moving prediction pixels from a reference frame according to motion information, which describes the relationship between pixels in the current frame and their corresponding prediction pixels in the reference frame. *Id*. at 3:47-

60. The difference between the current frame being coded and the predicted frame is referred to as the prediction error frame. *Id*. at 3:51-66.

101. A video encoder can therefore represent a current frame in a more compact way by representing the frame in terms of the motion information required to form its prediction frame and the prediction error frame. *Id*. at 3:66-4:5. Thus, the "operating principle of video coders using motion compensation is to minimize the amount of information in a prediction error frame." *Id*. at 3:40-44.

102. However, as the '273 Patent explains, when using motion prediction, full pixel resolution is generally not sufficiently accurate to model real life motion, which has arbitrary precision. *Id*. at 6:24-35. Therefore, sub-pixel resolution is used. *Id*. at 6:35-41. Allowing motion vectors to have sub-pixel resolution adds to the complexity and burden of the encoding and decoding operations that must be performed, in part, because the sub-pixel values must be interpolated from full resolution pixels. *Id*. at 6:35-45, 7:20-23.

103. One problem with sub-pixel value interpolation is maintaining prediction accuracy while also limiting computational complexity and memory usage. For example, prior to the '273 Patent, conventional sub-pixel value interpolation methods used predetermined sets of particular nearby pixels and sub-pixels to interpolate other sub-pixel values, without providing any choice or flexibility and without sufficiently good balancing of interpolation accuracy and computational complexity. *Id*. at 11:6-13. These methods therefore required the unnecessary interpolation and storage of sub-pixel values that were needed only to interpolate other sub-pixels. *Id*. at 11:6-24. This increased computational complexity and memory requirements, as well as reduced the precision of the interpolated sub-pixel values (due to truncation when storing the values). *Id*. Other methods reduced unnecessary dependencies in the interpolation of certain sub-pixels, but

those methods retained numerical precision which required high precision arithmetic and high memory requirement. *Id*. at 13:9-18.

104. The ’273 Patent overcame these technical challenges in the prior systems by inventing a method of sub-pixel value interpolation that improves performance with respect to both computational complexity and memory requirements while maintaining prediction accuracy. The ’273 Patent employs the unconventional solution of providing flexibility and a choice of which pixels and sub-pixels to use in the interpolation of other sub-pixels. ’273 Patent at 44:56-59. For example, the ’273 Patent is superior to prior art methods in that it does not require any ¼ resolution sub-pixels to depend on previously interpolated ¼ resolution sub-pixels. *Id*. at 43:47-52. According to the ’273 Patent, ¼ resolution sub-pixels that have ¼ resolution in both the horizontal and vertical directions are interpolated by a diagonally located pixel and sub-pixel or two diagonally located sub-pixels, which avoids dependency on other ¼ resolution sub-pixels. *Id*. at 21:47-60, 13:54-64. This not only reduces the number of calculations required as compared to conventional technology, it also increases the precision by eliminating truncation and clipping that occurs in the intermediate interpolation steps. *Id*. at 43:52-64. Similarly, the ’273 Patent provides the unconventional flexibility of a choice for the interpolation of the ½ resolution sub-pixels. *Id*. at 44:43-47. This also results in the minimization of operations required to perform sub-pixel interpolation. *Id*. at 44:48-51.

105. As another example, the ’273 Patent is superior to prior art methods in that it does not require high precision arithmetic to be used in the calculation of all sub-pixels. *Id*. at 44:10-16. The selective use of lower precision arithmetic decreases the computational complexity and increases the speed at which the calculations can be performed. *Id*. at 44:16-31.

106. The ’273 Patent therefore provides a specific technological improvement to the functionality and capabilities of video coding technology that results in increased efficiency and reduced computational complexity and memory requirements.

107. Conventional technology prior to the ’273 Patent was not capable of providing any flexibility or choice of dependencies on pixels and sub-pixels used in sub-pixel interpolation. Conventional technology prior to the ’273 Patent also could not avoid unnecessary interpolation and storage of sub-pixel values that were needed only to interpolate other sub-pixels.

108. The ’273 Patent recognizes and solves these specific technological problems with the conventional technology at the time. The ’273 Patent’s ability to use a choice of either a weighted sum of the values of sub-pixels having co-ordinates $K/2^N$, $L/2^N$ with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the values of sub-pixels having co-ordinates $K/2^N$, $L/2^N$ with K equal to zero and L equal to an even value and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions and ability to use a choice of either a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates ½, ½, or a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of K and L, including zero, situated within a quadrant of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates ½, ½ and the nearest neighbouring pixel was a significant advancement over existing technology.

109. The novel solution of the ’273 Patent, including the flexibility of a choice of which pixels and sub-pixels to use in the interpolation of other sub-pixels, was not well-understood, routine, or conventional, nor was it simply comprised of well-understood, routine, and

conventional activities previously known to the industry. Furthermore, the ordered combination of elements, including using a choice of either a weighted sum of the values of sub-pixels having co-ordinates $K/2^N$, $L/2^N$ with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the values of sub-pixels having co-ordinates $K/2^N$, $L/2^N$ with K equal to zero and L equal to an even value and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions and using a choice of either a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates ½, ½, or a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of K and L, including zero, situated within a quadrant of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates ½, ½ and the nearest neighbouring pixel, was not well-understood, routine, or conventional.

G. U.S. Patent No. 6,856,701 ("the '701 Patent")

110. On February 15, 2005, the U.S. Patent and Trademark Office duly and legally issued U.S. Patent No. 6,856,701 ("the '701 Patent"), entitled "Method and System for Context-Based Adaptive Binary Arithmetic Coding," to inventors Marta Karczewicz and Ragip Kurceren. Nokia owns all rights to the '701 Patent necessary to bring this action. The '701 Patent issued from U.S. Application No. 09/995,240, filed on November 27, 2001, which claims priority to Provisional Application Number 60/322,112, filed on September 14, 2001. A true and correct copy of the '701 Patent is attached hereto as Exhibit 10 and incorporated herein by reference.

111. The '701 Patent is not directed to merely an abstract idea or any patent-ineligible concept. Instead, the '701 Patent is directed to novel and unconventional improvements to context-based adaptive binary arithmetic coding of digital video. The '701 Patent provides improvements over prior context-based adaptive binary arithmetic coding techniques that result

in substantial benefits to video compression, video quality, and video playback. These substantial benefits are enjoyed by users of the Accused Products when, for example, watching video over the Internet.

112. Digital video is formed from sequential still images. A still image in uncompressed form comprises an array of image pixels. '701 Patent at 1:17-18. In still image compression, "an image to be coded is first divided into an array of non-overlapping square blocks." *Id*. at 2:59-63. Transform coding is then applied to the image blocks to convert the pixel values to a set of coefficient values, which are then quantized "to further reduce the amount of data (i.e., number of bits) required" to represent the image blocks. *Id*. at 2:63-3:6.

113. The quantized transform coefficients are then scanned in a particular order to convert the two-dimensional array of quantized transform coefficients into a one-dimensional array. *Id*. at 5:56-62. The one-dimensional array is then converted into level and run pairs. That is, each non-zero quantized transform coefficient is represented by two values: level and run. *Id*. at 6:7-9. "Level is the value of the quantized coefficient and run is the number of consecutive zero-valued coefficients preceding the coefficient in question." *Id*. at 6:9-12.

114. These level and run pairs are then further compressed using entropy coding, in which a variable number of bits is assigned such that values of level and run pairs that are more likely to occur than other values are represented by code-words having fewer bits. *Id*. at 6:19-32. One type of such entropy coding is called "Context-based Adaptive Binary Arithmetic Coding (CABAC)," which is "a form of binary arithmetic coding which continually adapts to the statistical properties of the information being coded." *Id*. at 11:63-12:6. More specifically, in the CABAC method, "data symbols to be coded which have non-binary values are first converted to binary values," also called a "sequence of bins" each of which has a value of either a 0 or 1. *Id*.

at 12:42-49. Each of the bins is then assigned a "context" based on having similar statistics. For example, "each bin assigned to a particular context is assumed to have a similar probability of containing the value 1 or 0 as the other bins belonging to that context." *Id*. at 13:1-4. Then, "probability estimates used to generate code-words in the arithmetic coder are defined for each context rather than for each possible bin to be encoded." *Id*. at 13:4-7.

115. Prior to the '701 Patent, CABAC methods were "not optimal with regard to coding efficiency." *Id*. at 15:55-62. That is, the inventors of the '701 Patent "determined that certain relationships exist between the run and level values associated with DCT transform coefficients," as well as significant similarities between consecutive level values. *Id*. at 16:2-20. The inventors recognized that "these relationships can be used to construct improved context models which enable the CABAC method to operate with improved coding efficiency when applied to the run and level values." *Id*. at 16:2-10.

116. The '701 Patent overcame the shortcomings in the prior systems by inventing a new context model which takes into account the relationships between level values and between level and run values. *Id*. at 16:21-24. The '701 Patent employs the unconventional solution of assigning contexts taking into account the level value of another run-level pair. *Id*. at 15:64-16:2, 23:15-26. Because the inventors "determined that consecutive level values exhibit a significant similarity," the improved method of coding of the '701 Patent exhibited up to 4.74% reduction in bitrate. *Id*. at 16:10-16, 25:37-45.

117. The '701 Patent therefore provides a specific technological improvement to the functionality and capabilities of video coding technology that results in more reliable and efficient operations with superior bitrate savings. *Id*. at 16:10-16, 25:37-45.

118. Conventional technology prior to the ’701 Patent was not capable of taking into account relationships that exist between the level values or between the run and level values associated with DCT transform coefficients.

119. The ’701 Patent recognizes and solves these specific technological problems with the conventional technology at the time. The ’701 Patent’s ability to assign the first numbers of a run-level number pair to one of a plurality of contexts representative of the first numbers such that the first number of a first number pair is assigned to a context at least partly in dependence on a first number of a second number pair was a significant advancement over existing technology.

120. The novel solution of the ’701 Patent, including assigning contexts taking into account the level value of another run-level pair, was not well-understood, routine, or conventional, nor was it simply comprised of well-understood, routine, and conventional activities previously known to the industry. Furthermore, the ordered combination of elements, including assigning the first numbers of a run-level number pair to one of a plurality of contexts representative of the first numbers such that the first number of a first number pair is assigned to a context at least partly in dependence on a first number of a second number pair, was not well-understood, routine, or conventional.

H. U.S. Patent No. 9,800,891 (“the ’891 Patent”)

121. On October 24, 2017, the U.S. Patent and Trademark Office duly and legally issued U.S. Patent No. 9,800,891 (“the ’891 Patent”), entitled “Method and Associated Device for Filtering Digital Video Images,” to inventors Ossi Kalevo, Emre Aksu, and Marta Karczewicz. The ’891 Patent issued from U.S. Patent Application No. 09/766,035, filed on January 19, 2001, and claims priority to Finnish Patent Application Number 20000120, filed on January 20, 2000.

Nokia owns all rights to the ’891 Patent necessary to bring this action. A true and correct copy of the ’891 Patent is attached hereto as Exhibit 11 and incorporated herein by reference.

122. The ’891 Patent is not directed to merely an abstract idea or any patent-ineligible concept. Instead, the ’891 Patent is directed to novel and unconventional improvements to reducing visual artefacts at the block boundaries in a frame of a digital video. The ’891 Patent provides improvements to filtering techniques used to reduce visual artefacts due to block boundaries that result in substantial benefits to video compression, video quality, and video playback. These substantial benefits are enjoyed by users of the Accused Products when, for example, watching video over the Internet.

123. Digital video is formed from sequential still images or frames. Those frames are further divided into so-called “blocks.” ’891 Patent at 2:4-5. There are generally four types of blocks, where the type specifies a method of coding: (i) intra coding, (ii) copy coding, (iii) motion-compensated prediction coding, and (iv) not-coded coding. *Id*. at 2:5-24. Motion-compensated prediction coding uses prediction error, which is the difference between the pixel values of the actual frame and a reconstructed frame formed by using the motion compensated prediction. *Id*. at 2:12-19. In order to reduce the number of bits needed to represent the image, the prediction error blocks are further transform coded and quantized. *Id*. at 3:15-25.

124. As the ’891 Patent explains, “[q]uantization causes rounding errors, which can become visible in an image reconstructed from blocks, so that there is a discontinuity of pixel values at the boundary between adjacent blocks.” *Id*. at 3:25-29. These errors cause visible edges in the image and are called blocking artefacts. *Id*. at 3:29-34.

125. Prior to the ’891 Patent, methods that were used to reduce or remove blocking artefacts considered “the difference between the values of pixels across the block boundary, the

size of the quantization step of the coefficients received as the transformation result, and the difference of pixel values on different sides of the pixel being processed." *Id*. at 3:35-53. One significant problem with these methods was that they "tend to remove lines that belong to real features of the image" while "not always [being] capable of removing all blocking artefacts." *Id*. at 3:54-57.

126. The '891 Patent overcame these technical challenges in the prior systems by inventing a new kind of filtering arrangement that adjusts filtering parameters according to the type of blocks whose boundary is to be filtered. *Id*. at 3:61-4:1. The '891 Patent employs the unconventional solution of choosing different filtering parameters "according to the type of block on either side of the boundary in order to yield an improved filtering result." *Id*. at 4:1-3. "Because blocking artefacts only occur at block boundaries, according to the invention, filtering is advantageously only applied to pixels at block boundaries and the vicinity thereof." *Id*. at 4:41-43. That is, "only pixels containing blocking artefacts are selected for corrective filtering" and "the quality of edges that are part of the image itself is not affected during filtering." *Id*. at 4:45-49. This is accomplished by flexibly selecting the number of pixels for filtering based on the type of block on either side of the block boundary. *Id*. at 5:15-30, 4:4-8.

127. The '891 Patent therefore provides a specific technological improvement to the functionality and capabilities of video coding technology that results in more reliable and efficient operations with better video quality. *Id*. at 5:21-28, 3:63-65, 4:4-8. "That means that a larger amount of blocking artefacts can be removed without weakening the real image edges unreasonably." *Id*. at 5:28-30.

128. Conventional technology prior to the '891 Patent was not capable of adjusting filtering parameters according to the type of blocks whose boundary is to be filtered.

129. The ’891 Patent recognizes and solves these specific technological problems with the conventional technology at the time. The ’891 Patent’s ability to provide information on the first and second prediction encoding methods to a block boundary filter and ability to determine, by the block boundary filter, a first number of pixels to be examined on the first side of the block boundary and a second number of pixels to be examined on the second side of the block boundary as a parameter of the adaptive block boundary filtering operation, based on the types of the first and the second prediction encoding methods was a significant advancement over existing technology.

130. The novel solution of the ’891 Patent, including choosing different block boundary filtering parameters according to the type of block on either side of the boundary, was not well-understood, routine, or conventional, nor was it simply comprised of well-understood, routine, and conventional activities previously known to the industry. Furthermore, the ordered combination of elements, including providing information on the first and second prediction encoding methods to a block boundary filter and determining, by the block boundary filter, a first number of pixels to be examined on the first side of the block boundary and a second number of pixels to be examined on the second side of the block boundary as a parameter of the adaptive block boundary filtering operation, based on the types of the first and the second prediction encoding methods, was not well-understood, routine, or conventional.

I. U.S. Patent No. 6,968,005 (“the ’005 Patent”)

131. On November 22, 2005, the U.S. Patent and Trademark Office duly and legally issued U.S. Patent No. 6,968,005 (“the ’005 Patent”), entitled “Video Coding,” to inventor Miska Hannuksela. Nokia owns all rights to the ’005 Patent necessary to bring this action. The ’005 Patent issued from U.S. Application No. 09/855,640, filed on May 15, 2001, and claims priority

to Great Britain Patent Application Number 0011597, filed on May 15, 2000. A true and correct copy of the ’005 Patent is attached hereto as Exhibit 12 and incorporated herein by reference.

132. The ’005 Patent is not directed to merely an abstract idea or any patent-ineligible concept. Instead, the ’005 Patent is directed to novel and unconventional improvements to error detection when using motion-compensated prediction in the field of digital video coding. The ’005 Patent provides improvements over prior motion compensated prediction and video compression techniques that result in substantial benefits to motion prediction, video compression, video quality, and video playback. These substantial benefits are enjoyed by users of the Accused Products when, for example, watching video over the Internet.

133. A video sequence consists of a series of still pictures or frames, and “objects appearing in a previous image are also likely to appear in the current image.” ’005 Patent at 1:6-7, 1:14-16. For example, in a particular scene, the background may change very little, or a car may drive by, where the car itself does not change and merely moves across the screen. This type of redundancy is called “temporal redundancy.” *Id*. at 1:14-16. Video “[c]ompression can be achieved by taking advantage of this temporal redundancy and predicting the current picture from another picture, termed an anchor or reference picture.” *Id*. at 1:16-19.

134. As the ’005 Patent explains, not all pictures use temporal redundancy. “A compressed video clip typically consists of a sequence of pictures, which can be roughly categorized into temporally independent INTRA pictures and temporally differentially coded INTER pictures.” *Id*. at 1:59-1:62. That is, INTRA pictures do not use temporal redundancy and therefore do not rely on a temporal reference frame. Whereas, INTER pictures do use temporal redundancy and therefore rely on a temporal reference frame.

135. Similarly, not all pictures are used as reference pictures, *i.e.*, other pictures are not predicted from them. *Id*. at 1:48-50. Pictures that are not used as reference pictures "can be discarded (intentionally or unintentionally) without impacting the picture quality of future pictures." *Id*. at 1:50-52. Reference pictures, on the other hand, are used to predict other pictures. Therefore, an error in a reference picture "is propagated both spatially and temporally," which means that "once an error occurs, it is easily visible to the human eye for a relatively long time." *Id*. at 2:8-13.

136. Prior to the '005 Patent, one significant problem was that "the bit-stream does not include information identifying the reference picture," such that there is "no means to detect if a reference picture is lost." '005 Patent at 3:35-39. Therefore, there was no way to know if a picture error would propagate into future pictures and substantially degrade the quality of the video (as in the case with a reference picture) or whether the picture error could simply be ignored (as in the case with a non-reference picture). *Id*. at 3:52-55. Prior systems could not differentiate the two scenarios and instead would freeze until the next INTRA frame was received or expend resources to perform error concealment techniques. *Id*. at 3:56-62, 2:29-2:64.

137. The '005 Patent overcame these technical challenges in the prior systems by inventing a novel sequence indicator with an independent numbering scheme to identify reference pictures and to enable the differentiation between a loss of a reference picture from a loss of a non-reference picture. *Id*. at 4:3-12, 15:2-17. The '005 Patent employs the unconventional solution of incrementing a new sequence indicator each time a reference picture is encoded so that it is possible to differentiate between errors in or loss of a reference picture versus a non-reference picture. *Id*. at 4:13-19.

138. The ’005 Patent therefore provides a specific technological improvement to the functionality and capabilities of video coding technology that results in increased efficiency and video quality because the decoder no longer must freeze and wait for an INTRA frame when a non-reference picture has an error or is lost. *Id*. at 4:23-25, 4:34-37.

139. Conventional technology prior to the ’005 Patent was not capable of identifying and distinguishing the loss of a reference frame from a non-reference frame, and therefore could not take the action appropriate to compensate for the loss of a reference frame as compared to a non-reference frame.

140. The ’005 Patent recognizes and solves these specific technological problems with the conventional technology at the time. The ’005 Patent’s ability to use a sequence indicator having an independent numbering scheme, such that consecutive pictures used to form reference pictures in encoding order are assigned sequence indicator values that differ with respect to each other by a predetermined amount independent of the number of non-reference pictures encoded between successive reference pictures was a significant advancement over existing technology.

141. The novel solution of the ’005 Patent, including a sequence indicator with an independent numbering scheme to identify reference pictures, was not well-understood, routine, or conventional, nor was it simply comprised of well-understood, routine, and conventional activities previously known to the industry. Furthermore, the ordered combination of elements, including using a sequence indicator having an independent numbering scheme, such that consecutive pictures used to form reference pictures in encoding order are assigned sequence indicator values that differ with respect to each other by a predetermined amount independent of the number of non-reference pictures encoded between successive reference pictures, was not well-understood, routine, or conventional.

J. U.S. Patent No. 8,144,764 ("the '764 Patent")

142. On March 27, 2012, the U.S. Patent and Trademark Office duly and legally issued U.S. Patent No. 8,144,764 ("the '764 Patent"), entitled "Video Coding," to inventor Miska Hannuksela. Nokia owns all rights to the '764 Patent necessary to bring this action. The '764 Patent issued from U.S. Application No. 11/242,888, filed on October 5, 2005, which is a continuation of U.S. Application No. 09/855,640, filed on May 15, 2001, now U.S. Patent No. 6,968,005, which claims priority to Great Britain Patent Application Number 0011597.2, filed on May 15, 2000. A true and correct copy of the '764 Patent is attached hereto as Exhibit 13 and incorporated herein by reference.

143. The '764 Patent is not directed to merely an abstract idea or any patent-ineligible concept. Instead, the '764 Patent is directed to novel and unconventional improvements to error detection when using motion-compensated prediction in the field of digital video coding. The '764 Patent provides improvements over prior motion compensated prediction and video compression techniques that result in substantial benefits to motion prediction, video compression, video quality, and video playback. These substantial benefits are enjoyed by users of the Accused Products when, for example, watching video over the Internet.

144. A video sequence consists of a series of still pictures or frames, and "objects appearing in a previous image are also likely to appear in the current image." '764 Patent at 1:12-13, 1:20-22. For example, in a particular scene, the background may change very little, or a car may drive by, where the car itself does not change and merely moves across the screen. This type of redundancy is called "temporal redundancy." *Id*. at 1:20-22. Video "[c]ompression can be achieved by taking advantage of this temporal redundancy and predicting the current picture from another picture, termed an anchor or reference picture." *Id*. at 1:22-25.

145. As the ’764 Patent explains, not all pictures use temporal redundancy. “A compressed video clip typically consists of a sequence of pictures, which can be roughly categorized into temporally independent INTRA pictures and temporally differentially coded INTER pictures.” *Id*. at 1:65-2:1. That is, INTRA pictures do not use temporal redundancy and therefore do not rely on a temporal reference frame. Whereas, INTER pictures do use temporal redundancy and therefore rely on a temporal reference frame.

146. Similarly, not all pictures are used as reference pictures, *i.e.*, other pictures are not predicted from them. *Id*. at 1:54-55. Pictures that are not used as reference pictures “can be discarded (intentionally or unintentionally) without impacting the picture quality of future pictures.” *Id*. at 1:56-57. Reference pictures, on the other hand, are used to predict other pictures. Therefore, an error in a reference picture “is propagated both spatially and temporally,” which means that “once an error occurs, it is easily visible to the human eye for a relatively long time.” *Id*. at 2:15-20.

147. Prior to the ’764 Patent, one significant problem was that “the bit-stream does not include information identifying the reference picture,” such that there is “no means to detect if a reference picture is lost.” ’764 Patent at 3:40-43. Therefore, there was no way to know if a picture error would propagate into future pictures and substantially degrade the quality of the video (as in the case with a reference picture) or whether the picture error could simply be ignored (as in the case with a non-reference picture). *Id*. at 3:55-58. Prior systems could not differentiate the two scenarios and instead would freeze until the next INTRA frame was received or expend resources to perform error concealment techniques. *Id*. at 3:59-65, 2:36-3:2.

148. The ’764 Patent overcame these technical challenges in the prior systems by inventing a novel sequence indicator with an independent numbering scheme to identify reference

pictures and to enable the differentiation between a loss of a reference picture from a loss of a non-reference picture. *Id*. at 4:7-15, 14:47-56. The ’764 Patent employs the unconventional solution of incrementing a new sequence indicator each time a reference picture is encoded so that it is possible to differentiate between errors in or loss of a reference picture versus a non-reference picture. *Id*. at 4:16-21.

149. The ’764 Patent therefore provides a specific technological improvement to the functionality and capabilities of video coding technology that results in increased efficiency and video quality because the decoder no longer must freeze and wait for an INTRA frame when a non-reference picture has an error or is lost. *Id*. at 4:25-27, 4:35-38.

150. Conventional technology prior to the ’764 Patent was not capable of identifying and distinguishing the loss of a reference frame from a non-reference frame, and therefore could not take the action appropriate to compensate for the loss of a reference frame as compared to a non-reference frame.

151. The ’764 Patent recognizes and solves these specific technological problems with the conventional technology at the time. The ’764 Patent’s ability to use an independent numbering scheme, to assign consecutive reference pictures in encoding order with respective sequence indicator values that differ with respect to each other by a predetermined amount independent of one or more of the number of non-reference pictures encoded between consecutive reference pictures and the number of non-coded pictures between consecutive reference pictures was a significant advancement over existing technology.

152. The novel solution of the ’764 Patent, including a sequence indicator with an independent numbering scheme to identify reference pictures, was not well-understood, routine, or conventional, nor was it simply comprised of well-understood, routine, and conventional

activities previously known to the industry. Furthermore, the ordered combination of elements, including using an independent numbering scheme, to assign consecutive reference pictures in encoding order with respective sequence indicator values that differ with respect to each other by a predetermined amount independent of one or more of the number of non-reference pictures encoded between consecutive reference pictures and the number of non-coded pictures between consecutive reference pictures, was not well-understood, routine, or conventional.

K. U.S. Patent No. 8,175,148 ("the '148 Patent")

153. On May 8, 2012, the U.S. Patent and Trademark Office duly and legally issued U.S. Patent No. 8,175,148 ("the '148 Patent"), entitled "Method and Device for Indicating Quantizer Parameters in a Video Coding System," to inventor Jani Lainema. Nokia owns all rights to the '148 Patent necessary to bring this action. The '148 Patent issued from U.S. Application No. 11/881,367, filed on July 26, 2007, which is a division of U.S. Application No. 10/421,629, filed on April 23, 2003, now U.S. Patent No. 7,263,125, which claims priority to Provisional Application Number 60/374,667, filed on April 23, 2002. A true and correct copy of the '148 Patent is attached hereto as Exhibit 14 and incorporated herein by reference.

154. The '148 Patent is not directed to merely an abstract idea or any patent-ineligible concept. Instead, the '148 Patent is directed to novel and unconventional improvements to providing quantization parameter values used in motion-compensated prediction in the field of digital video coding. The '148 Patent provides improvements over prior motion compensated prediction and video compression techniques that result in substantial benefits to motion prediction, video compression, video quality, and video playback. These substantial benefits are enjoyed by users of the Accused Products when, for example, watching video over the Internet.

155. Digital video is comprised of "still images, the illusion of motion being created by displaying the images one after the other" at a relatively fast rate. '148 Patent at 1:20-24. The

term "temporal redundancy" refers to the fact that objects appearing in one image or frame of a video are likely to appear in subsequent frames. *Id*. at 2:55-59. A technique called "motion-compensated prediction" takes advantage of temporal redundancy and predicts a current frame from a previous ("reference") frame "by tracking the motion of objects or regions of an image between a frame to be coded (compressed) and the reference frame(s) using 'motion vectors'." *Id*. at 3:13-23. As the '148 Patent explains, "[m]otion-compensated prediction alone rarely provides a sufficiently precise representation of the image content of a video frame and therefore it is typically necessary to provide a so-called 'prediction error' (PE) frame" with each temporally coded frame." *Id*. at 3:29-33. The "prediction error frame comprises values that represent the difference between pixel values in the frame to be coded and corresponding reconstructed pixel values formed on the basis of a predicted version of the frame in question." *Id*. at 3:36-40.

156. As described in the '148 Patent, the prediction error values undergo transform coding and quantization in order to reduce the number of bits to be coded. *Id*. at 6:10-18, 4:39-50. However, "in order to stay in synchronization with the encoder, the decoder has to know the exact value of the QP [quantization parameter] used in the coded video sequence." *Id*. at 10:33-35. Prior techniques indicated the QP value "in the encoded bit-stream at the beginning of each picture" and then "also indicated [the QP value] at the beginning of each slice of the frame." *Id*. at 10:44-62.

157. Prior to the '148 Patent, one significant problem was that transmitting the QP value increased the number of bits needed to encode the image. *Id*. at 10:35-37. More specifically, in prior systems, "1.2 kbps is spent for the QP information alone." *Id*. at 10:40-43. Thus, prior systems were "very costly in terms of the number of bits required to represent the quantization parameter information." *Id*. at 10:62-67. Additionally, since prior systems allowed the value of

QP to vary at the macroblock level, QP information in the bit-stream represented "a significant proportion of the overall available bandwidth." *Id*. at 11:3-17.

158. The '148 Patent overcame these technical challenges in the prior systems by inventing a default or reference level of QP that could be applied to multiple pictures. The '148 Patent employs the unconventional solution of using a default level of QP that only needs to be transmitted with a sequence of multiple frames. *Id*. at 11:25-36. Because the default level of QP does not have to be transmitted at the beginning of every picture and every slice, there is a substantial reduction in transmission bit-rate, as well as a reduction in the time needed to transmit the encoded video sequence. *Id*.

159. The '148 Patent therefore provides a specific technological improvement to the functionality and capabilities of video coding technology that results in increased efficiency and significant reduction in transmission bit-rate. *Id*. at 11:37-42 (from 1.2 kpbs to 0.2 kpbs, in one example).

160. Conventional technology prior to the '148 Patent was not capable of transmitting or receiving a default or reference level of QP that could be applied to multiple pictures. Conventional technology prior to the '148 Patent was also not capable of transmitting a difference between a level of quantization and the default level of quantization at the beginning of each picture or each slice.

161. The '148 Patent recognizes and solves these specific technological problems with the conventional technology at the time. The '148 Patent's ability to define a default level of quantization for use in encoding of the digital video sequence to quantize the sets of transform coefficient values and ability to provide an indication of the default level of quantization to a decoding process was a significant advancement over existing technology.

162. The novel solution of the ’148 Patent, including providing an indication of a default level of QP used for a plurality of pictures, was not well-understood, routine, or conventional, nor was it simply comprised of well-understood, routine, and conventional activities previously known to the industry. Furthermore, the ordered combination of elements, including defining a default level of quantization for use in encoding of the digital video sequence to quantize the sets of transform coefficient values and providing an indication of the default level of quantization to a decoding process, was not well-understood, routine, or conventional.

L. U.S. Patent No. 8,077,991 (“the ’991 Patent”)

163. On December 13, 2011, the U.S. Patent and Trademark Office duly and legally issued U.S. Patent No. 8,077,991 (“the ’991 Patent”), entitled “Spatially Enhanced Transform Coding,” to inventor Jani Lainema. Nokia owns all rights to the ’991 Patent necessary to bring this action. The ’991 Patent issued from U.S. Application No. 12/101,019, filed on April 10, 2008, and claims priority to Provisional Application Number 60/911,480, filed on April 12, 2007. A true and correct copy of the ’991 Patent is attached hereto as Exhibit 15 and incorporated herein by reference.

164. The ’991 Patent is not directed to merely an abstract idea or any patent-ineligible concept. Instead, the ’991 Patent provides improvements over conventional video coding techniques that result in substantial benefits to video compression, video quality, and video playback. These substantial benefits are enjoyed by users of the Accused Products when, for example, watching video over the Internet.

165. Encoders compress video into a representation suited for storage or transmission. ’991 Patent at 1:21-24. Decoders can decompress the compressed video representation into viewable form. *Id.* As described in the ’991 Patent, typical codecs encode video information in two phases. In the first phase, pixel values can be predicted, for example, by motion compensation

mechanisms, which involve finding and indicating an area in one of the previously coded video frames that corresponds closely to the block being coded. *Id*. at 1:29-34. Additionally, pixel values can be predicted via spatial mechanisms, which involve using the pixel values around the block to be coded in a specified manner. *Id*. at 1:35-37. The second phase involves coding the prediction error (*i.e.*, the difference between the predicted block of pixels and the original block of pixels), which involves transforming the difference in pixel values using a specified transform (*e.g.*, a Discrete Cosine Transform (DCT) or a variant thereof), quantizing the coefficients, and entropy coding the quantized coefficients. *Id*. at 1:37-44.

166. Prior to the ’991 Patent, one significant problem was that transform coding was only efficient under certain circumstances. More specifically, when the prediction error values were less correlated with one another, the transform coding performance deteriorated and caused suboptimal performance. *Id*. at 2:29-38.

167. The ’991 Patent overcame these technical challenges in the prior systems by inventing a novel method of using both transform coding and spatial coding to construct the prediction error signal. *Id*. at 2:64-3:2. The ’991 Patent employs the unconventional solution of performing both transform coding and spatial coding to allow for the efficient spatial representation of those components of the prediction error signal of the same image block that are not well correlated with the transform basis functions (such as certain types of sensor noise, high frequency texture and edge information). *Id*. at 3:18-21, 3:6-12.

168. The ’991 Patent therefore provides a specific technological improvement to the functionality and capabilities of video coding technology that results in increased efficiency and video quality. *Id*. at 3:35-37. For instance, the ’991 Patent describes an example where four scalar values are to be coded or decoded, in which the invention of the ’991 Patent results in compression

efficiency improvement, as the signal is represented by a single transform coefficient and a single spatial sample instead of four transform coefficients. *Id*. at 4:44-58.

169. Conventional technology prior to the ’991 Patent was not capable of using both transform coding and spatial coding to construct the prediction error signal to allow for the efficient spatial representation of those components of the prediction error signal of the same image block that are not well correlated with the transform basis functions.

170. The ’991 Patent recognizes and solves these specific technological problems with the conventional technology at the time. The ’991 Patent’s ability to perform transform coding to the difference signal, perform spatial coding to the difference signal, and join the two representations to form a coded prediction error signal was a significant advancement over existing technology.

171. The novel solution of the ’991 Patent, including performing transform coding to the difference signal, performing spatial coding to the difference signal, and joining the two representations to form a coded prediction error signal, was not well-understood, routine, or conventional, nor was it simply comprised of well-understood, routine, and conventional activities previously known to the industry. Furthermore, the ordered combination of elements, including performing transform coding to the difference signal, performing spatial coding to the difference signal, and joining the two representations to form a coded prediction error signal, was not well-understood, routine, or conventional.

M. U.S. Patent No. 9,571,833 (“the ’833 Patent”)

172. On February 14, 2017, the U.S. Patent and Trademark Office duly and legally issued U.S. Patent No. 9,571,833 (“the ’833 Patent”), entitled “Method for Coding and an Apparatus,” to inventors Mehmet Oguz Bici, Jani Lainema, and Kemal Ugur. Nokia owns all rights to the ’833 Patent necessary to bring this action. The ’833 Patent issued from U.S.

Application No. 13/666,680, filed on November 1, 2012, and claims priority to U.S. Provisional Application No. 61/555,703, filed November 4, 2011. A true and correct copy of the ’833 Patent is attached hereto as Exhibit 16 and incorporated herein by reference.

173. The ’833 Patent is not directed to merely an abstract idea or any patent-ineligible concept. Instead, the ’833 Patent provides improvements over conventional video coding motion compensation techniques that result in substantial benefits to video compression, video quality, and video playback. These substantial benefits are enjoyed by users of the Accused Products when, for example, watching video over the Internet.

174. Encoders compress video into a representation suitable for storage or transmission. ’833 Patent at 1:21-25. Decoders can decompress the compressed video representation into viewable form. *Id*. As described in the ’833 Patent, typical codecs encode video information in two phases. In the first phase, pixel values can be predicted, for example, by motion compensation mechanisms, which involve finding and indicating an area in one of the previously coded video frames that corresponds closely to the block being coded. *Id*. at 1:32-38. Additionally, pixel values can be predicted via spatial mechanisms, which involve using the pixel values around the block to be coded in a specified manner. *Id*. at 1:38-41. The second phase involves coding the prediction error (*i.e.*, the difference between the predicted block of pixels and the original block of pixels), which involves transforming the difference in pixel values using a specified transform (*e.g.*, a Discrete Cosine Transform (DCT) or a variant thereof), quantizing the coefficients, and entropy coding the quantized coefficients. *Id*. at 1:47-53.

175. As described in the ’833 Patent, motion information is indicated by motion vectors associated with each motion compensated image block. *Id*. at 2:48-54. One method used by conventional systems to create motion vectors was to predict them in a predefined manner, for

example by calculating the median of the encoded or decoded motion vectors of the adjacent blocks. *Id*. at 2:56-60. Another method used by conventional systems was to generate a list or a set of candidate predictions from blocks in the current frame and/or co-located blocks in temporal reference pictures and signaling the chosen candidate as the motion vector prediction. *Id*. at 2:61-65.

176. Prior to the ’833 Patent, one significant problem was that after a list of the motion vector prediction candidates was generated, some of the motion vector prediction candidates may have the same motion information, which created redundancy. *Id*. at 3:55-59. Another problem arose when temporal motion vector prediction information was unavailable, for example, due to loss of a reference frame. *Id*. at 3:59-63. Conventional technology did not know if the temporal motion vector prediction candidates in the list should be removed. *Id*. Therefore, methods that determined the inclusion or removal of motion vector prediction candidates based on comparing motion information with temporal motion vector prediction, resulted in the false assignment of motion vector prediction candidates, which caused degradation in picture quality. *Id*. at 3:63-4:3.

177. The ’833 Patent overcame these technical challenges in the prior systems by inventing a method that recognized that the size of the motion vector prediction candidates list could be reduced and associated signaling cost could be reduced by eliminating motion vector prediction candidates based on the geometry of the region being predicted and by using a limited number of candidate comparisons. *Id*. at 4:7-27. The ’833 Patent employs the unconventional solution of obtaining spatial candidates from the motion information of spatial neighbour blocks, for example, and performing a limited number of motion information comparisons between candidate pairs to remove the redundant candidates, rather than comparing every available candidate pair, which reduces complexity and redundancy. *Id*.

178. The ’833 Patent therefore provides a specific technological improvement to the functionality and capabilities of video coding technology that results in reduced complexity and improved prediction accuracy, which in turn reduces the information to be transmitted. *Id*. at 4:8-12, 8:13-16.

179. Conventional technology prior to the ’833 Patent was not capable of determining a subset of spatial motion vector prediction candidates based on a location of the block associated with a first spatial motion vector prediction candidate, nor was conventional technology capable of determining to exclude the first spatial motion vector prediction candidate from a merge list of motion vector prediction candidates based on comparing motion information of the first spatial motion vector prediction candidate with motion information of a limited number of other spatial motion vector prediction candidates without making a comparison of each pair from the set of spatial motion vector prediction candidates.

180. The ’833 Patent recognizes and solves these specific technological problems with the conventional technology at the time. The ’833 Patent’s ability to determine a subset of spatial motion vector prediction candidates based on a location of the block associated with a first spatial motion vector prediction candidate and ability to determine to exclude the first spatial motion vector prediction candidate from a merge list of motion vector prediction candidates based on comparing motion information of the first spatial motion vector prediction candidate with motion information of a limited number of other spatial motion vector prediction candidates without making a comparison of each pair from the set of spatial motion vector prediction candidates was a significant advancement over existing technology.

181. The novel solution of the ’833 Patent, including determining a subset of spatial motion vector prediction candidates based on a location of the block associated with a first spatial

motion vector prediction candidate and determining to exclude the first spatial motion vector prediction candidate from a merge list of motion vector prediction candidates based on comparing motion information of the first spatial motion vector prediction candidate with motion information of a limited number of other spatial motion vector prediction candidates without making a comparison of each pair from the set of spatial motion vector prediction candidates, was not well-understood, routine, or conventional, nor was it simply comprised of well-understood, routine, and conventional activities previously known to the industry. Furthermore, the ordered combination of elements, including determining a subset of spatial motion vector prediction candidates based on a location of the block associated with a first spatial motion vector prediction candidate and determining to exclude the first spatial motion vector prediction candidate from a merge list of motion vector prediction candidates based on comparing motion information of the first spatial motion vector prediction candidate with motion information of a limited number of other spatial motion vector prediction candidates without making a comparison of each pair from the set of spatial motion vector prediction candidates, was not well-understood, routine, or conventional.

N. U.S. Patent No. 11,805,267 ("the '267 Patent")

182. On October 31, 2023, the U.S. Patent and Trademark Office duly and legally issued U.S. Patent No. 11,805,267 ("the '267 Patent"), entitled "Motion Prediction in Video Coding," to inventors Kemal Ugur, Jani Lainema, and Antti Hallapuro. Nokia owns all rights to the '267 Patent necessary to bring this action. The '267 Patent issued from U.S. Application No. 17/328,750, filed on May 24, 2021, which is a continuation of U.S. Application No. 16/729,974, filed on December 30, 2019, which issued as U.S. Patent No. 11,019,354, which is a continuation of U.S. Application No. 15/876,495, filed on January 22, 2018, which issued as U.S. Patent No. 10,523,960, which is a continuation of U.S. Application No. 15/490,469, filed on April 18, 2017,

which issued as U.S. Patent No. 9,877,037, which is a continuation of U.S. Application No. 15/250,124, filed on August 29, 2016, which issued as U.S. Patent No. 9,628,816, which is a continuation of U.S. Patent Application No. 13/344,893, filed on January 6, 2012, which issued as U.S. Patent No. 9,432,693, and claims priority to U.S. Provisional Application No. 61/430,694, filed January 7, 2011. A true and correct copy of the ’267 Patent is attached hereto as Exhibit 17 and incorporated herein by reference.

183. The ’267 Patent is not directed to merely an abstract idea or any patent-ineligible concept. Instead, the ’267 Patent provides improvements over conventional video coding techniques that result in substantial benefits to video compression, video quality, and video playback. These substantial benefits are enjoyed by users of the Accused Products when, for example, watching video over the Internet.

184. Encoders compress video into a representation suitable for storage or transmission. ’267 Patent at 1:26-33. Decoders can decompress the compressed video representation into viewable form. *Id*. One compression technique used to reduce the size of an encoded bitstream is called “Motion Compensated Prediction (MCP).” *Id*. at 2:20-34. In MCP, a prediction for a current frame is formed using a previously coded frame or using multiple previously coded frames. *Id*. An example of a frame that is predicted using multiple previously coded frames is called a “B-picture.” B-pictures are bi-predicted pictures which use two other pictures as reference pictures, or two prediction blocks within one reference picture. *Id*. at 2:34-45.

185. As described in the ’267 Patent, in bi-prediction, the prediction signal of the block may be formed by averaging two motion compensated prediction blocks, followed by either up or down rounding, which may introduce rounding errors. *Id*. at 3:49-55.

186. Prior to the ’267 Patent, one significant problem was that the accumulation of rounding errors in bi-prediction degraded the coding efficiency. *Id*. at 3:56-65. Conventional technology attempted to remove or decrease this rounding error accumulation by signaling whether rounding up or rounding down was used or, alternatively, by alternating the usage of the rounding up and rounding down for each frame. *Id*. However, such prior methods increased the complexity of the process, as two separate code branches were required and the motion estimation routines in the encoder had to be doubled for both cases of rounding and truncating. *Id*. at 4:21-26.

187. The ’267 Patent overcame these technical challenges in the prior systems by inventing a method of maintaining the prediction signals at a higher precision during the prediction calculation and then reducing the precision after the two or more prediction signals have been combined with each other. *Id*. at 4:29-35. The ’267 Patent employs the unconventional solution of maintaining a higher accuracy until the prediction signals have been combined to obtain the bi-prediction or multi-prediction signal, which eliminates the need for including a rounding direction indicator in the bitstream or the added complexity of alternating the rounding directions between frames. *Id*. at 4:36-43, 6:51-57. With the invention of the ’267 Patent, the encoder can transmit residual data based on the difference between the combined prediction and the block of pixels, and the decoder can reconstruct the block of pixels based on the combined prediction. *Id*. at 14:51-59, 15:14-24, 16:14-24.

188. The ’267 Patent therefore provides a specific technological improvement to the functionality and capabilities of video coding technology that results in increased efficiency and significant reduction in the information to be transmitted and received. *Id*. at 7:31-38.

189. Conventional technology prior to the ’267 Patent was not capable of reducing the accumulation of rounding errors in bi-prediction or multi-prediction without signaling the rounding offset or using different methods for rounding for different frames. *Id*. at 6:51-57.

190. The ’267 Patent recognizes and solves these specific technological problems with the conventional technology at the time. The ’267 Patent’s ability to obtain a first prediction and a second prediction, each having a precision which is higher than the precision of the reference pixel values, and after combining the first prediction and the second prediction, decreasing the precision of said combined prediction by shifting bits of the combined prediction to the right such that the residual data in the bitstream is based on the difference between the combined prediction and the block of pixels, and such that the combined prediction is used by the decoder to reconstruct the block of pixels, was a significant advancement over existing technology.

191. The novel solution of the ’267 Patent, including obtaining a first prediction and a second prediction, each having a precision which is higher than the precision of the reference pixel values, and after adding the first prediction and the second prediction with a rounding value, decreasing the precision of said combined prediction by shifting bits of the combined prediction to the right such that the residual data in the bitstream is based on the difference between the combined prediction and the block of pixels, and such that the combined prediction is used by the decoder to reconstruct the block of pixels, was not well-understood, routine, or conventional, nor was it simply comprised of well-understood, routine, and conventional activities previously known to the industry. Furthermore, the ordered combination of elements, including obtaining a first prediction and a second prediction, each having a precision which is higher than the precision of the reference pixel values, and after adding the first prediction and the second prediction with a rounding value, decreasing the precision of said combined prediction by shifting bits of the

combined prediction to the right such that the residual data in the bitstream is based on the difference between the combined prediction and the block of pixels, and such that the combined prediction is used by the decoder to reconstruct the block of pixels, was not well-understood, routine, or conventional.

O. U.S. Patent No. 9,390,137 ("the '137 Patent")

192. On July 12, 2016, the U.S. Patent and Trademark Office duly and legally issued U.S. Patent No. 9,390,137 ("the '137 Patent"), entitled "Method and Apparatus for Providing an Ordering Metric for a Multi-Dimensional Contextual Query," to inventors Vidya Setlur and Agathe Battestini. Nokia owns all rights to the '137 Patent necessary to bring this action. The '137 Patent issued from U.S. Application No. 13/172,425, filed on June 29, 2011. A true and correct copy of the '137 Patent is attached hereto as Exhibit 18 and incorporated herein by reference.

193. The '137 Patent is not directed to merely an abstract idea or any patent-ineligible concept. Instead, the '137 Patent provides improvements over conventional content delivery and network services techniques that result in substantial benefits to the retrieval and presentation of contextual attributes of objects stored in databases. These substantial benefits are enjoyed by users of the Accused Products when, for example, accessing, searching for, and watching video content over the Internet, including users of mobile devices.

194. In order to deliver compelling network services and relevant content to users, network and content services often rely on the context of the users accessing the network services or the devices by which the users access the networks services. '137 Patent at 1:8-14. An example of a context is the current location of a user or device. *Id*. at 8:2-6.

195. Prior to the '137 Patent, one significant problem was that determining the context becomes inadequate when multiple contexts and/or multiple users' actions need to be analyzed

over a period of time— particularly when the context is determined in conjunction with a mobile device that has limited processing power and/or bandwidth. *Id*. at 1:14-19. Conventional technology faced significant technical challenges to providing rich, situational-aware, context-sensitive datasets in a mobile environment. *Id*. at 1:23-26. More specifically, prior to the ’137 Patent, conventional technology could not incorporate correlations between specific contexts and user actions into context-aware databases that could be useful for multiple contexts and/or multiple users and provide access to such datasets to mobile devices in a practical manner. *Id*. at 3:61-67, 1:19-23.

196. The ’137 Patent overcame these technical challenges in the prior systems by inventing a method to introduce the capability to enrich and enhance situational-aware, context-sensitive databases using mechanisms on, for example, a mobile device. *Id*. at 4:1-4. The ’137 Patent employs the unconventional solution of using a multi-dimensional query associated with at least one user device, wherein the multi-dimensional query specifies, at least in part, one or more personas based at least in part on more than one person, and executing the multi-dimensional query on at least one context-sensitive database to generate one or more results that are ordered based, at least in part, on one or more user contextual attributes. *Id*. at 1:34-45.

197. The ’137 Patent therefore provides a specific technological improvement to the functionality and capabilities of context-aware querying, retrieval and presentation of contextual attributes of objects stored in databases. *Id*. at 4:7-11. By associating the users and/or the device with personas and related user contextual attributes, the ’137 Patent is able to determine and handle a wider range of context information based on situational awareness. *Id*. at 11:16-34.

198. Conventional technology prior to the ’137 Patent was not capable of performing a multi-dimensional query associated with at least one user device, which specifies one or more

personas based on more than one person, and executing the multi-dimensional query on at least one context-sensitive database to generate one or more results that are ordered based on one or more user contextual attributes. *Id*. at 3:61-67, 1:19-23.

199. The ’137 Patent recognizes and solves these specific technological problems with the conventional technology at the time. The ’137 Patent’s ability to determine a multi-dimensional query associated with at least one user device, wherein the multi-dimensional query specifies, at least in part, one or more personas, based, at least in part, on more than one person, associated with the at least one user device and execute the multi-dimensional query on a context-sensitive database to generate one or more results ordered based, at least in part, on one or more user contextual attributes of the at least one user device, was a significant advancement over existing technology.

200. The novel solution of the ’137 Patent, including determining a multi-dimensional query associated with at least one user device, wherein the multi-dimensional query specifies, at least in part, one or more personas, based, at least in part, on more than one person, associated with the at least one user device and executing the multi-dimensional query on a context-sensitive database to generate one or more results ordered based, at least in part, on one or more user contextual attributes of the at least one user device, was not well-understood, routine, or conventional, nor was it simply comprised of well-understood, routine, and conventional activities previously known to the industry. Furthermore, the ordered combination of elements, including determining a multi-dimensional query associated with at least one user device, wherein the multi-dimensional query specifies, at least in part, one or more personas, based, at least in part, on more than one person, associated with the at least one user device and executing the multi-dimensional query on a context-sensitive database to generate one or more results ordered

based, at least in part, on one or more user contextual attributes of the at least one user device, was not well-understood, routine, or conventional.

201. The ’808, ’321, ’818, ’469, ’599, ’273, ’701, ’891, ’005, ’764, ’148, ’991, ’833, ’267, and ’137 Patents are collectively referred to as the “Asserted Patents.”

202. Nokia exclusively owns all rights, title, and interest in the Asserted Patents necessary to bring this action, including the right to recover past and future damages. The Asserted Patents were previously owned by Nokia’s predecessor-in-interest, Nokia Corporation, and were transferred to Nokia Technologies Oy in 2015. Nokia or its predecessor-in-interest has owned all rights to the Asserted Patents necessary to bring this action throughout the period of Amazon’s infringement and still owns those rights to the Asserted Patents. Amazon is not currently licensed to practice the Asserted Patents.

203. The Asserted Patents are valid and enforceable.

204. Amazon has made, used, marketed, offered for sale, sold, and/or imported into the United States products and services, such as video content on Amazon.com, Amazon Prime Video and Twitch.tv, in a manner that infringes the Asserted Patents by, for example, implementing or being capable of implementing the video encoding and decoding features and techniques claimed in the Asserted Patents.

205. Amazon has been placed on actual notice of infringement by Nokia prior to the filing of this Complaint as to certain of the Asserted Patents. At a minimum, in accordance with 35 U.S.C. § 287, Amazon has had actual notice and knowledge of Nokia’s charge of infringement as to all the Asserted Patents at least as early as the filing of this Original Complaint and/or the date this Original Complaint was served upon Amazon. Despite such notice, Amazon continues

to make, use, import into, market, offer for sale, and/or sell in the United States products and services that infringe the Asserted Patents.

GENERAL ALLEGATIONS

206. Amazon has infringed and continues to infringe each of the Asserted Patents by engaging in acts constituting infringement under 35 U.S.C. § 271, including but not necessarily limited to one or more of making, using, selling, and offering to sell, in this District and elsewhere in the United States, and importing into the United States, the Accused Products.

207. Amazon's acts of infringement have caused damage to Nokia. Nokia is entitled to recover from Amazon the damages sustained by Nokia as a result of Amazon's wrongful acts in an amount subject to proof at trial.

208. Amazon's infringement of the Asserted Patents has been and continues to be willful. Amazon has committed and continues to commit acts of infringement despite a high likelihood that its actions constitute infringement, and Amazon knew or should have known that its actions constituted an unjustifiably high risk of infringement.

209. Nokia has complied with any applicable marking requirements under 35 U.S.C. § 287(a) at least because the asserted method claims do not require marking and/or there is nothing to mark.

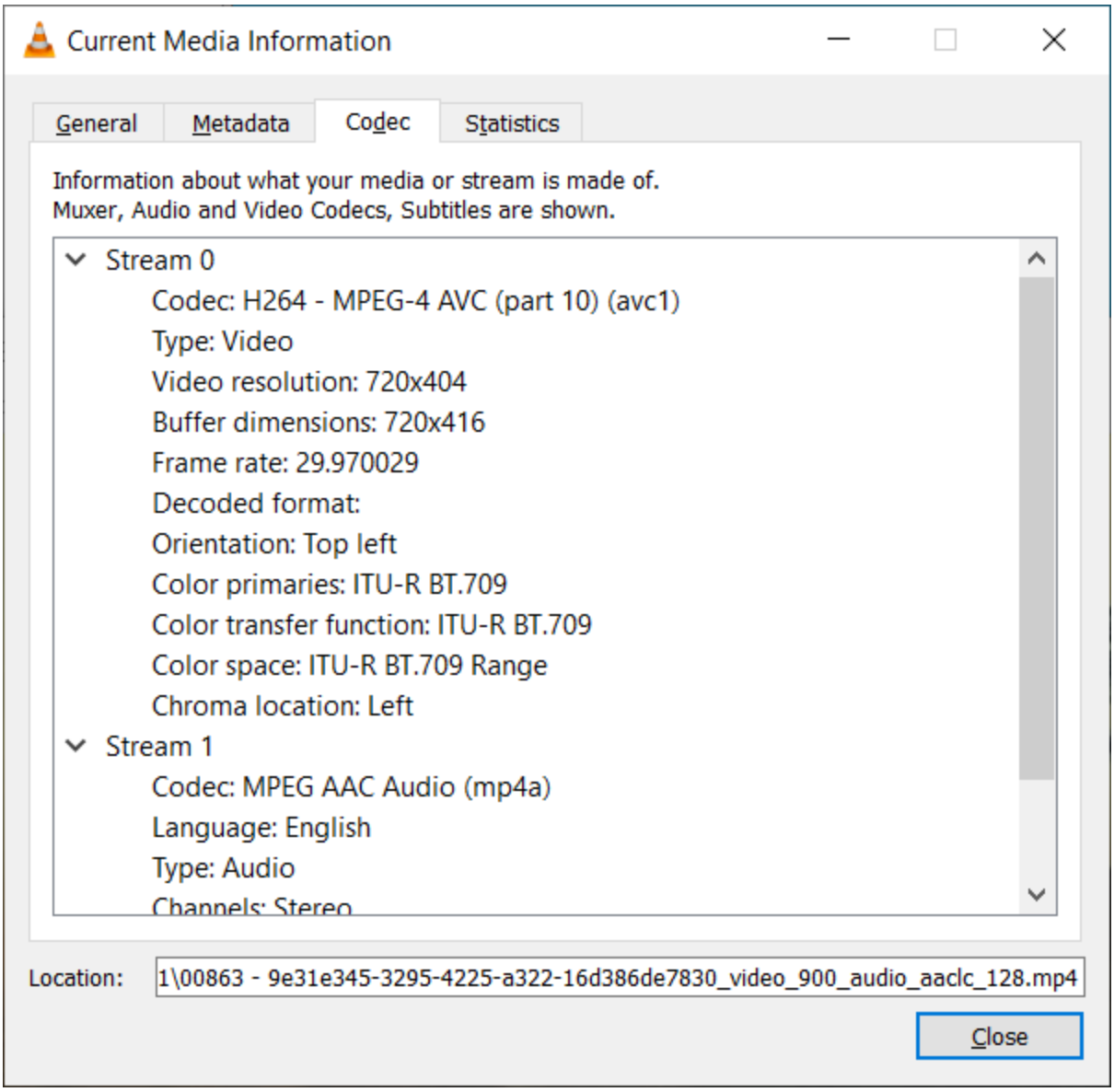
210. Nokia has identified below at least one exemplary claim per patent to demonstrate infringement by one exemplary product. However, the selection of claims should not be considered limiting, and additional claims of the Asserted Patents that are infringed by Amazon will be disclosed in compliance with the Court's rules related to infringement contentions and discovery.

AMAZON'S ACCUSED PRODUCTS

A. Amazon Makes, Imports, Uses, Sells, and/or Offers for Sale Products and Services that Infringe the '808 Patent.

211. The Accused Products infringe one or more claims of the '808 Patent, including, for example, claim 1.

212. As just one example of infringement, on information and belief, Amazon performs a method of encoding a video sequence in a manner that is covered by claim 1 of the '808 Patent for Amazon Prime Video content (such as trailers), as demonstrated in the screenshots below using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

213. When encoding Amazon Prime Video trailers, for example, using Context-based Adaptive Binary Arithmetic Coding (CABAC), on information and belief, Amazon performs assigning a skip coding mode to a first segment of a first frame of the sequence, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info

Filter | Hex

SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	0
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	3
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	-3
deblocking_filter_control_present_flag	1
constrained_intra_pred_flag	0
redundant_pic_cnt_present_flag	0
rbsp_stop_one_bit	1
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

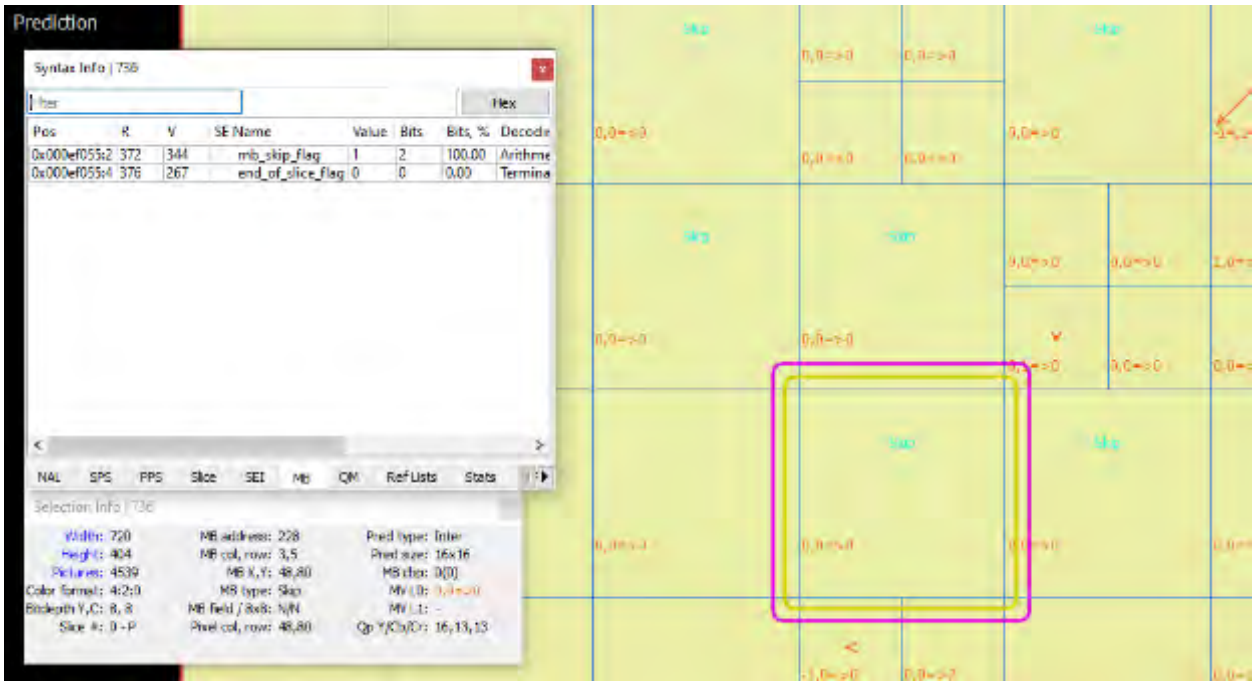
Slice #0 size in bits: 74416 SLICE_NONIDR

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	5
pic_parameter_set_id	0
frame_num	1
pic_order_cnt_lsb	90
num_ref_idx_active_override_flag	0
> **ref_pic_list_reordering()**	
> **pred_weight_table()**	
> **dec_ref_pic_marking()**	
cabac_init_idc	0
slice_qp_delta	-10
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	-1
slice_beta_offset_div2	-1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

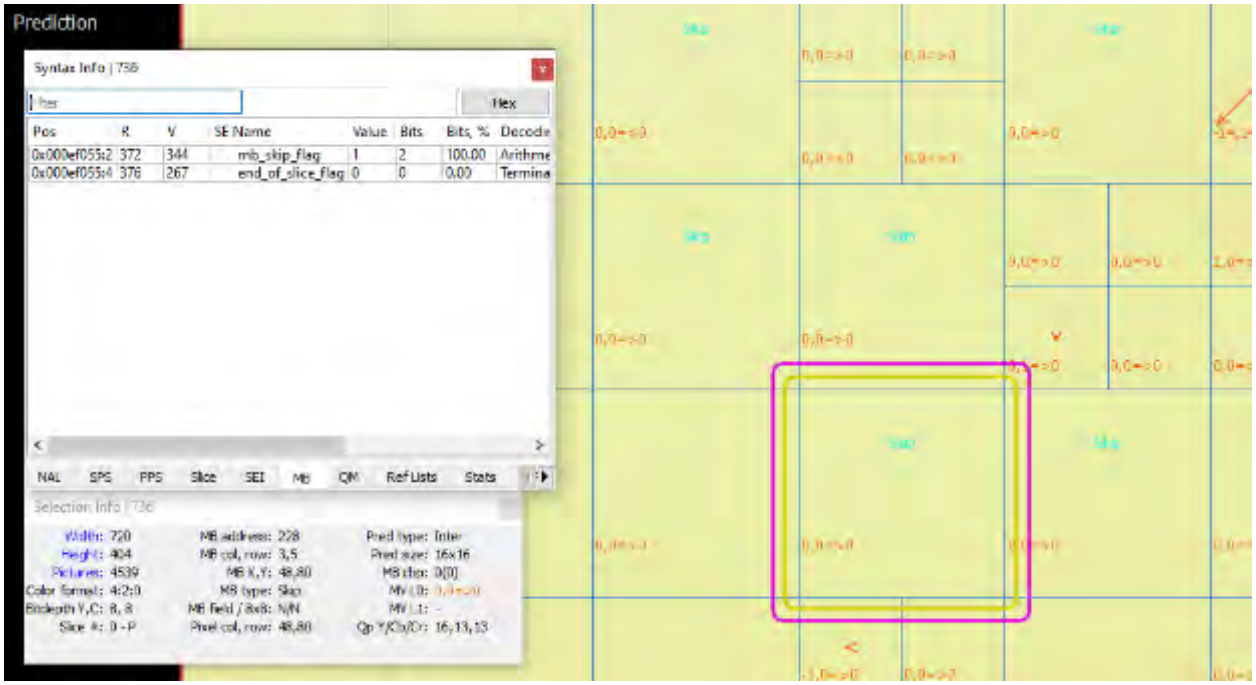
NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

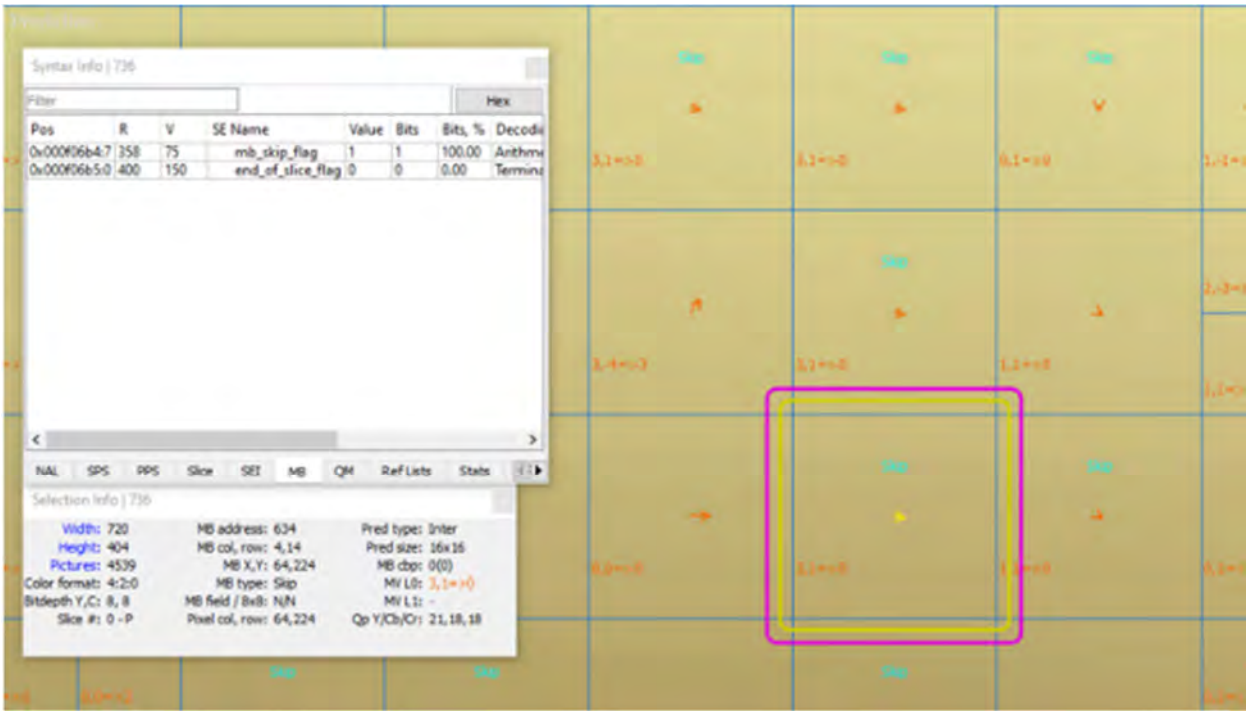


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

214. When encoding Amazon Prime Video trailers, for example, using Context-based Adaptive Binary Arithmetic Coding (CABAC), on information and belief, Amazon performs assigning either a zero motion vector or a predicted non-zero motion vector for the skip coding mode for the first segment based at least in part on the motion information of a second segment neighboring the first segment, as demonstrated in the screenshots below using VQ Analyzer software.



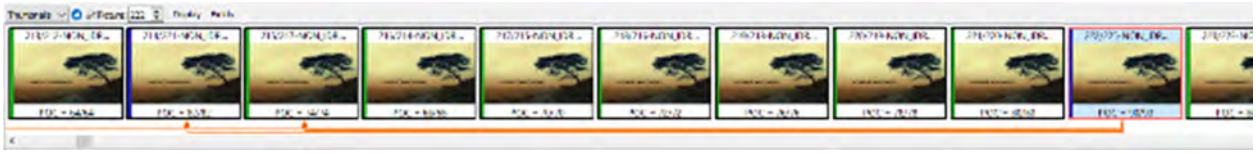
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



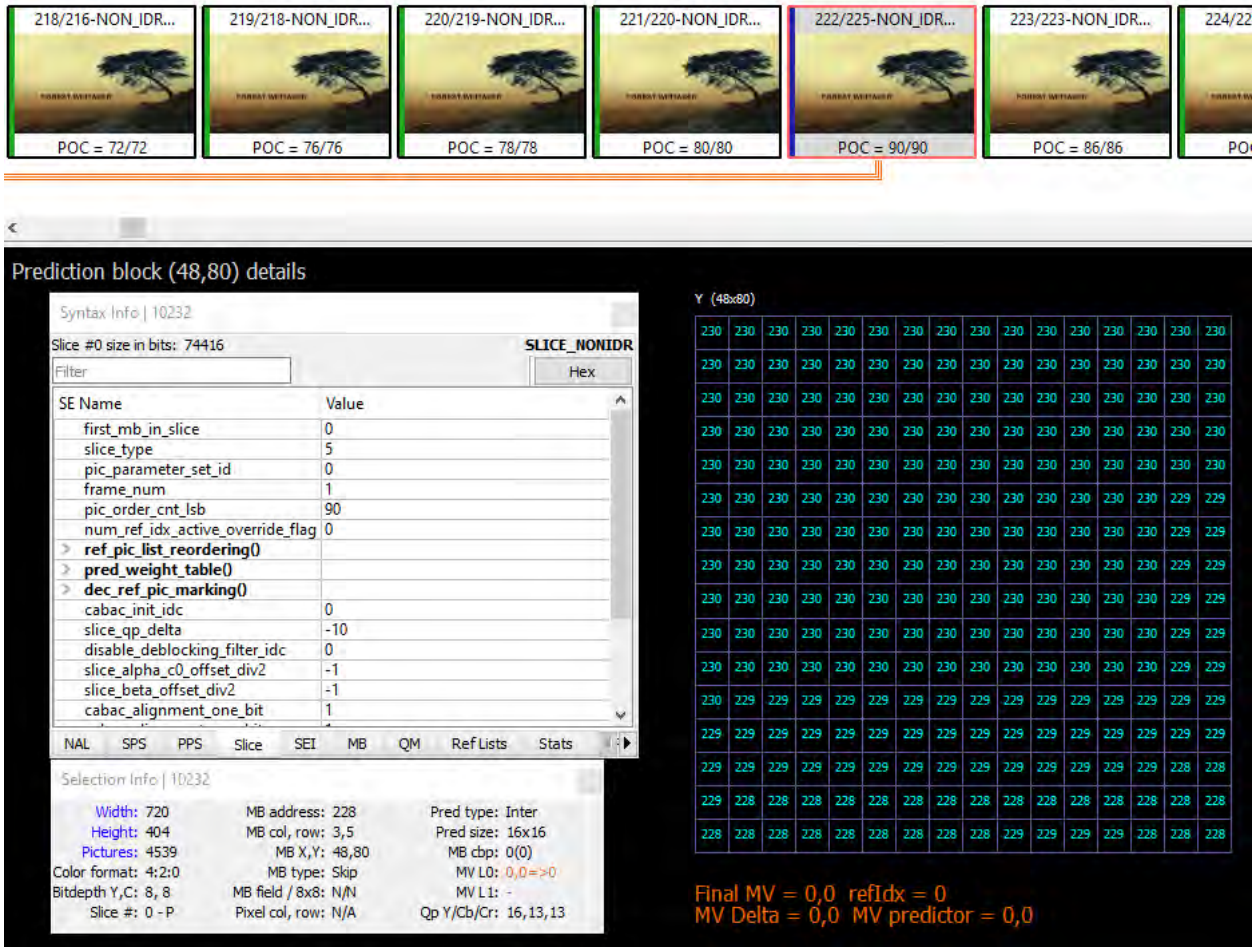
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

215. When encoding Amazon Prime Video trailers, for example, using Context-based Adaptive Binary Arithmetic Coding (CABAC), on information and belief, Amazon performs

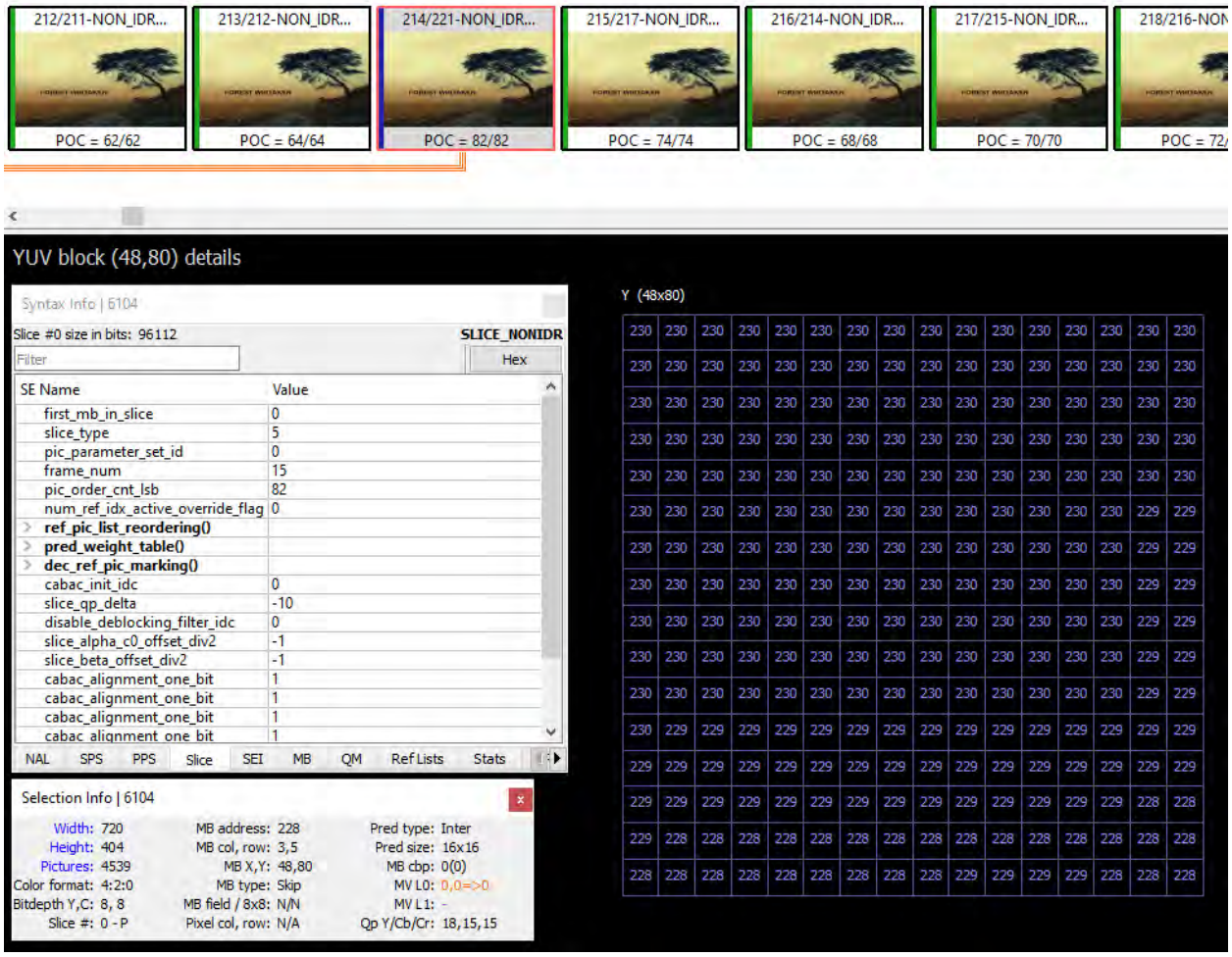
forming a prediction for the first segment with respect to a reference frame based at least in part on the assigned motion vector for the skip coding mode, wherein the assigned motion vector is one of the zero motion vector and the predicted non-zero motion vector, as demonstrated in the screenshots below using VQ Analyzer software.



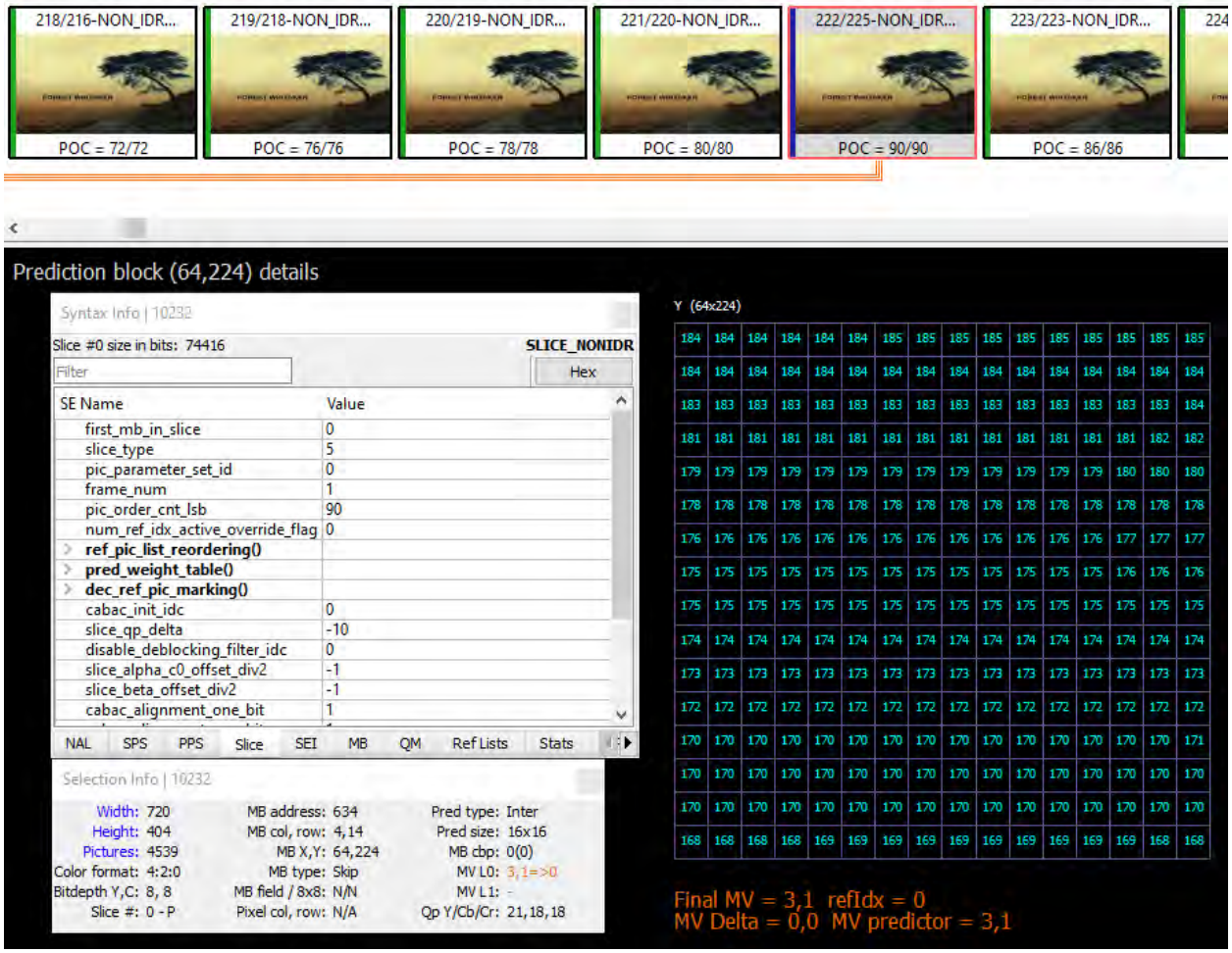
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



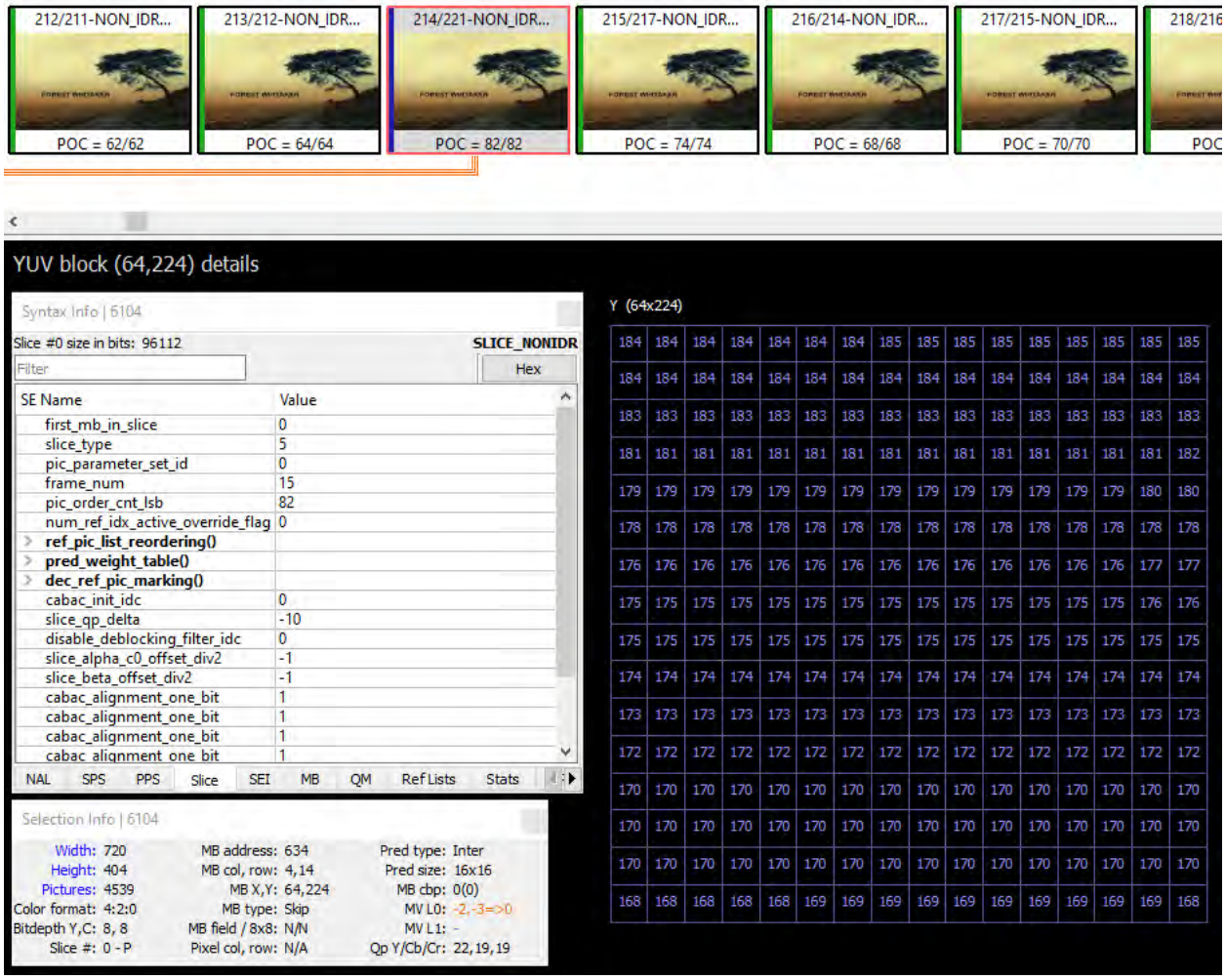
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

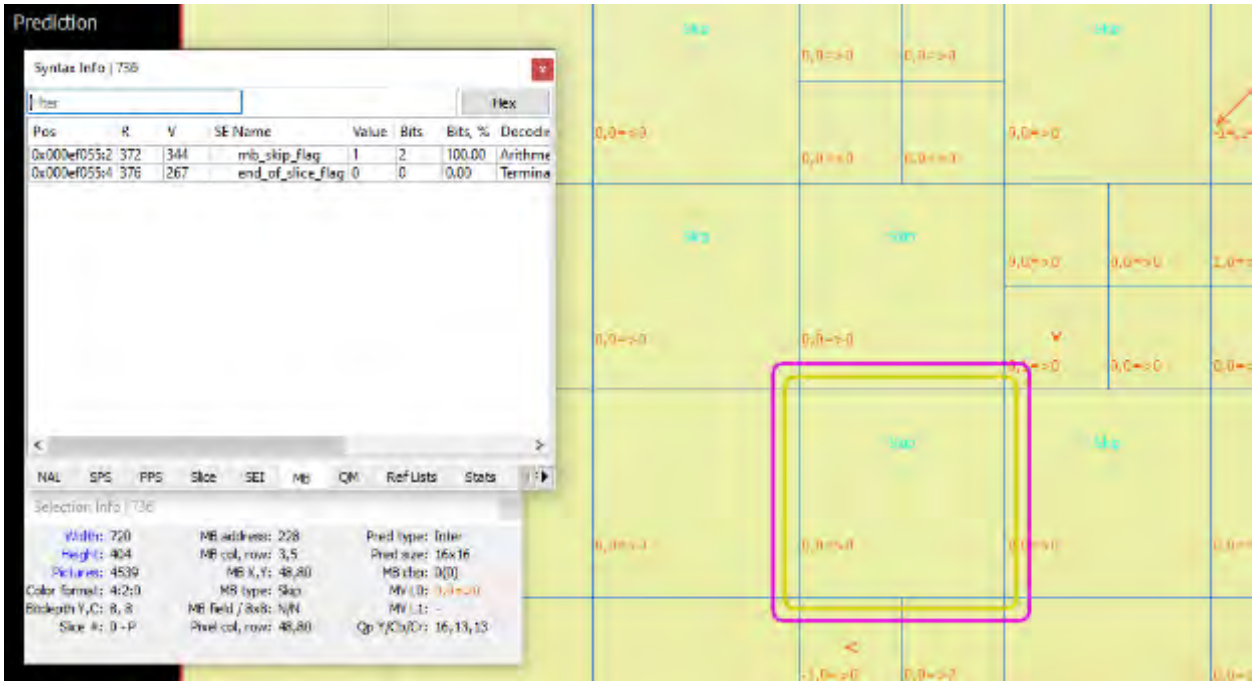


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

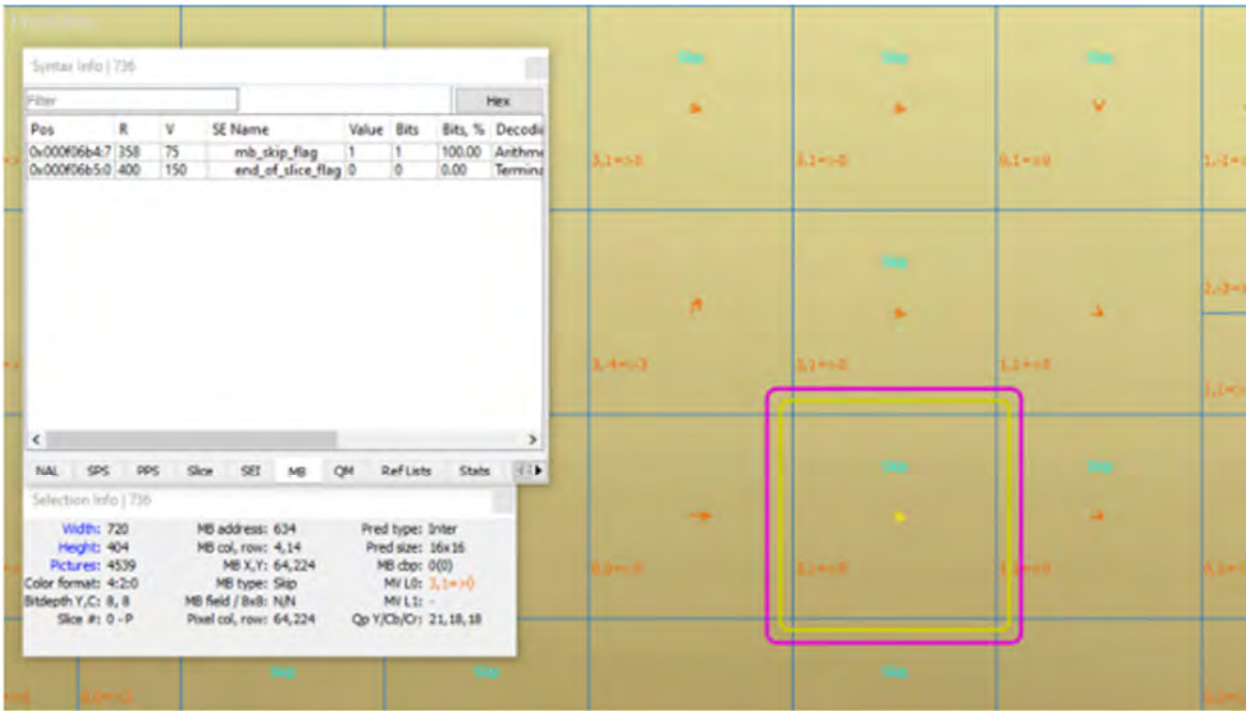


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

216. When encoding Amazon Prime Video trailers, for example, using Context-based Adaptive Binary Arithmetic Coding (CABAC), on information and belief, Amazon performs providing in an encoded bitstream an indication of the skip coding mode, wherein no further motion vector information for the first segment is coded in the encoded bitstream, as demonstrated in the screenshots below using VQ Analyzer software.



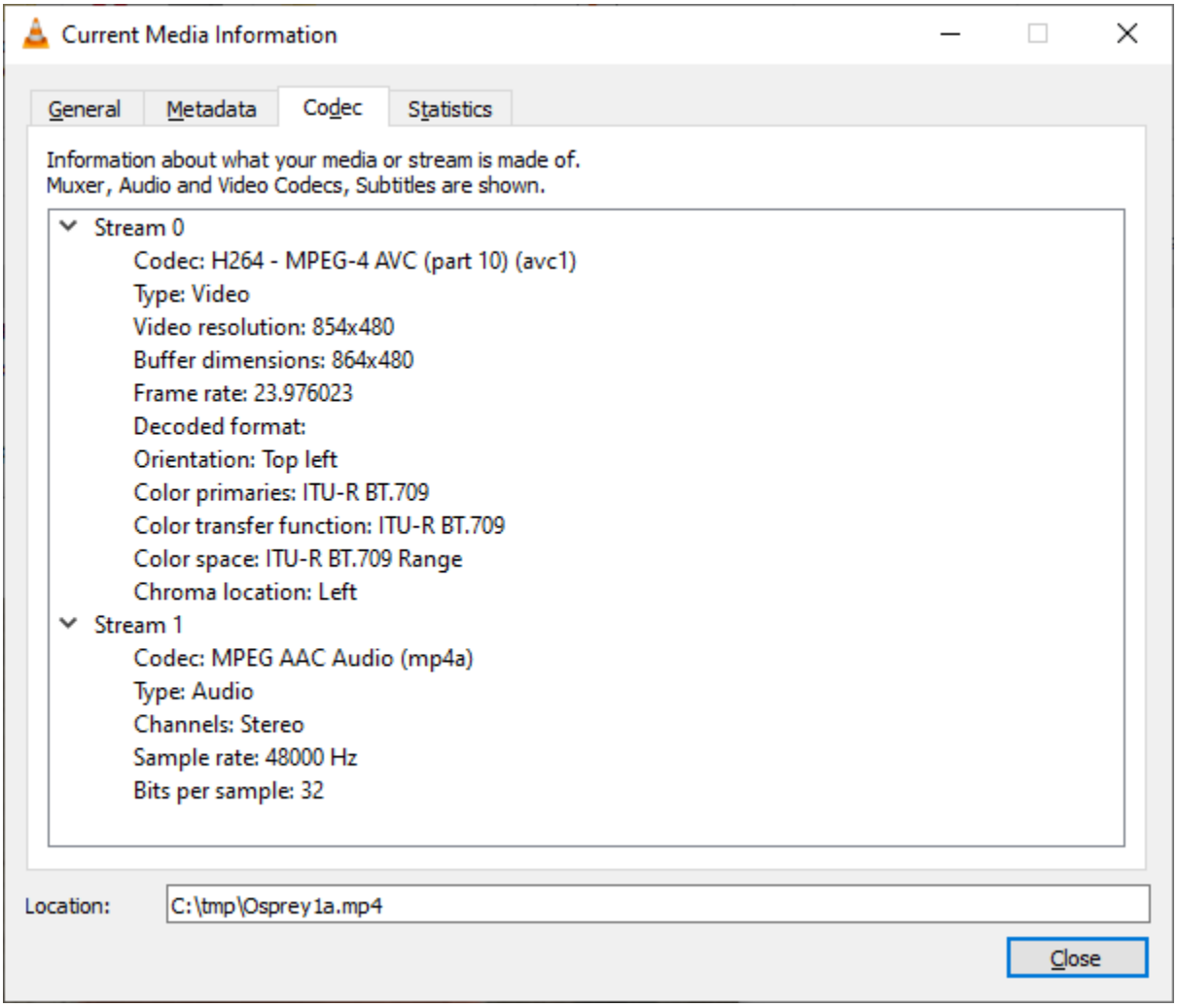
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

217. For another example, on information and belief, Amazon performs a method of encoding a video sequence in a manner that is covered by claim 1 of the ’808 Patent for

Amazon.com advertisements, as demonstrated in the screenshots below using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

218. When encoding Amazon.com advertisements, for example, using Context-based Adaptive Binary Arithmetic Coding (CABAC), on information and belief, Amazon performs assigning a skip coding mode to a first segment of a first frame of the sequence, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info | 6332

Filter | Hex

SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	1
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	2
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	0
deblocking_filter_control_present_flag	1
constrained_intra_pred_flag	0
redundant_pic_cnt_present_flag	0
transform_8x8_mode_flag	1
pic_scaling_matrix_present_flag	0
second_chroma_qp_index_offset	0
rbsp_stop_one_bit	1
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info

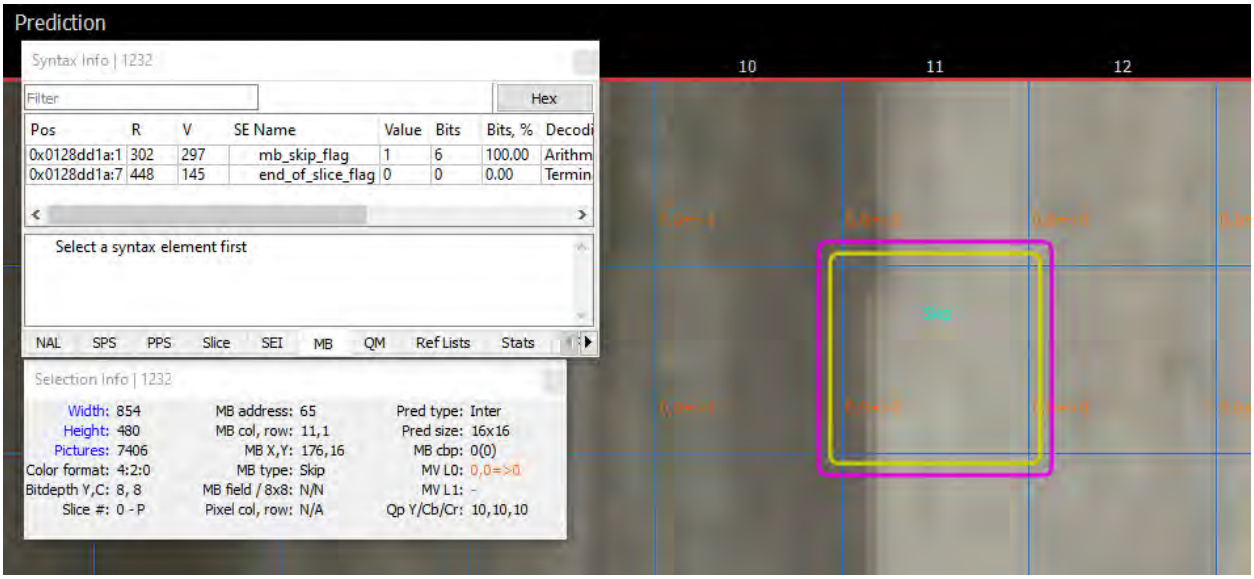
Slice #0 size in bits: 327256 SLICE_NONIDR

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	5
pic_parameter_set_id	0
frame_num	3
pic_order_cnt_lsb	16
delta_pic_order_cnt_bottom	0
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	1
ref_pic_list_reordering()	
pred_weight_table()	
dec_ref_pic_marking()	
cabac_init_idc	0
slice_qp_delta	-16
disable_deblocking_filter_idc	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

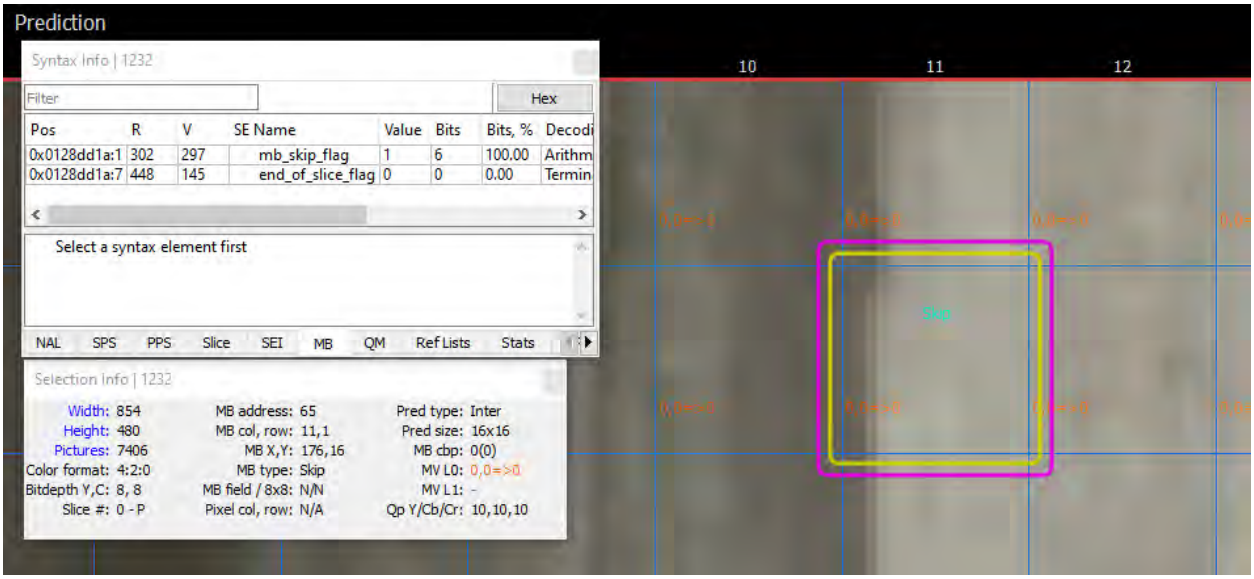
NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

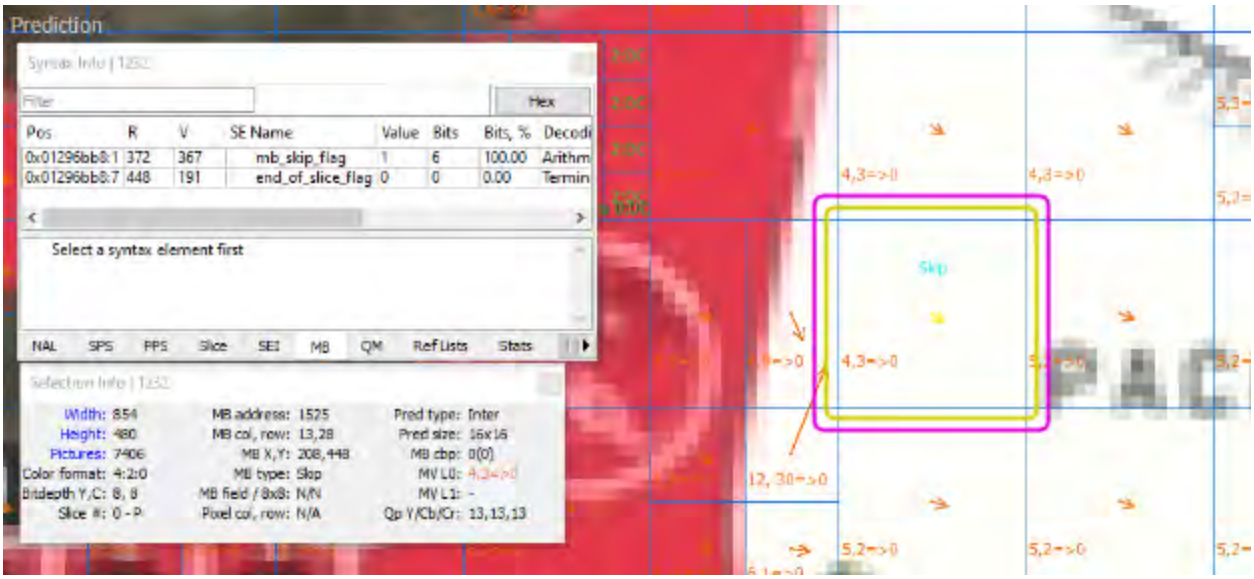


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

219. When encoding Amazon.com advertisements, for example, using Context-based Adaptive Binary Arithmetic Coding (CABAC), on information and belief, Amazon performs assigning either a zero motion vector or a predicted non-zero motion vector for the skip coding mode for the first segment based at least in part on the motion information of a second segment neighboring the first segment, as demonstrated in the screenshots below using VQ Analyzer software.



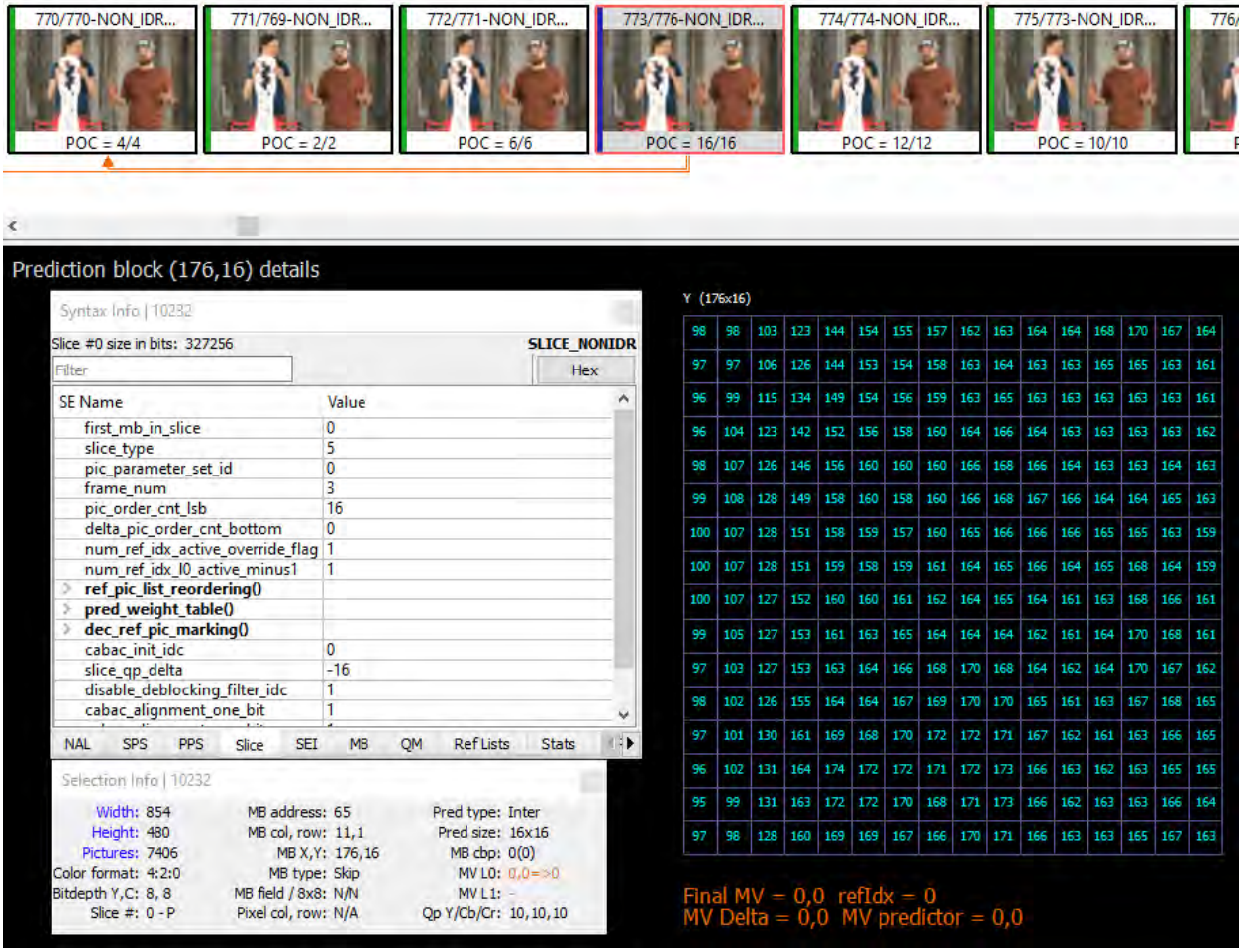
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



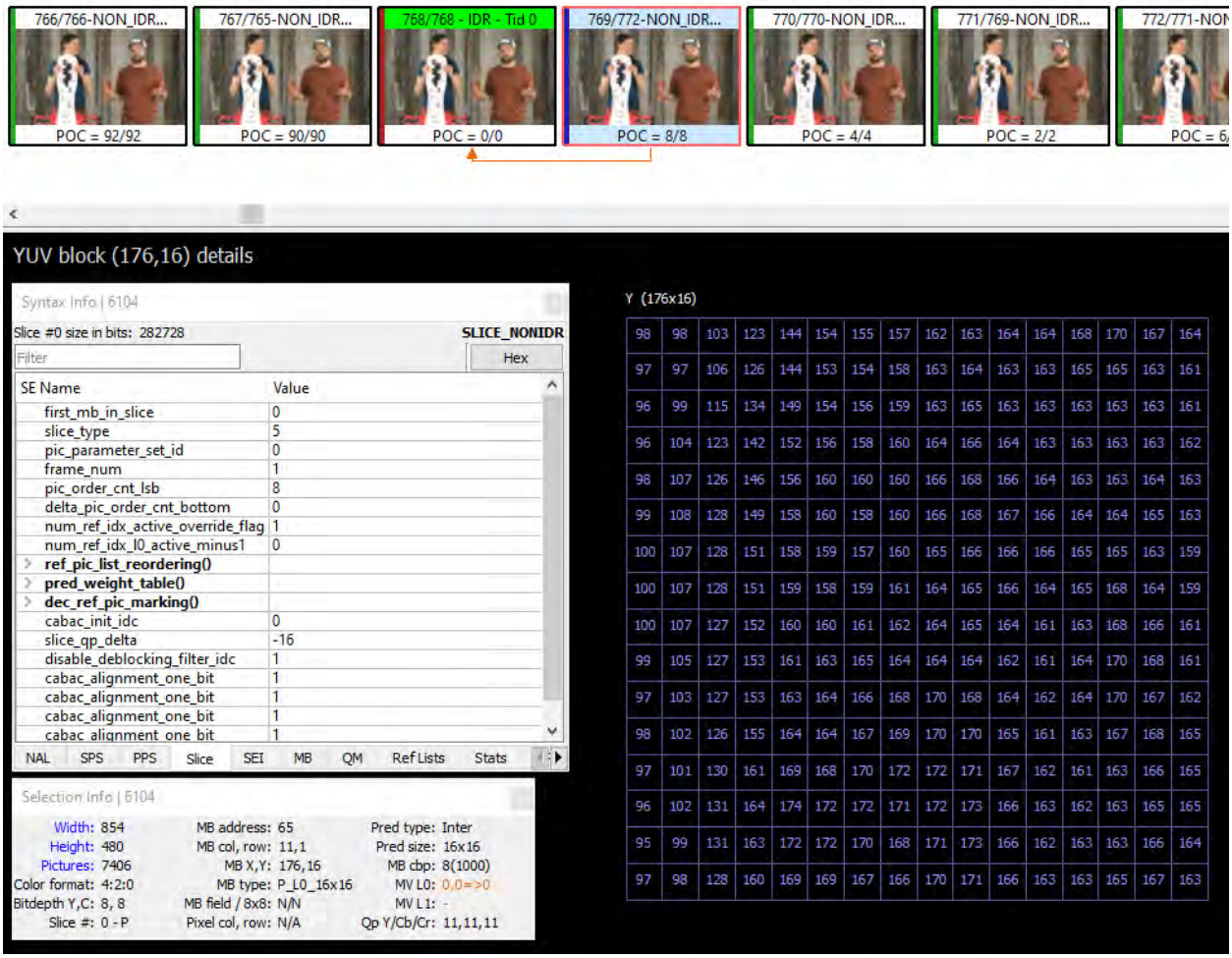
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

220. When encoding Amazon.com advertisements, for example, using Context-based Adaptive Binary Arithmetic Coding (CABAC), on information and belief, Amazon performs forming a prediction for the first segment with respect to a reference frame based at least in part on the assigned motion vector for the skip coding mode, wherein the assigned motion vector is one of the zero motion vector and the predicted non-zero motion vector, as demonstrated in the screenshots below using VQ Analyzer software.

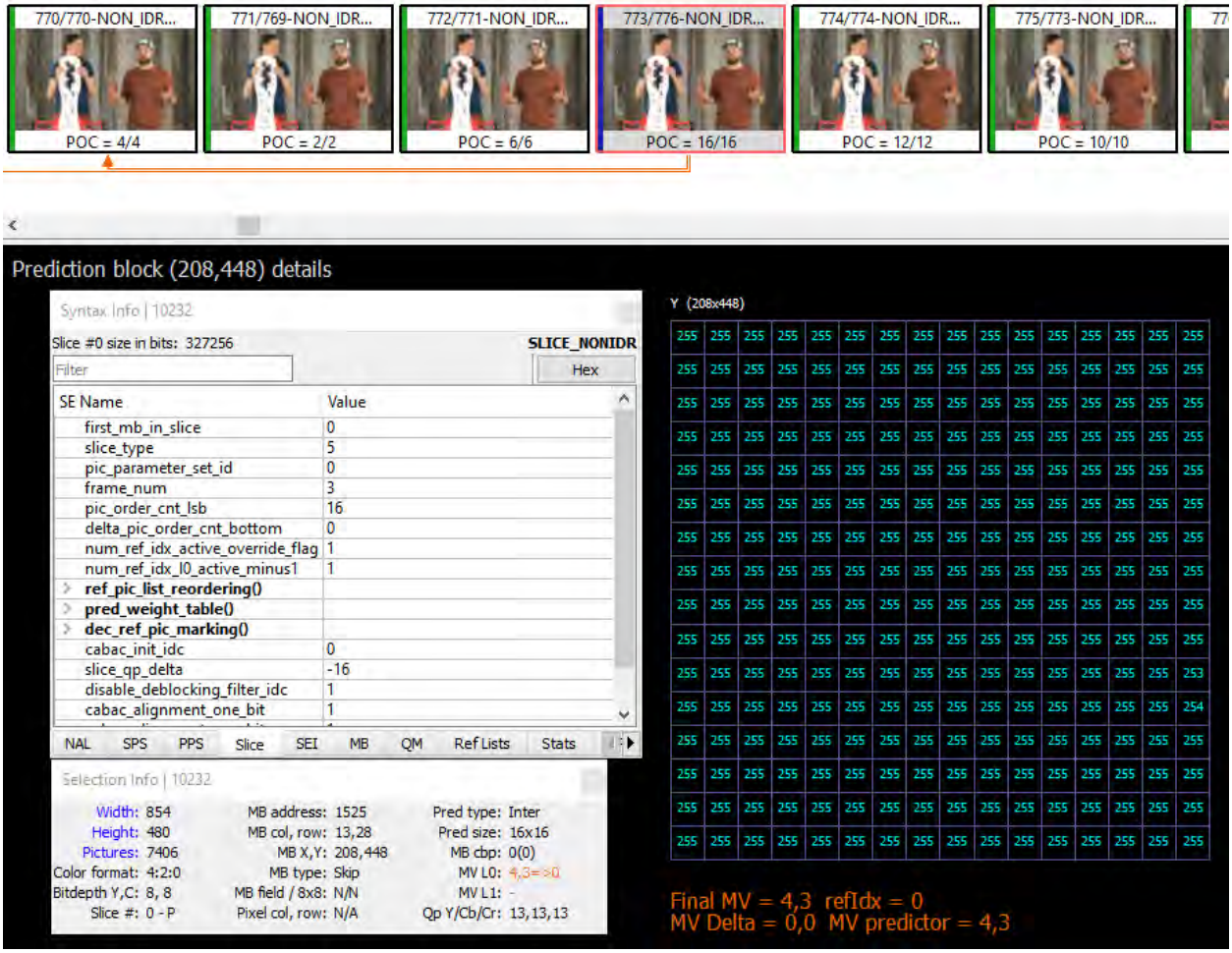
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



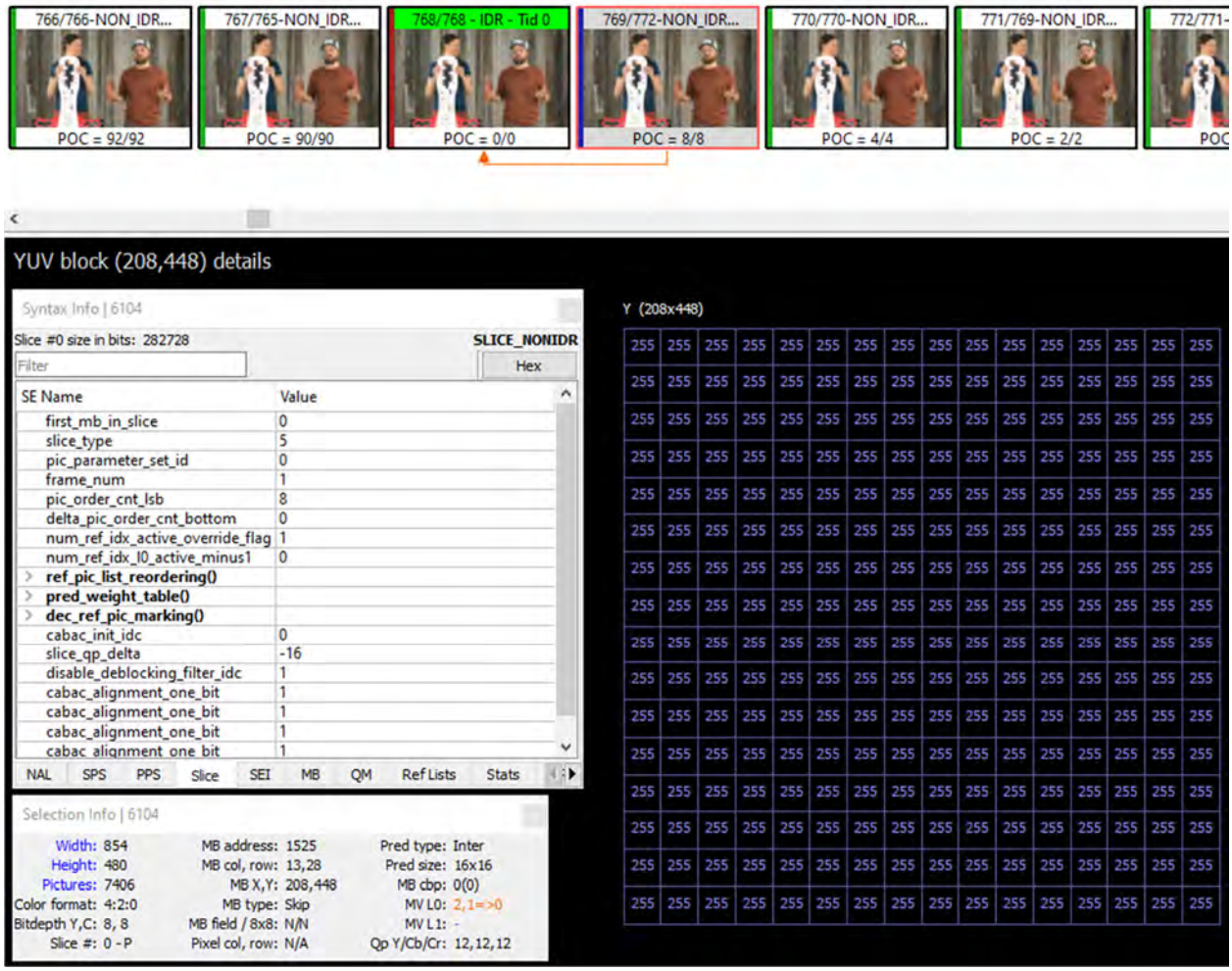
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

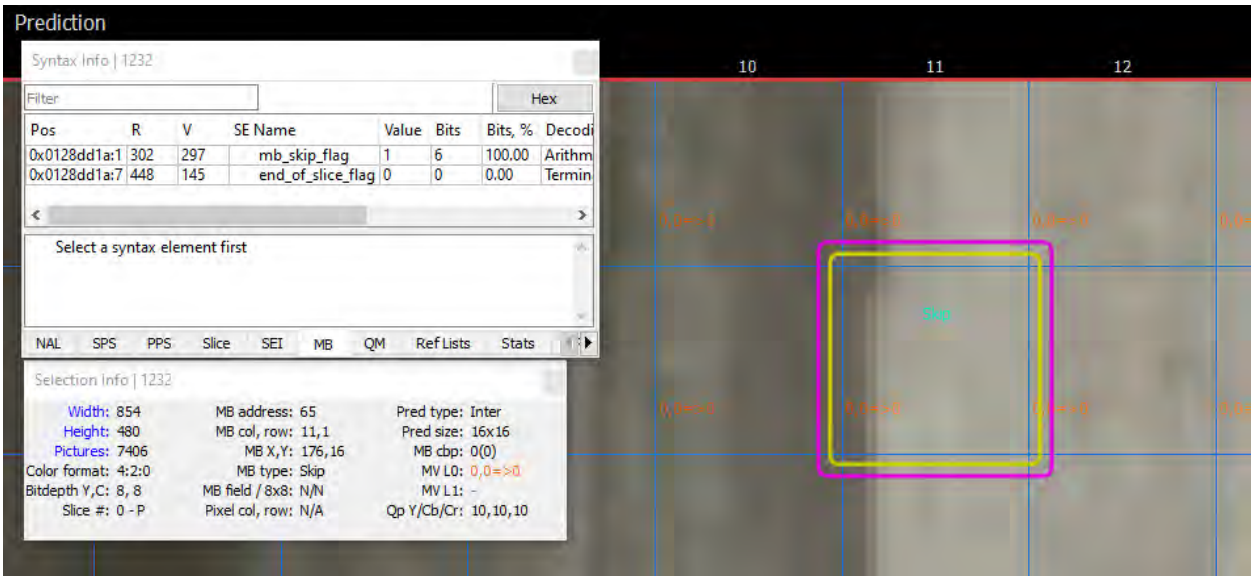


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

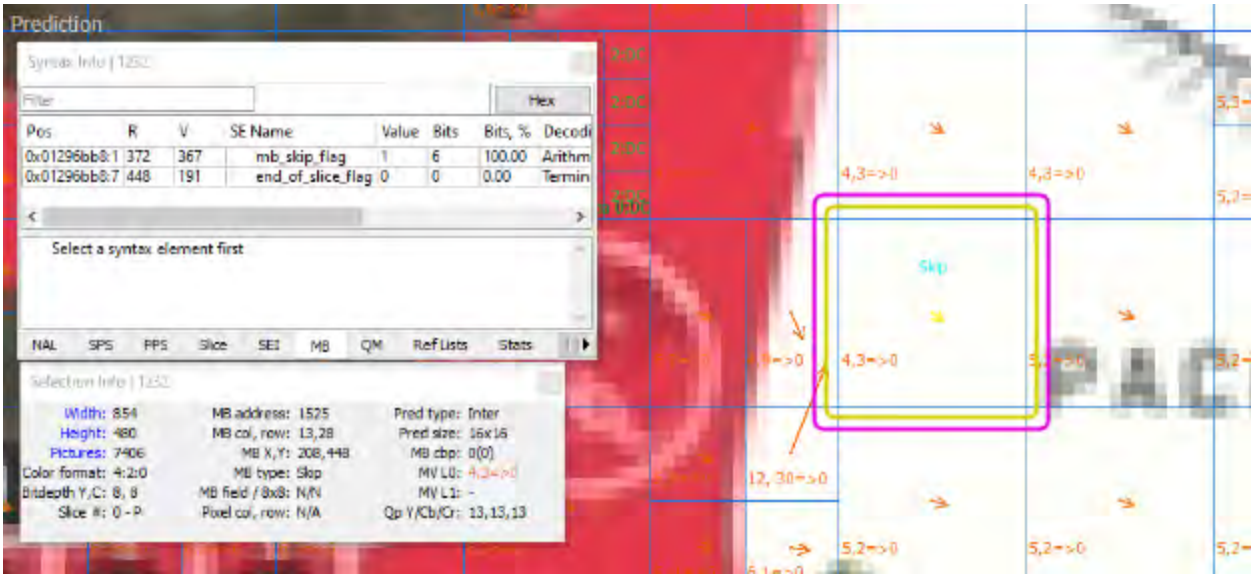


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

221. When encoding Amazon.com advertisements, for example, using Context-based Adaptive Binary Arithmetic Coding (CABAC), on information and belief, Amazon performs providing in an encoded bitstream an indication of the skip coding mode, wherein no further motion vector information for the first segment is coded in the encoded bitstream, as demonstrated in the screenshots below using VQ Analyzer software.

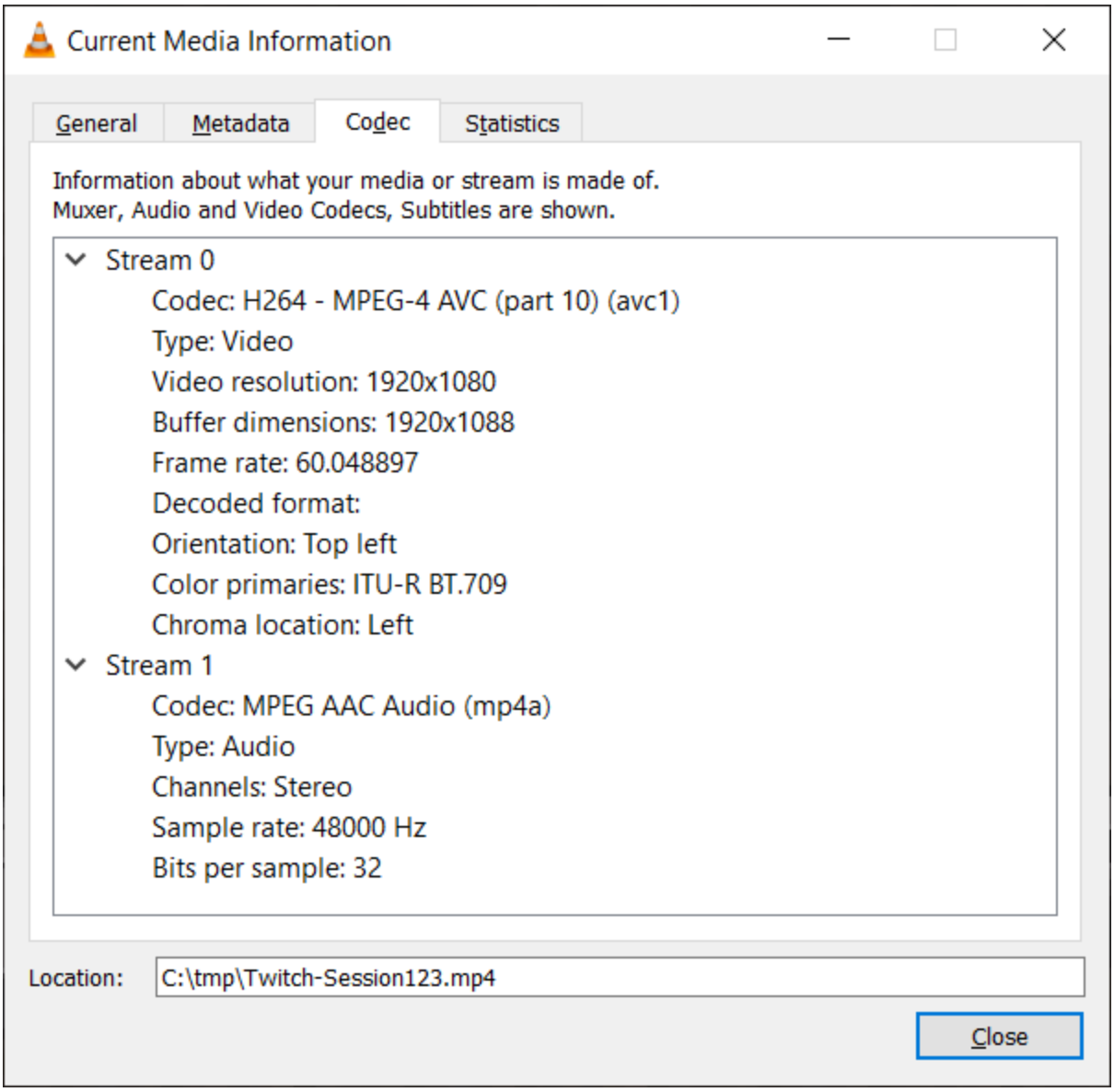


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

222. For another example, on information and belief, Amazon performs a method of encoding a video sequence in a manner that is covered by claim 1 of the ’808 Patent for Twitch.tv video, as demonstrated in the screenshots below using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

223. When encoding Twitch.tv video, for example, using Context-based Adaptive Binary Arithmetic Coding (CABAC), on information and belief, Amazon performs assigning a skip coding mode to a first segment of a first frame of the sequence, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info

Filter | Hex

SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	0
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	4
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	-3
deblocking_filter_control_present_flag	1
constrained_intra_pred_flag	0
redundant_pic_cnt_present_flag	0
transform_8x8_mode_flag	1
pic_scaling_matrix_present_flag	0
second_chroma_qp_index_offset	-3
rbsp_stop_one_bit	1

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info

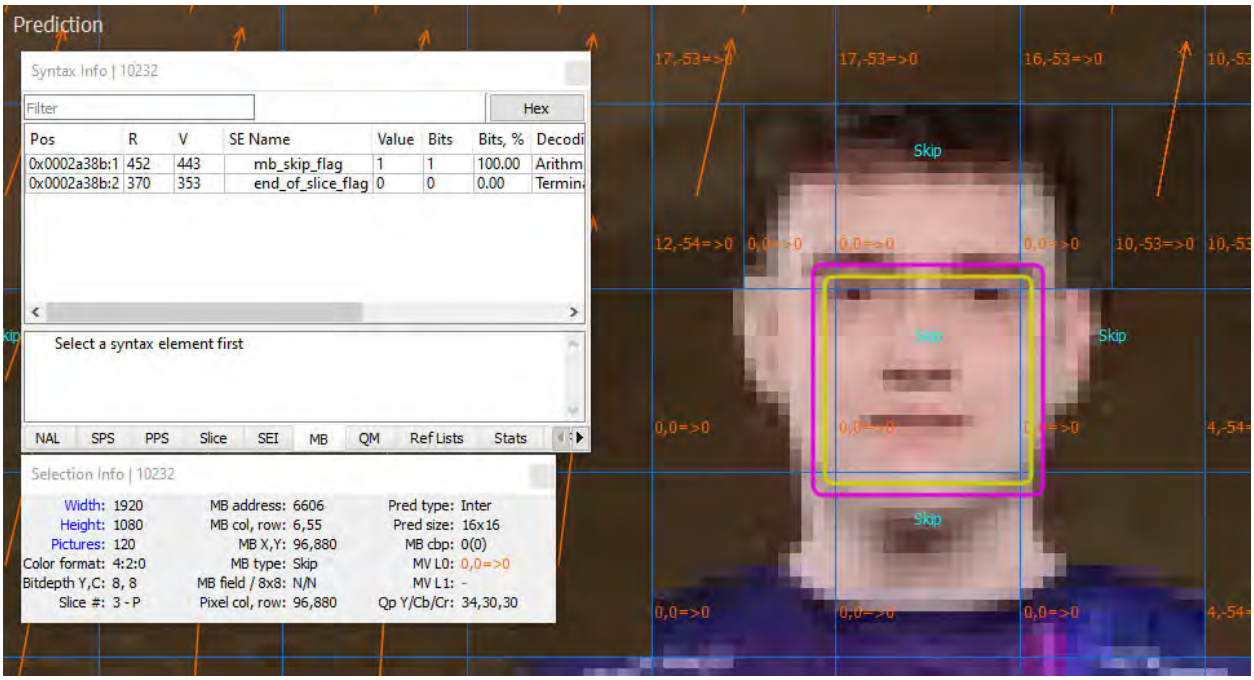
Slice #3 size in bits: 31232 SLICE_NONIDR

Filter Hex

SE Name	Value
first_mb_in_slice	6120
slice_type	5
pic_parameter_set_id	0
frame_num	1
pic_order_cnt_lsb	6
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	0
ref_pic_list_reordering()	
pred_weight_table()	
dec_ref_pic_marking()	
cabac_init_idc	0
slice_qp_delta	3
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	-1
slice_beta_offset_div2	-1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

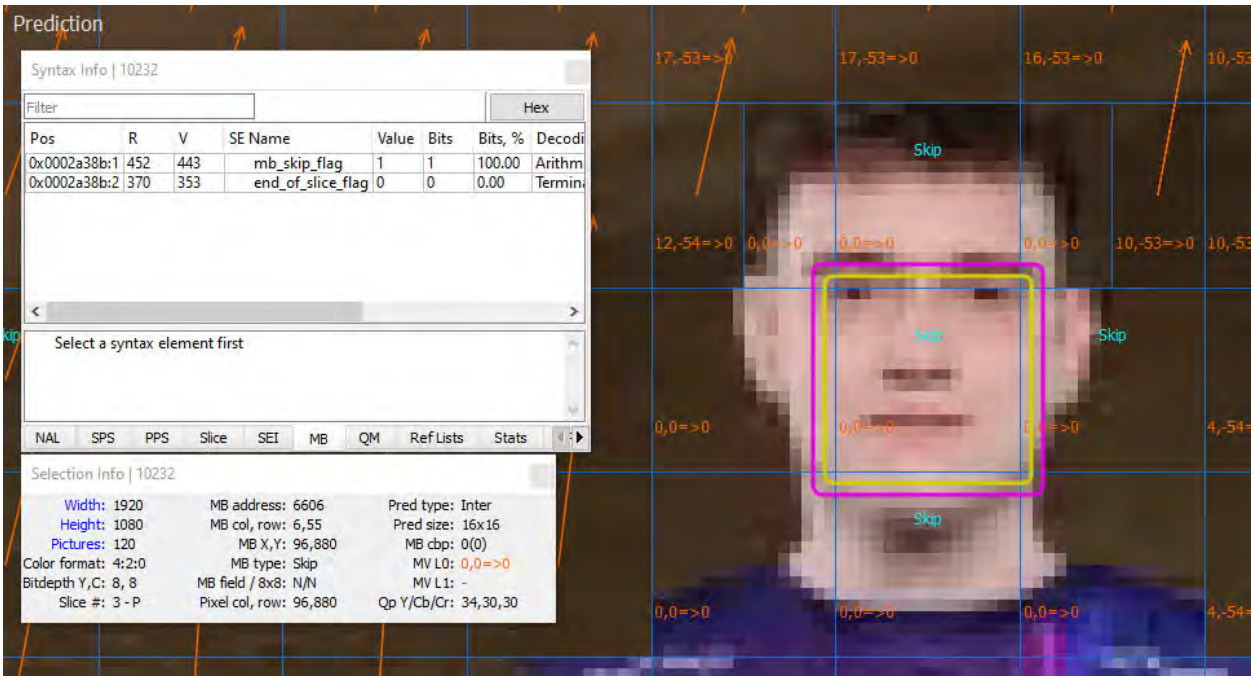
NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

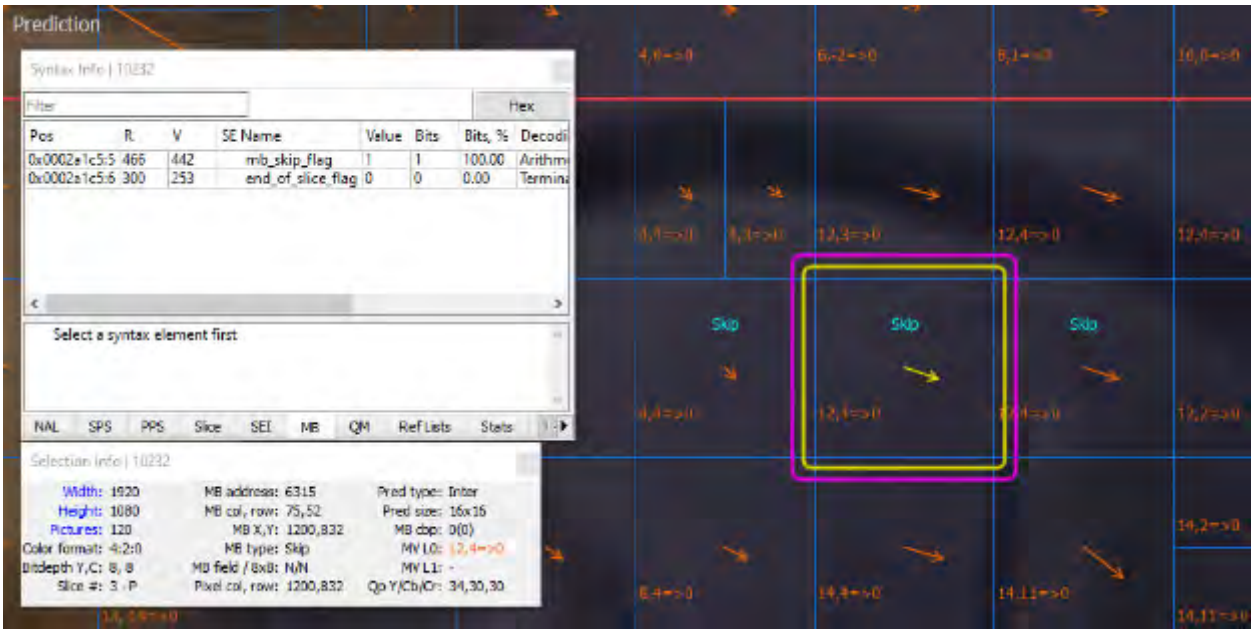


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

224. When encoding Twitch.tv video, for example, using Context-based Adaptive Binary Arithmetic Coding (CABAC), on information and belief, Amazon performs assigning either a zero motion vector or a predicted non-zero motion vector for the skip coding mode for the first segment based at least in part on the motion information of a second segment neighboring the first segment, as demonstrated in the screenshots below using VQ Analyzer software.

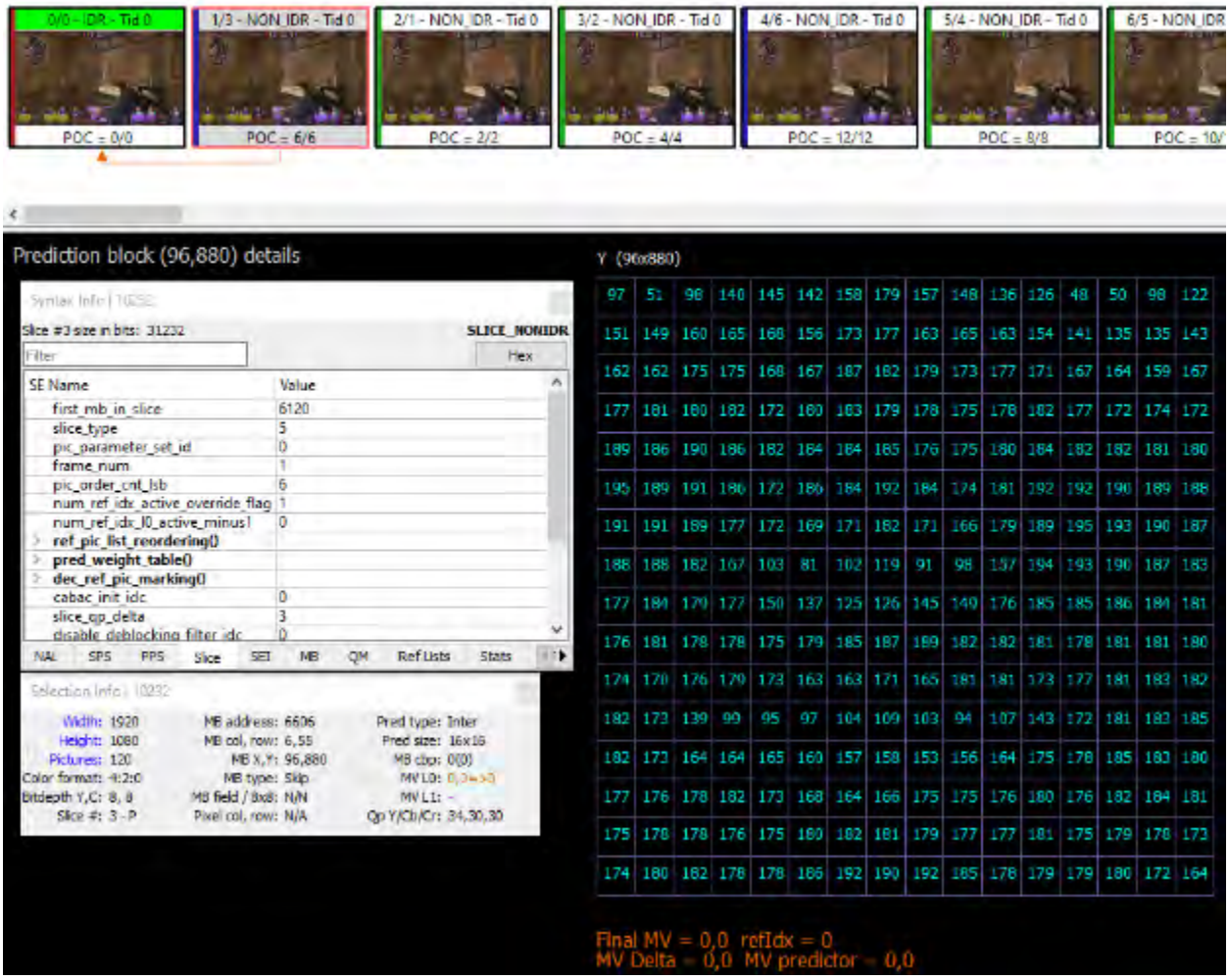


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

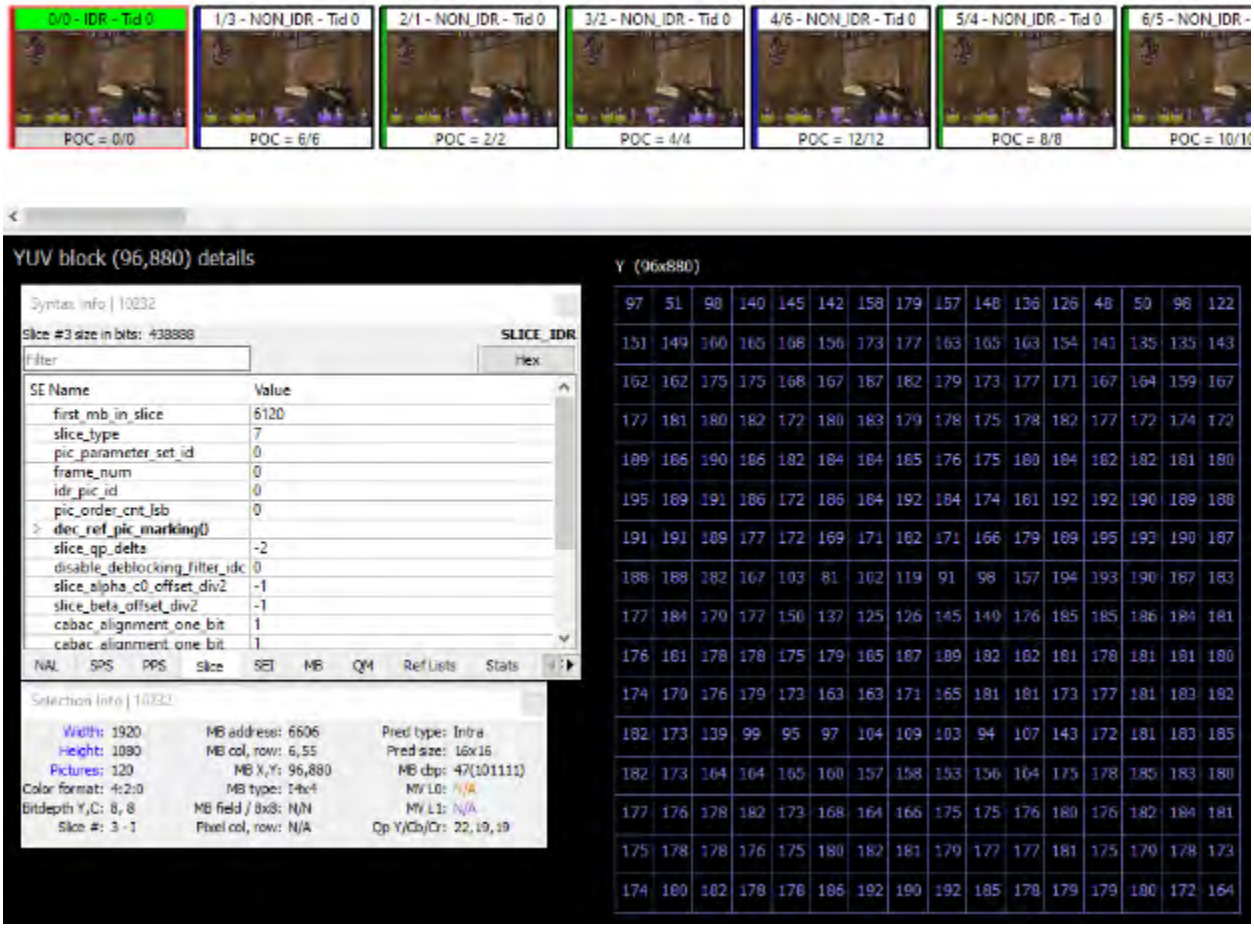


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

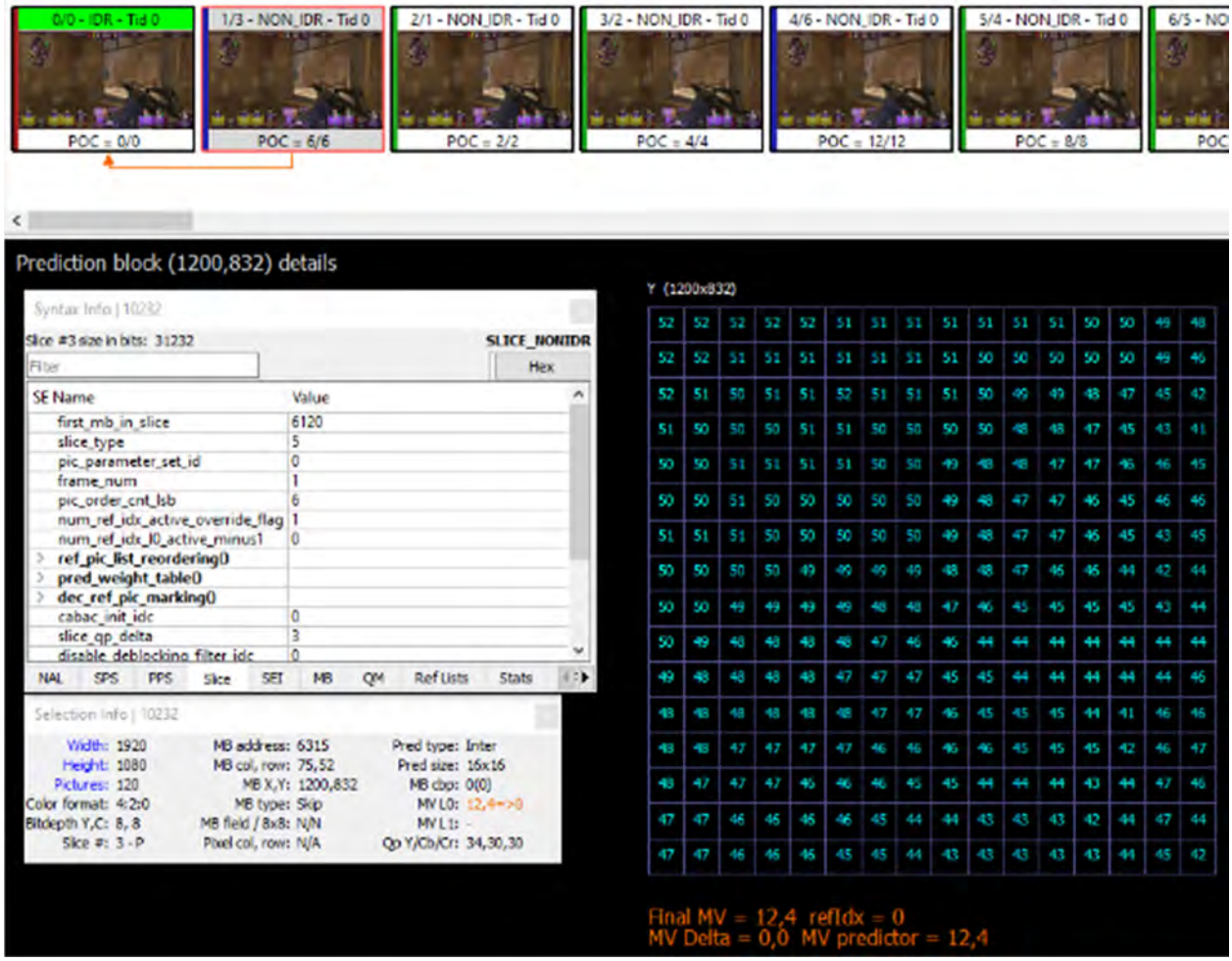
225. When encoding Twitch.tv video, for example, using Context-based Adaptive Binary Arithmetic Coding (CABAC), on information and belief, Amazon performs forming a prediction for the first segment with respect to a reference frame based at least in part on the assigned motion vector for the skip coding mode, wherein the assigned motion vector is one of the zero motion vector and the predicted non-zero motion vector, as demonstrated in the screenshots below using VQ Analyzer software.



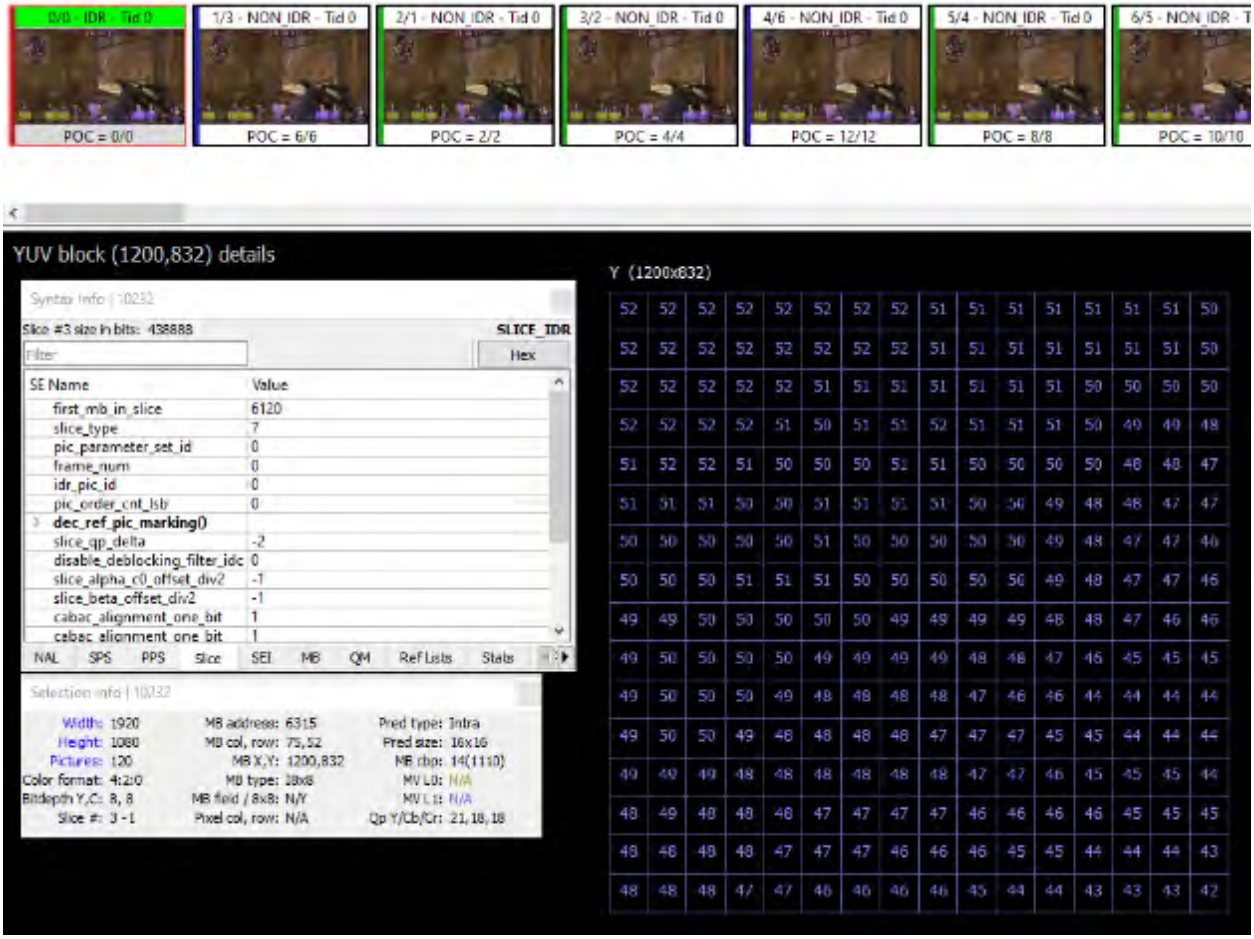
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

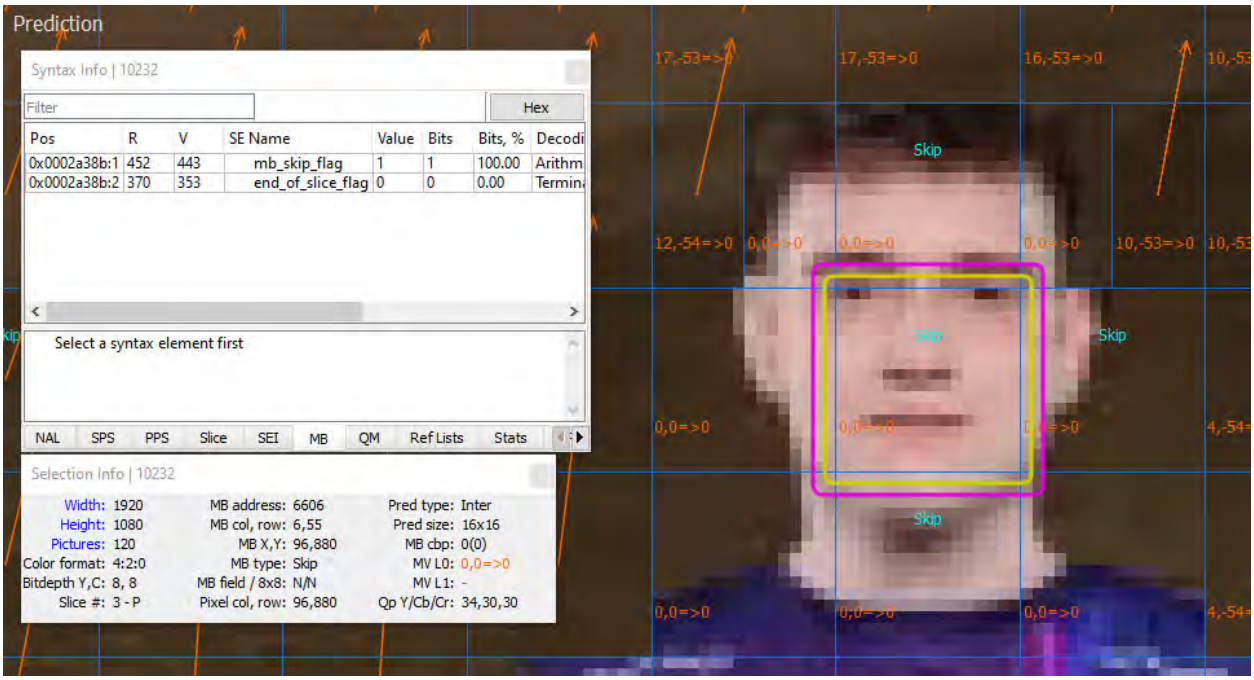


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

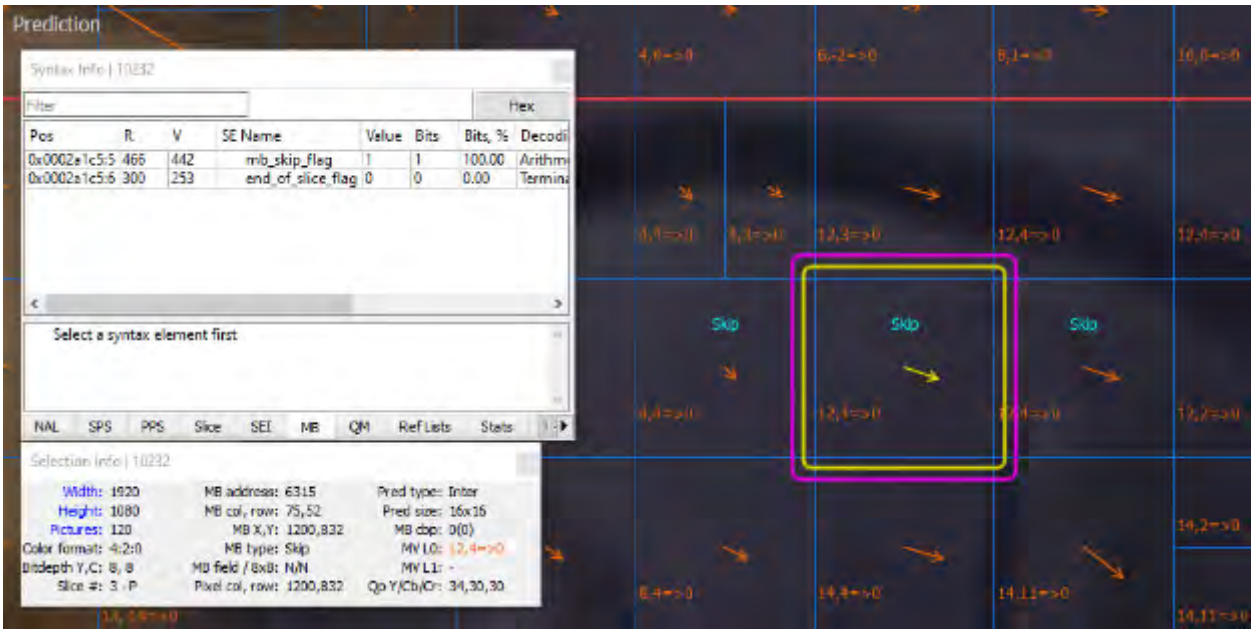


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

226. When encoding Twitch.tv video, for example, using Context-based Adaptive Binary Arithmetic Coding (CABAC), on information and belief, Amazon performs providing in an encoded bitstream an indication of the skip coding mode, wherein no further motion vector information for the first segment is coded in the encoded bitstream, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



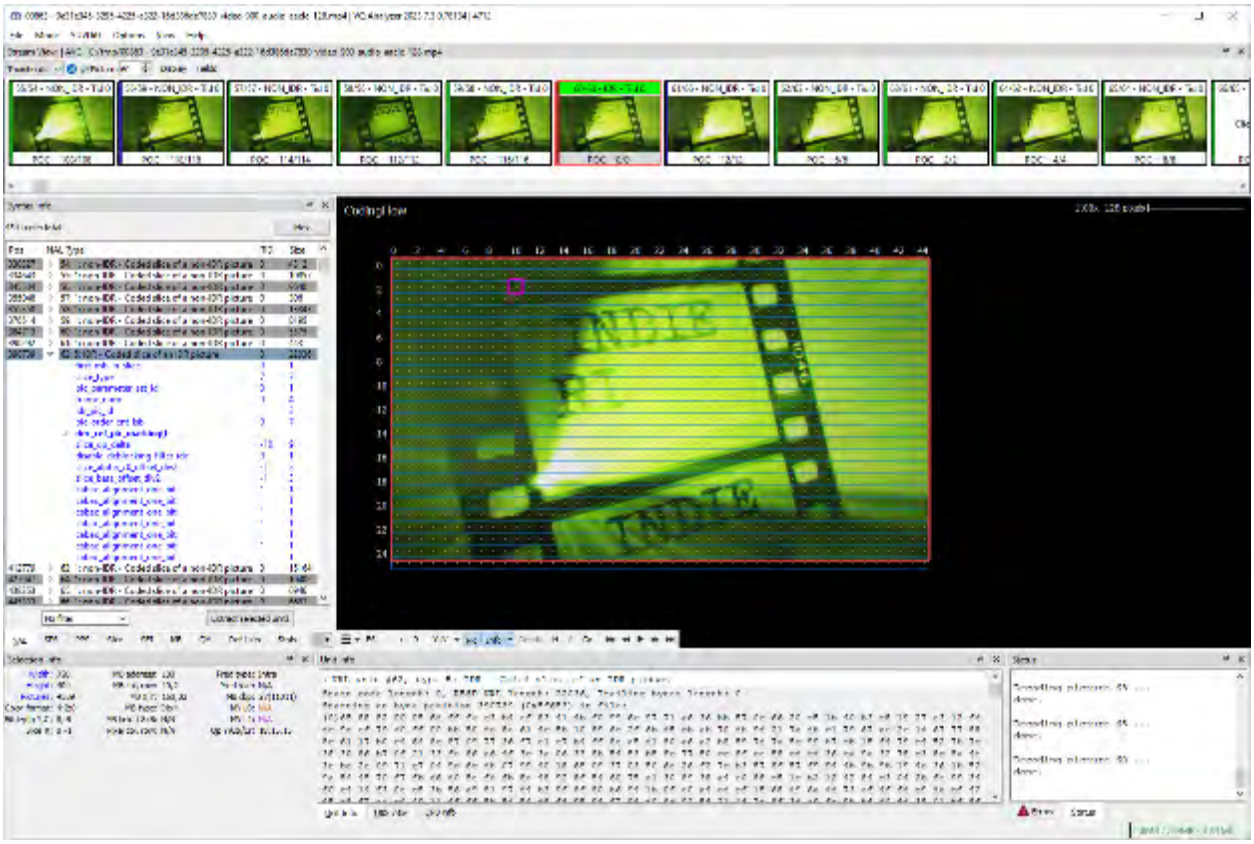
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

B. Amazon Makes, Imports, Uses, Sells, and/or Offers for Sale Products and Services that Infringe the ’321 Patent.

227. The Accused Products infringe one or more claims of the ’321 Patent, including, for example, claim 1.

228. As just one example of infringement, on information and belief, Amazon performs a method of encoding a video sequence in a manner that is covered by claim 1 of the ’321 Patent for Amazon Prime Video content (such as trailers), as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 212.

229. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs encoding a video sequence comprising an independent sequence of image frames, wherein all motion-compensated temporal prediction references of the independent sequence refer only to image frames within said independent sequence, as demonstrated in the screenshots below using VQ Analyzer software.

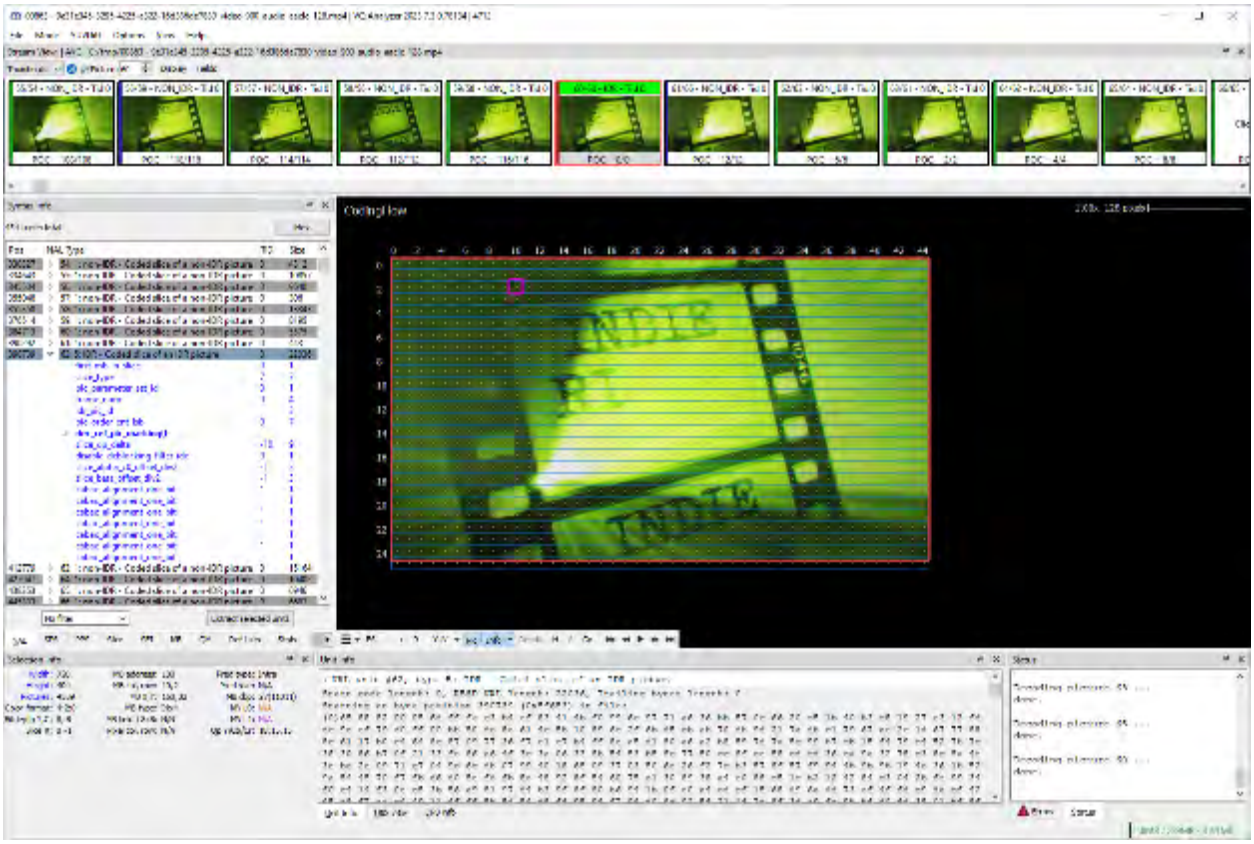


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

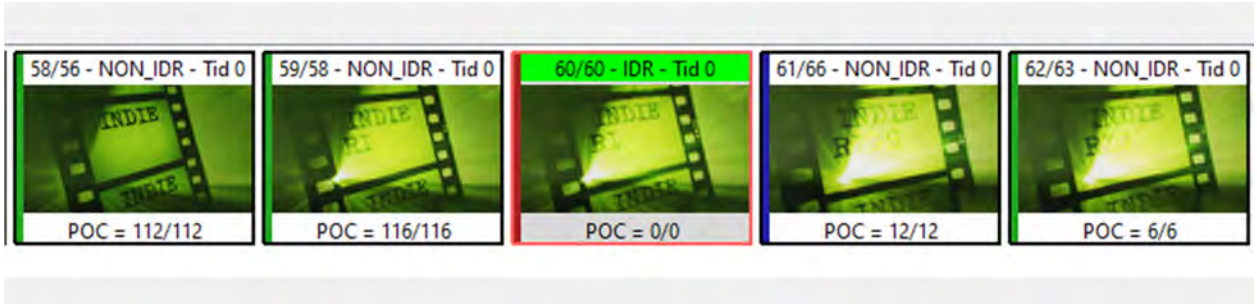
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

230. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs encoding into the video sequence an indication of at least one image frame, which is the first image frame, in decoding order, of the independent sequence, as demonstrated in the screenshots below using VQ Analyzer software.

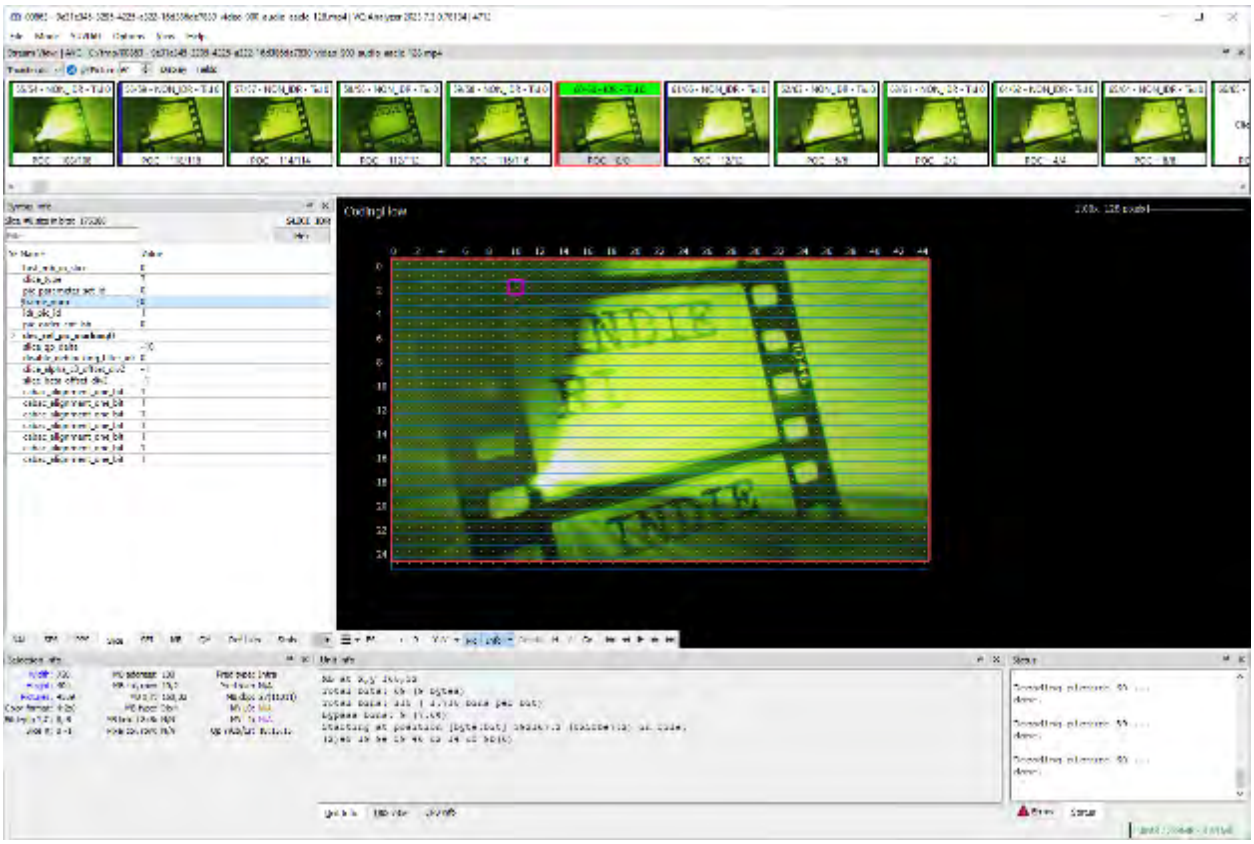


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

231. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs encoding identifier values for the image frames according to a numbering scheme, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

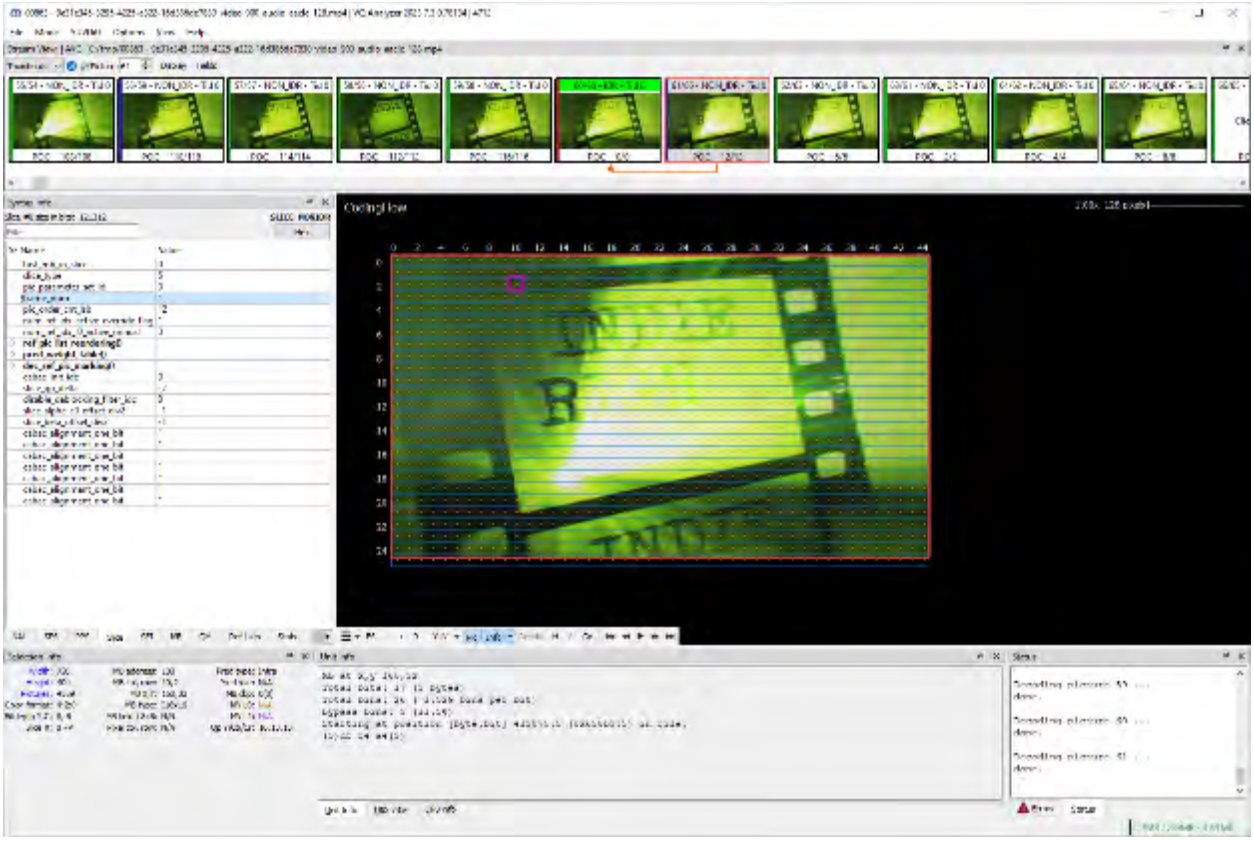
Slice #0 size in bits: 176288 **SLICE_IDR**

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	7
pic_parameter_set_id	0
frame_num	0
idr_pic_id	1
pic_order_cnt_lsb	0
dec_ref_pic_marking()	
slice_qp_delta	-10
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	-1
slice_beta_offset_div2	-1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

Slice #0 size in bits: 121312 SLICE_NONIDR

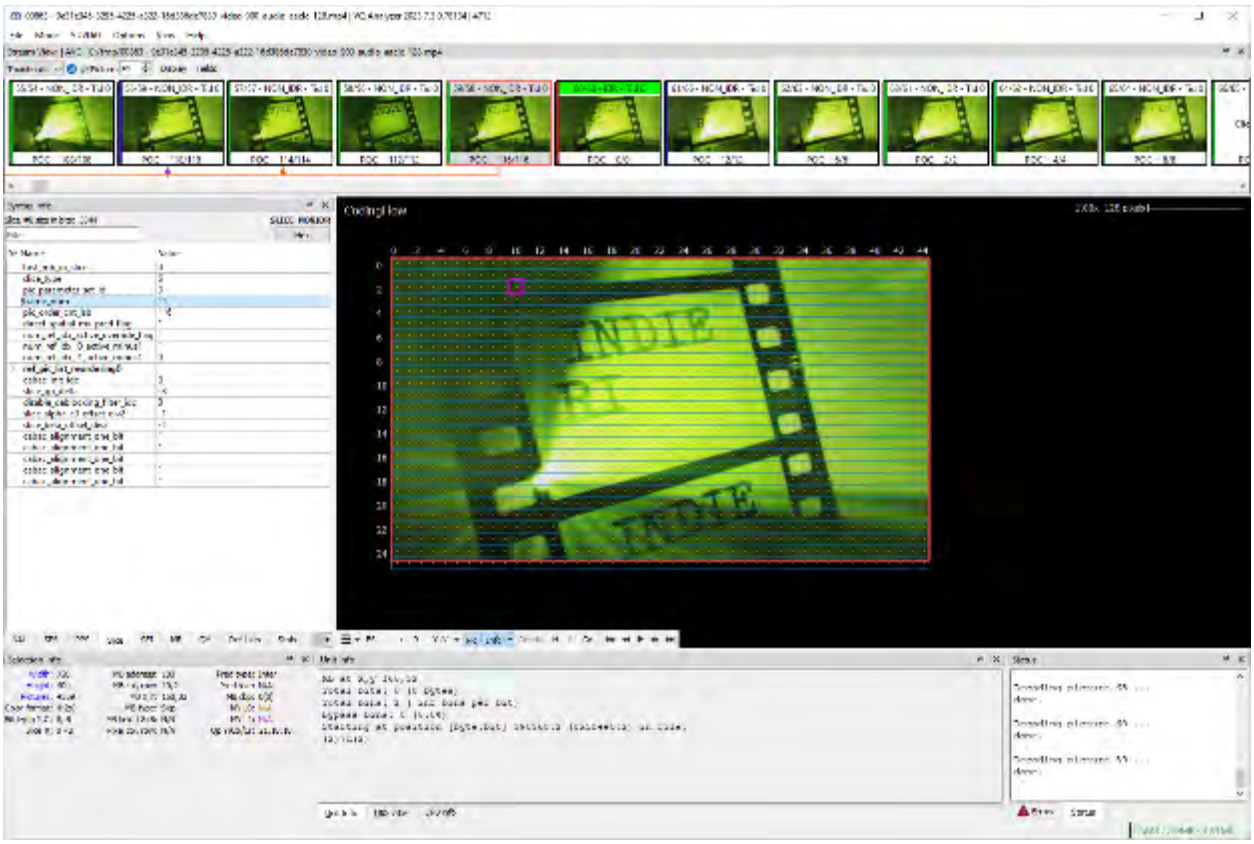
Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	5
pic_parameter_set_id	0
frame_num	1
pic_order_cnt_lsb	12
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	0
ref_pic_list_reordering()	
pred_weight_table()	
dec_ref_pic_marking()	
cabac_init_idc	0
slice_qp_delta	-7
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	-1
slice_beta_offset_div2	-1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

232. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs resetting the identifier value for the indicated first image frame of the independent sequence, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

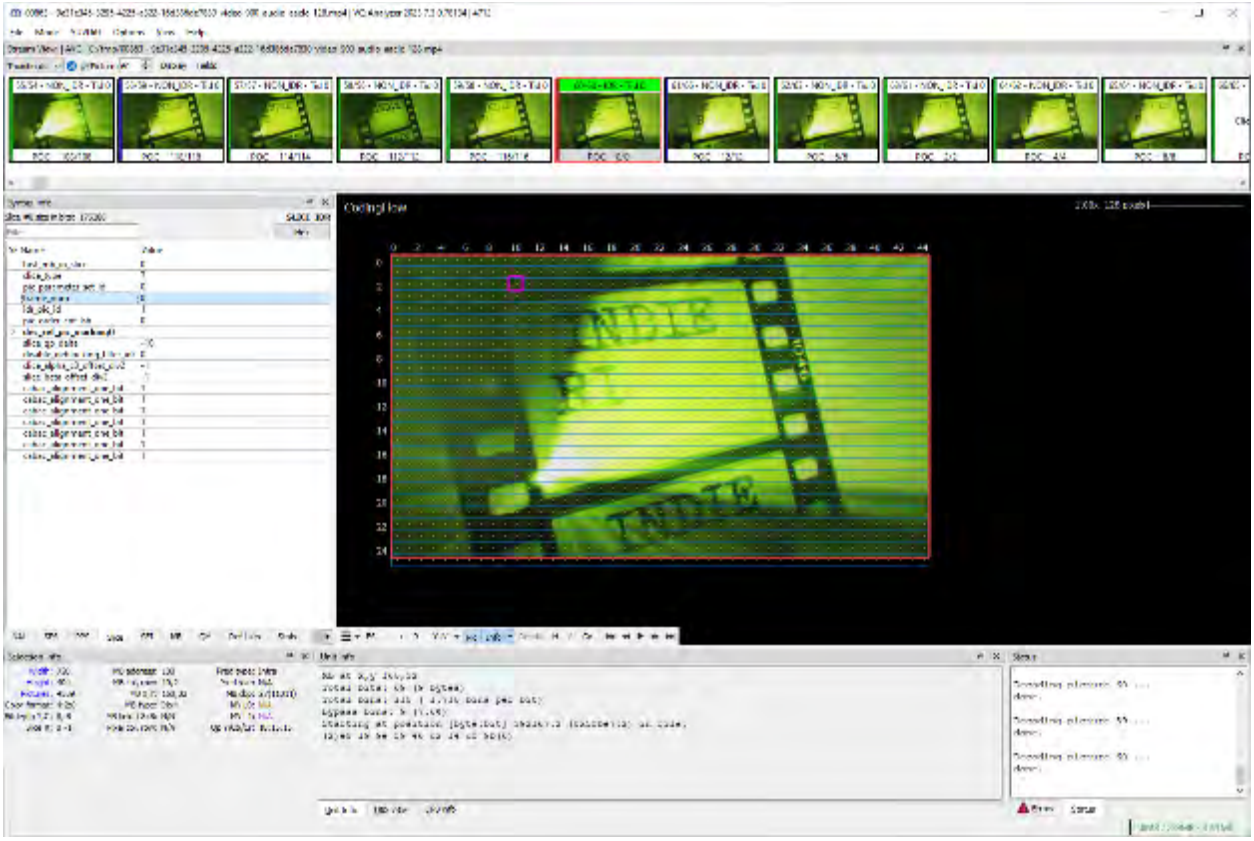
Slice #0 size in bits: 3544 **SLICE_NONIDR**

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	6
pic_parameter_set_id	0
frame_num	11
pic_order_cnt_lsb	116
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	1
num_ref_idx_l1_active_minus1	0
ref_pic_list_reordering()	
cabac_init_idc	0
slice_qp_delta	-8
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	-1
slice_beta_offset_div2	-1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

Slice #0 size in bits: 176288 SLICE_IDR

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	7
pic_parameter_set_id	0
frame_num	0
idr_pic_id	1
pic_order_cnt_lsb	0
dec_ref_pic_marking()	
slice_qp_delta	-10
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	-1
slice_beta_offset_div2	-1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

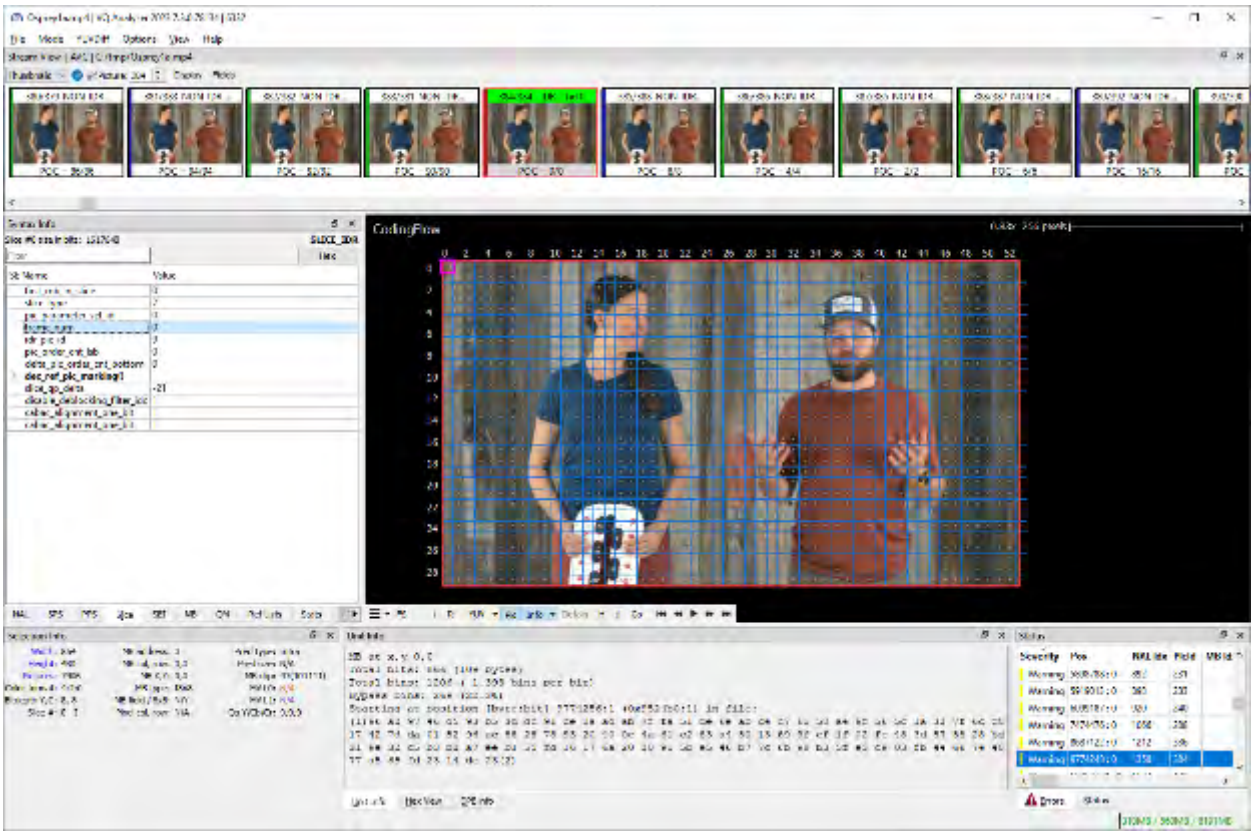
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

233. For another example, on information and belief, Amazon performs a method of encoding video in a manner that is substantially the same as described above when Amazon Prime Video contents are encoded to produce a bitstream that indicates it can be decoded by an H.265-compliant decoder, such that it is covered by claim 1 of the ’321 Patent.

Screenshots of Amazon Prime Video indicating "Codec: hevc."

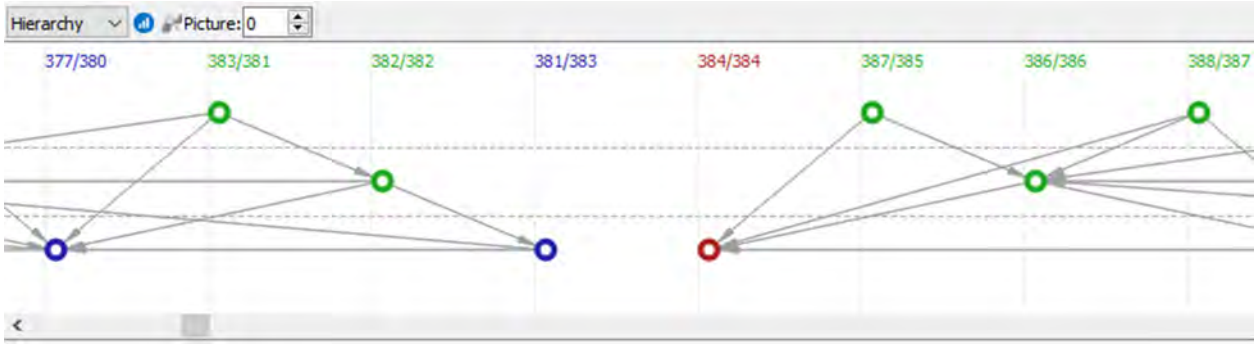
234. For another example, on information and belief, Amazon performs a method of encoding a video sequence in a manner that is covered by claim 1 of the '321 Patent for Amazon.com advertisements, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 217.

235. When encoding Amazon.com advertisements, on information and belief, Amazon performs encoding a video sequence comprising an independent sequence of image frames, wherein all motion-compensated temporal prediction references of the independent sequence refer only to image frames within said independent sequence, as demonstrated in the screenshots below using VQ Analyzer software.



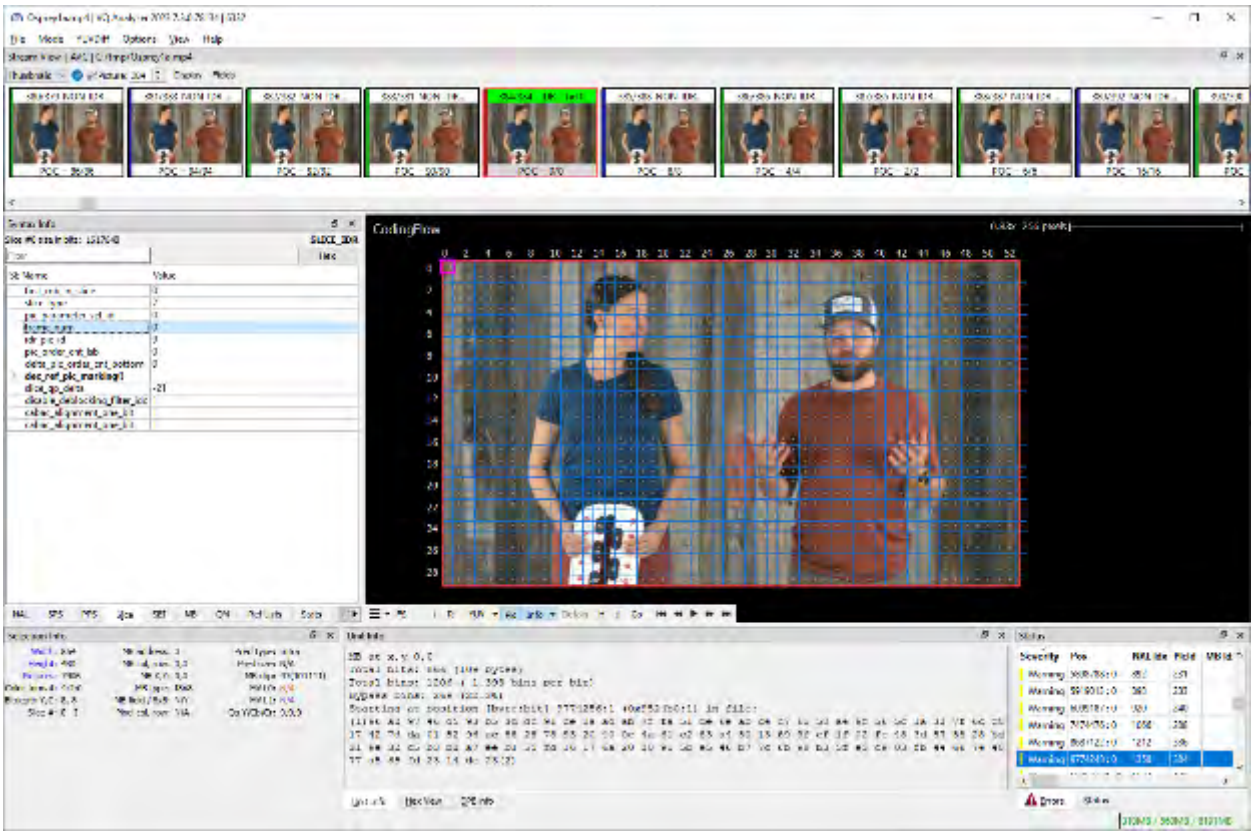
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

236. When encoding Amazon.com advertisements, on information and belief, Amazon performs encoding into the video sequence an indication of at least one image frame, which is the first image frame, in decoding order, of the independent sequence, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info

Slice #0 size in bits: 1617648 SLICE_IDR

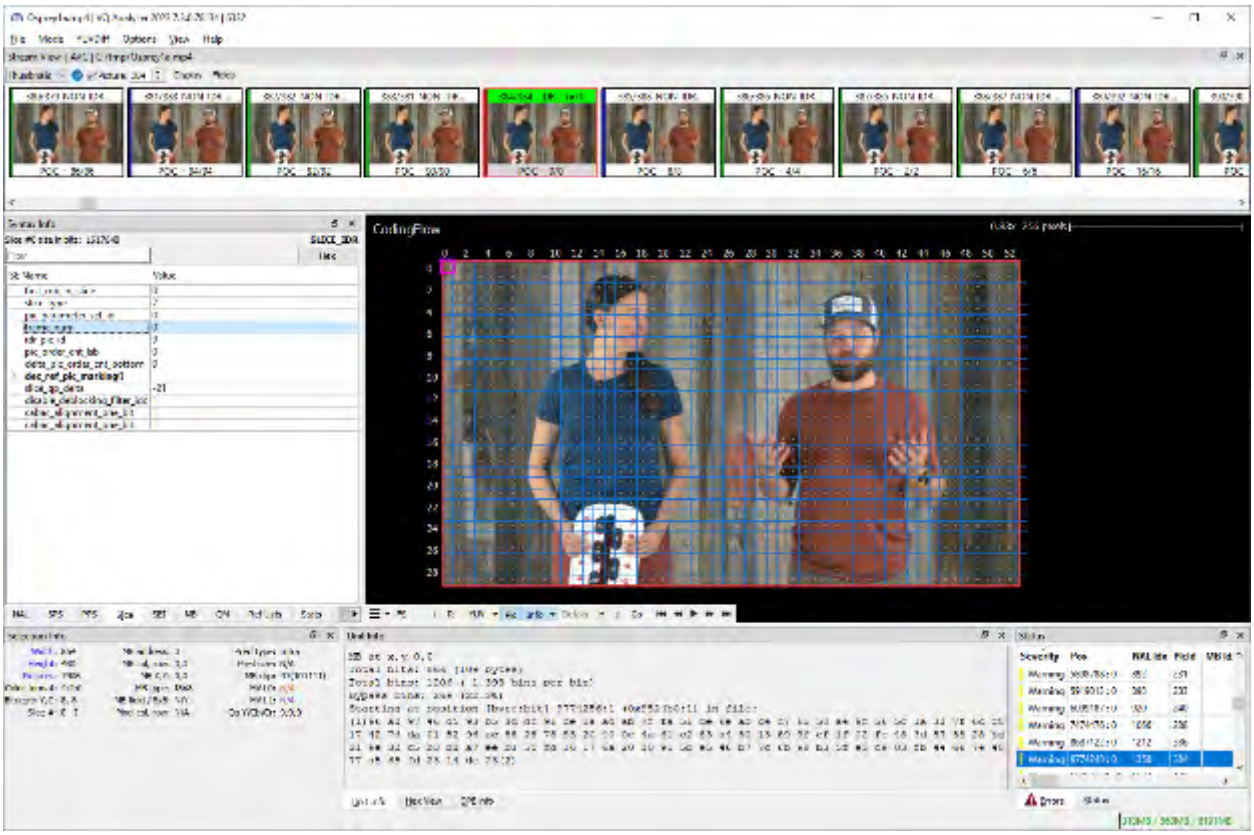
Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	7
pic_parameter_set_id	0
frame_num	0
idr_pic_id	9
pic_order_cnt_lsb	0
delta_pic_order_cnt_bottom	0
dec_ref_pic_marking()	
slice_qp_delta	-21
disable_deblocking_filter_idc	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

237. When encoding Amazon.com advertisements, on information and belief, Amazon performs encoding identifier values for the image frames according to a numbering scheme, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info

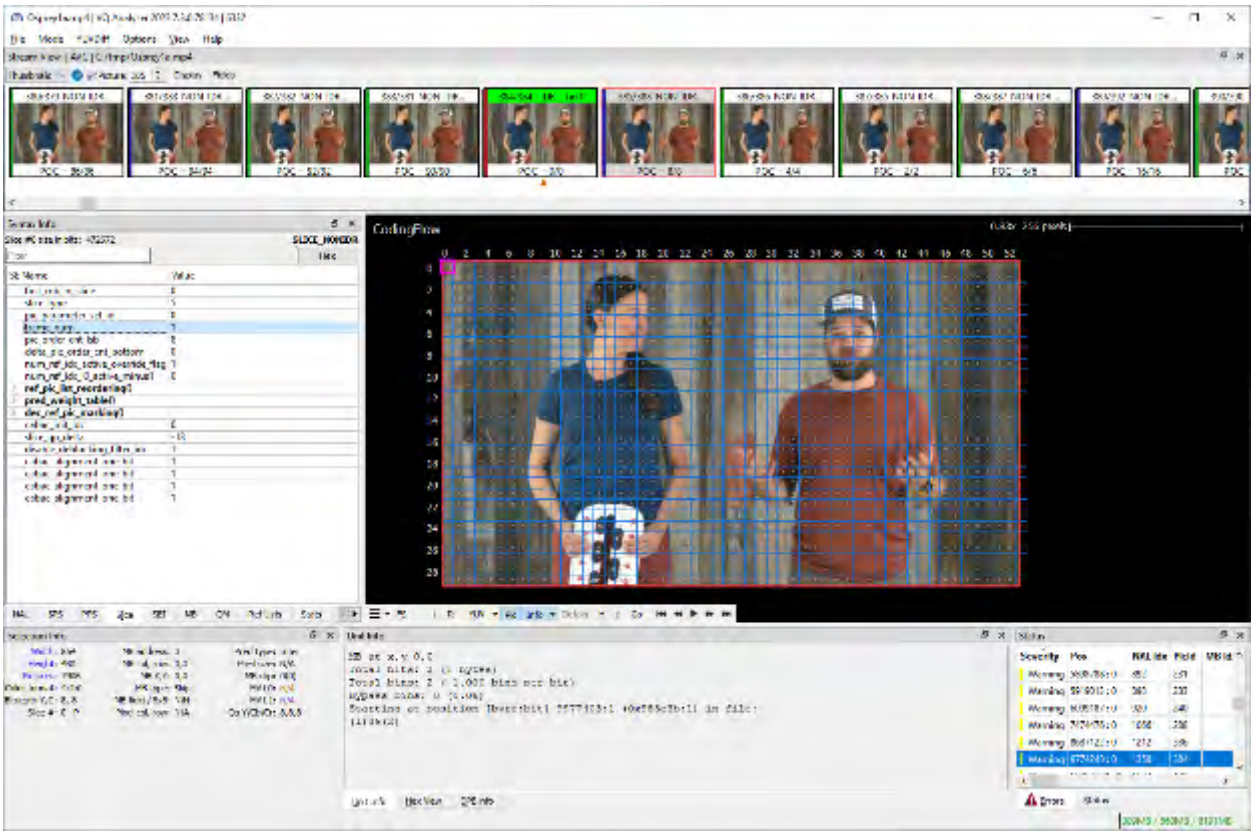
Slice #0 size in bits: 1617648 SLICE_IDR

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	7
pic_parameter_set_id	0
frame_num	0
idr_pic_id	9
pic_order_cnt_lsb	0
delta_pic_order_cnt_bottom	0
dec_ref_pic_marking()	
slice_qp_delta	-21
disable_deblocking_filter_idc	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info

Slice #0 size in bits: 472672 SLICE_NONIDR

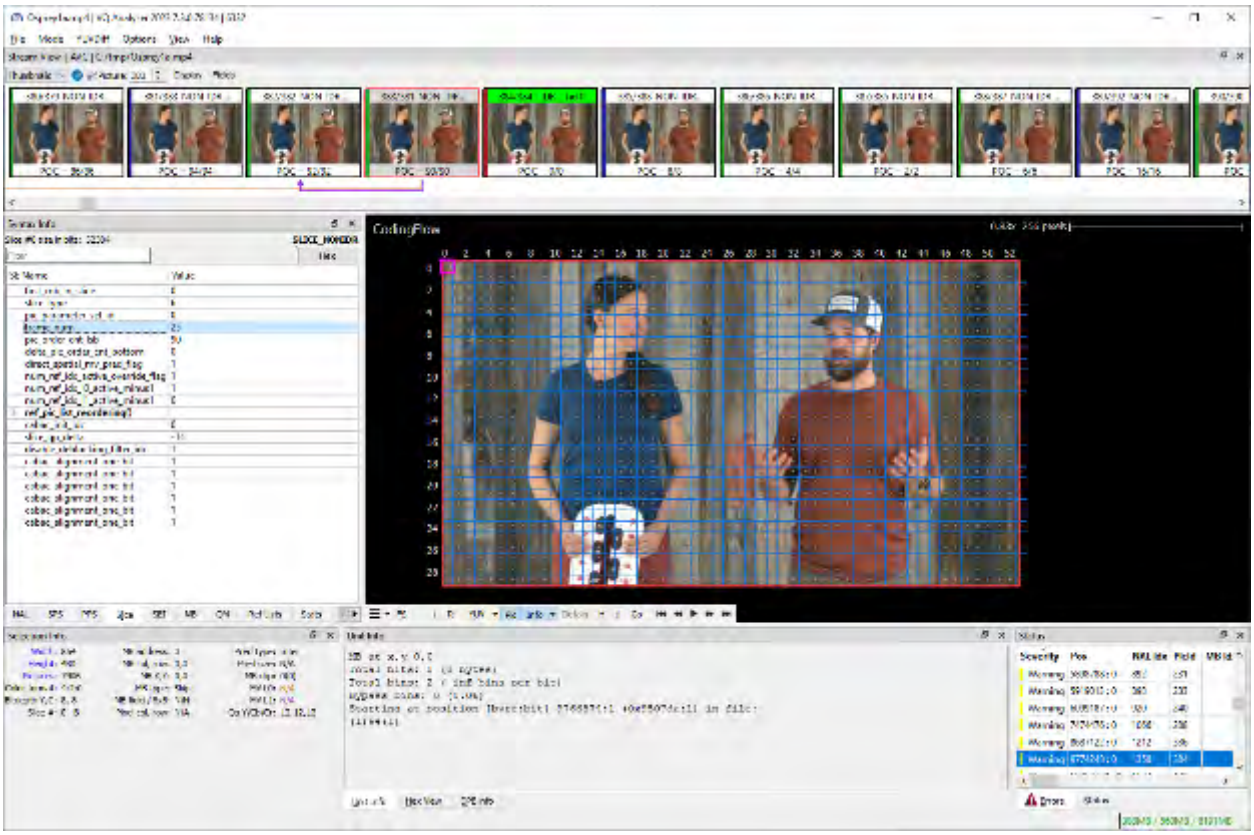
Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	5
pic_parameter_set_id	0
frame_num	1
pic_order_cnt_lsb	8
delta_pic_order_cnt_bottom	0
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	0
ref_pic_list_reordering()	
pred_weight_table()	
dec_ref_pic_marking()	
cabac_init_idc	0
slice_qp_delta	-18
disable_deblocking_filter_idc	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

238. When encoding Amazon.com advertisements, on information and belief, Amazon performs resetting the identifier value for the indicated first image frame of the independent sequence, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info

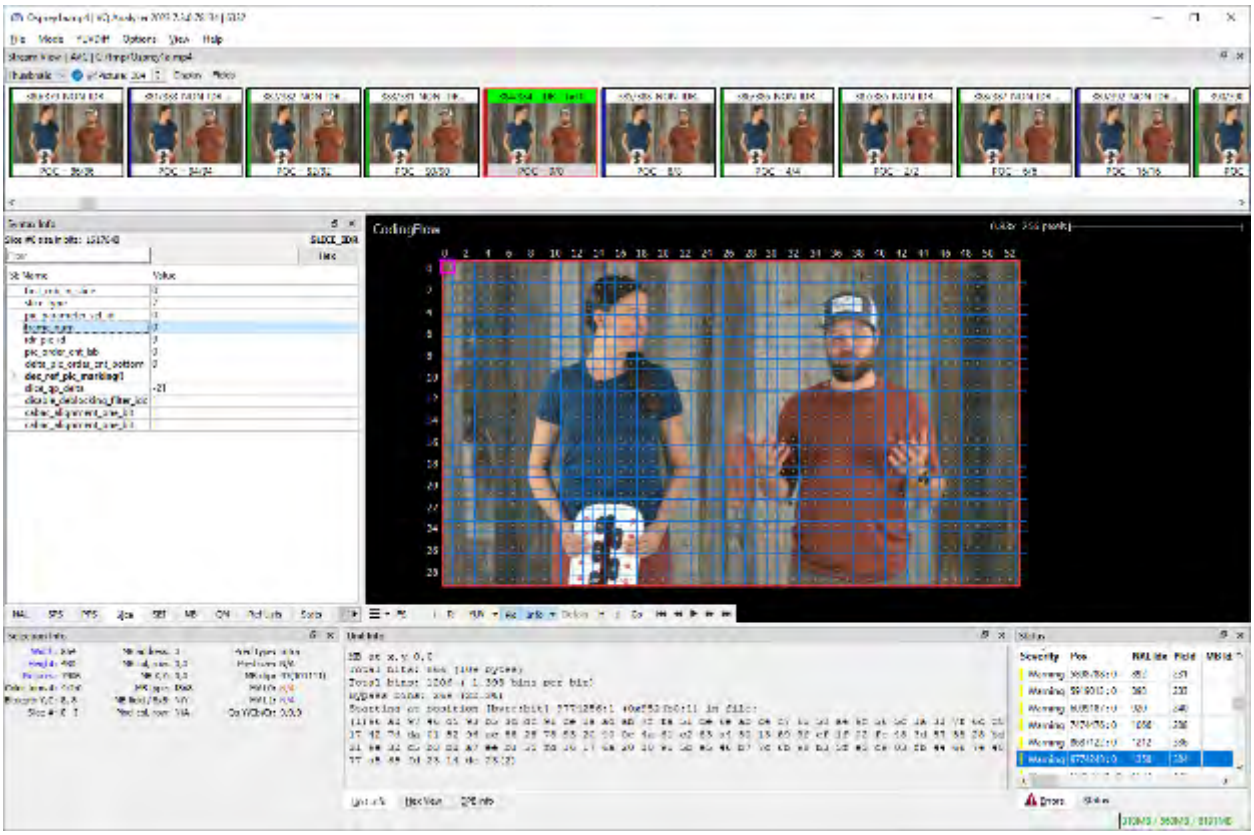
Slice #0 size in bits: 52384 SLICE_NONIDR

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	6
pic_parameter_set_id	0
frame_num	25
pic_order_cnt_lsb	90
delta_pic_order_cnt_bottom	0
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	1
num_ref_idx_l1_active_minus1	0
ref_pic_list_reordering()	
cabac_init_idc	0
slice_qp_delta	-14
disable_deblocking_filter_idc	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info

Slice #0 size in bits: 1617648 SLICE_IDR

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	7
pic_parameter_set_id	0
frame_num	0
idr_pic_id	9
pic_order_cnt_lsb	0
delta_pic_order_cnt_bottom	0
dec_ref_pic_marking()	
slice_qp_delta	-21
disable_deblocking_filter_idc	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

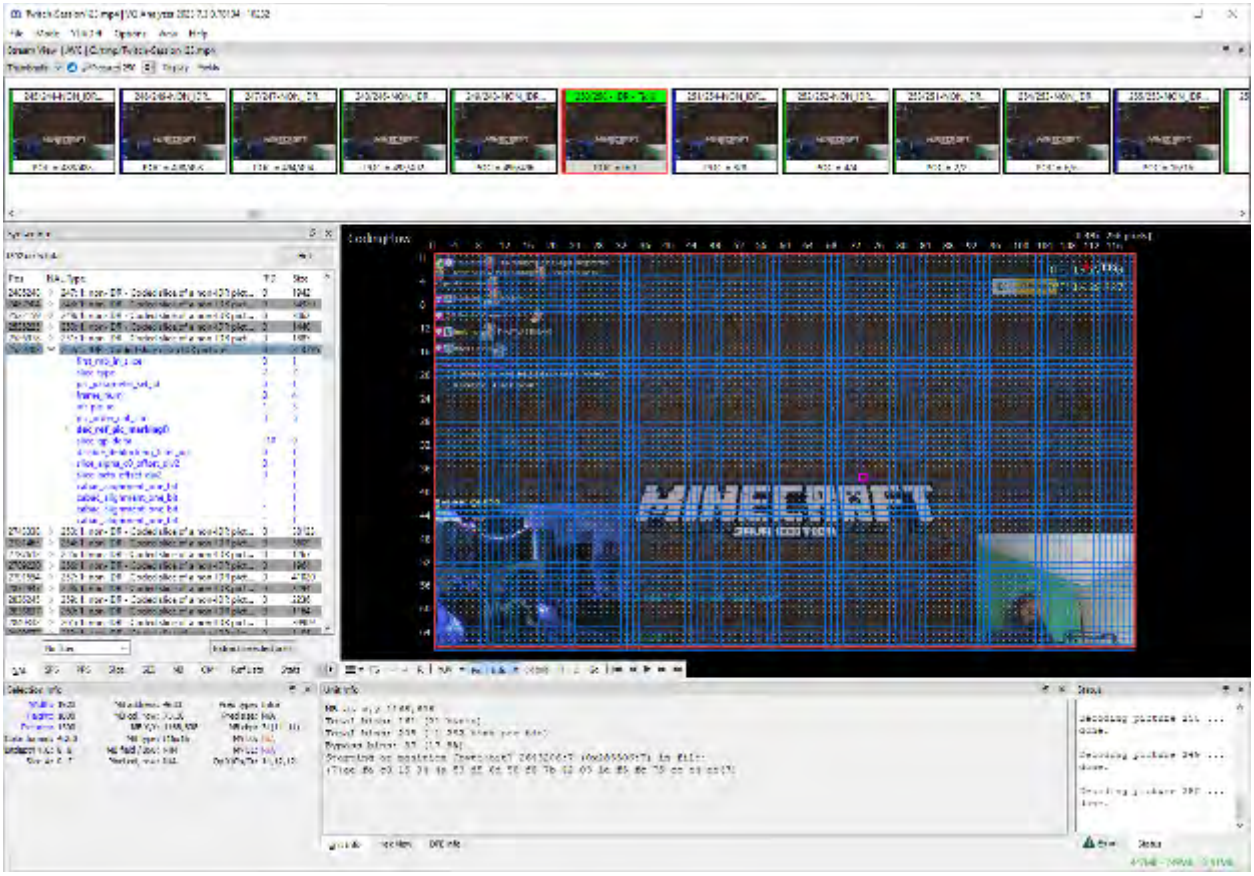
NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

239. For another example, on information and belief, Amazon performs a method of encoding video in a manner that is substantially the same as described above when Amazon.com videos are encoded to produce a bitstream that indicates it can be decoded by an H.265-compliant decoder, such that it is covered by claim 1 of the ’321 Patent.

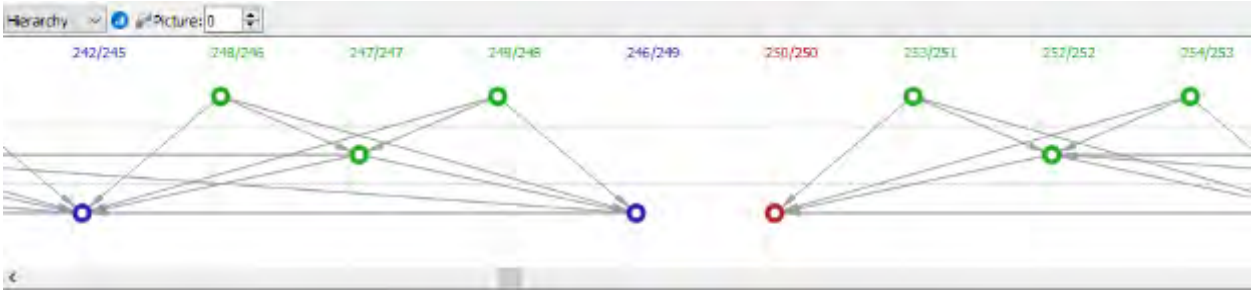
240. For another example, on information and belief, Amazon performs a method of encoding a video sequence in a manner that is covered by claim 1 of the ’321 Patent for Twitch.tv video, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 222.

241. When encoding Twitch.tv video, on information and belief, Amazon performs encoding a video sequence comprising an independent sequence of image frames, wherein all motion-compensated temporal prediction references of the independent sequence refer only to image frames within said independent sequence, as demonstrated in the screenshots below using VQ Analyzer software.



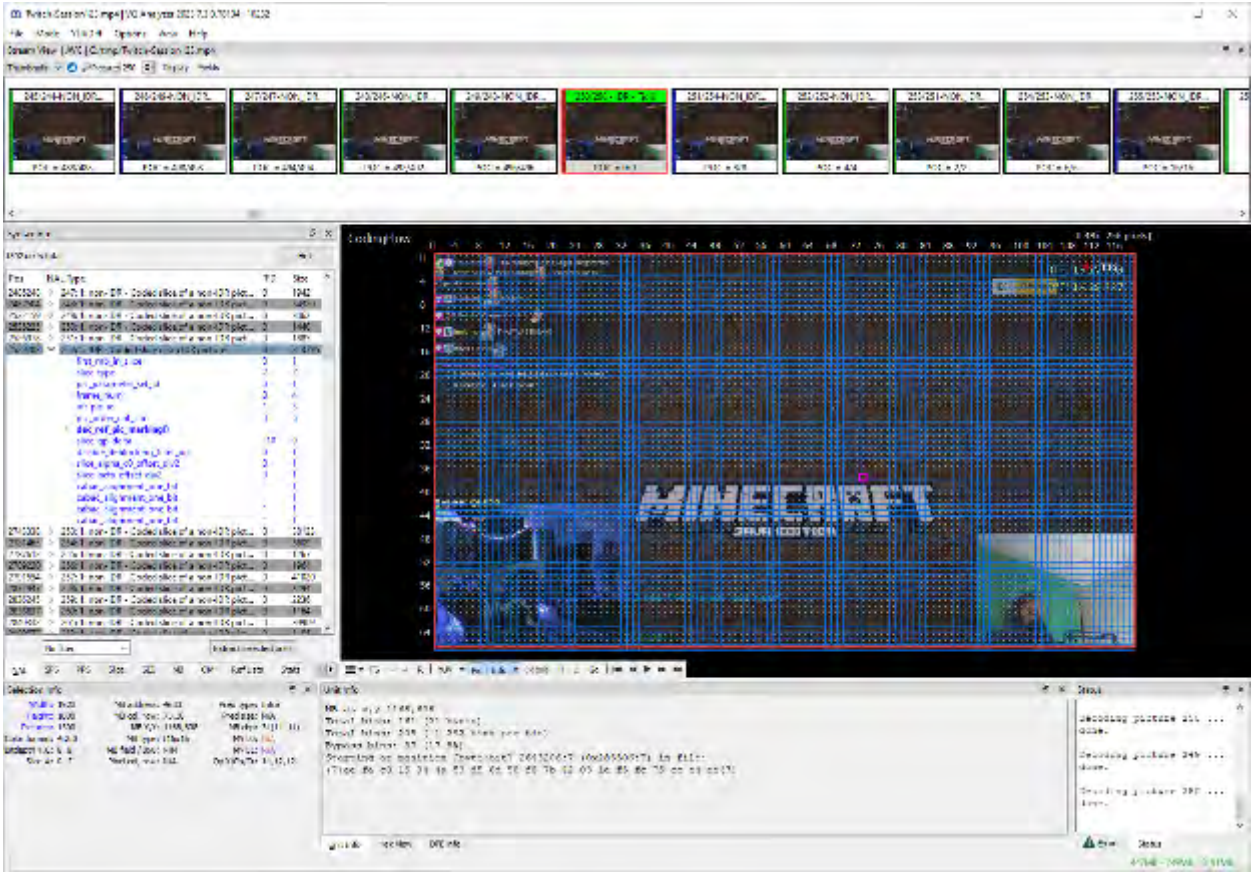
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

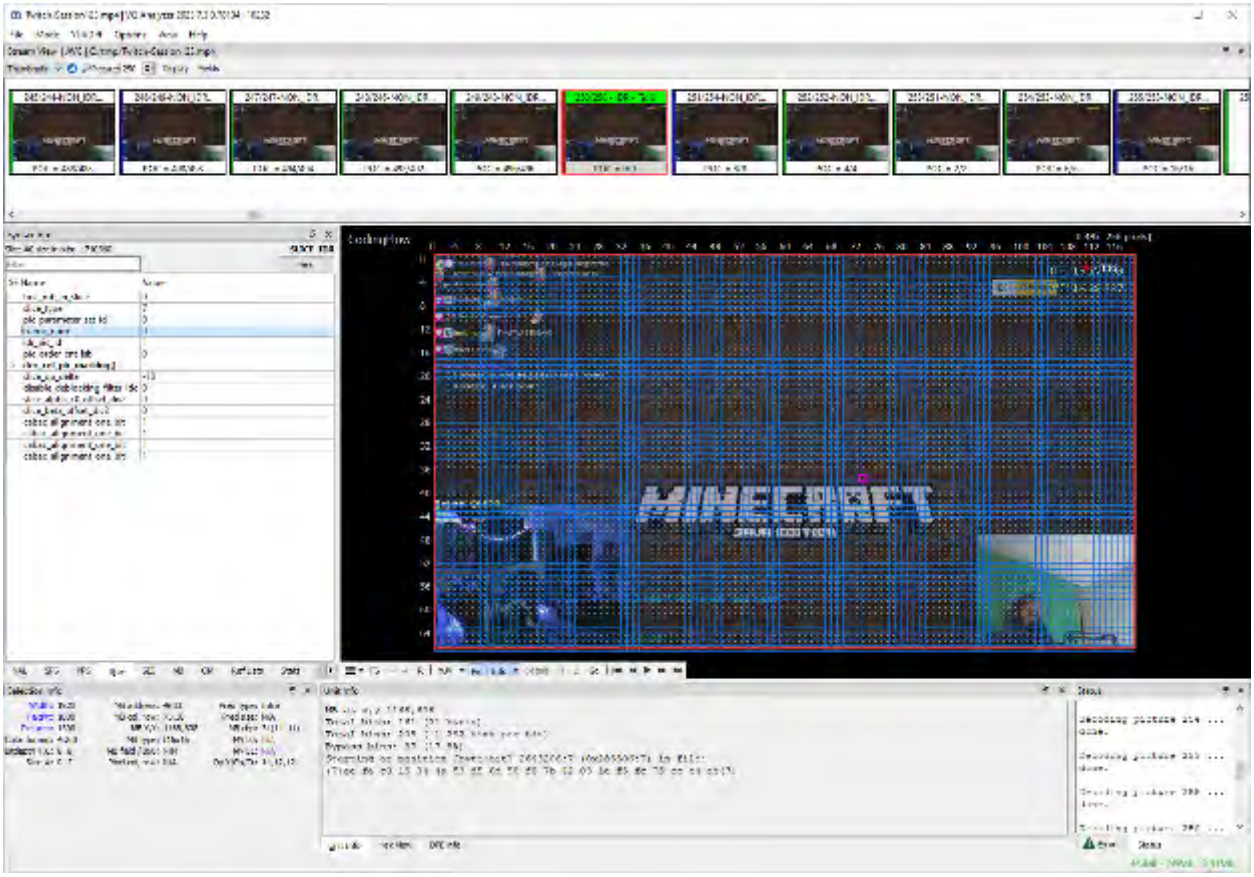
242. When encoding Twitch.tv video, on information and belief, Amazon performs encoding into the video sequence an indication of at least one image frame, which is the first image frame, in decoding order, of the independent sequence, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

243. When encoding Twitch.tv video, on information and belief, Amazon performs encoding identifier values for the image frames according to a numbering scheme, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info

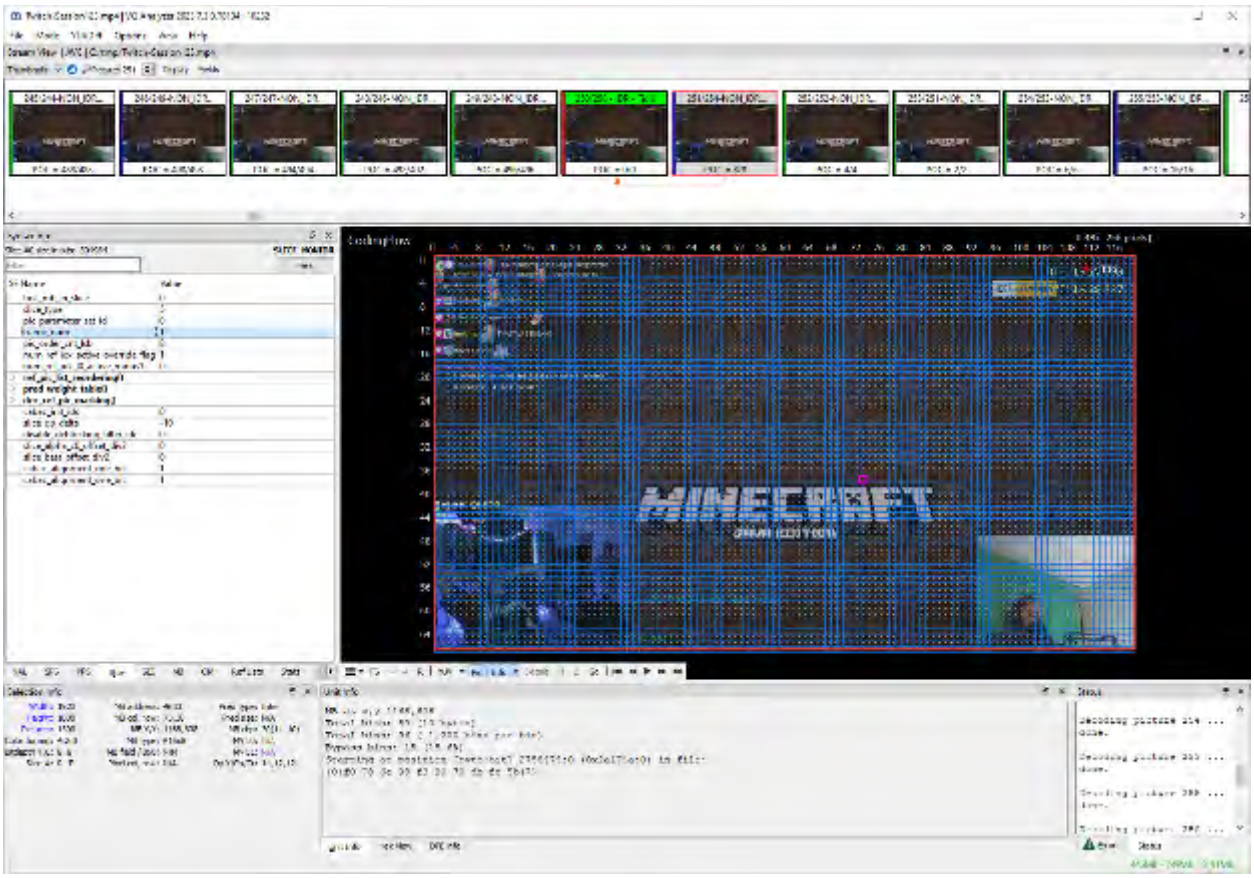
Slice #0 size in bits: 1710360 **SLICE_IDR**

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	7
pic_parameter_set_id	0
frame_num	0
idr_pic_id	1
pic_order_cnt_lsb	0
dec_ref_pic_marking()	
slice_qp_delta	-10
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	0
slice_beta_offset_div2	0
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info

Slice #0 size in bits: 304984 SLICE_NONIDR

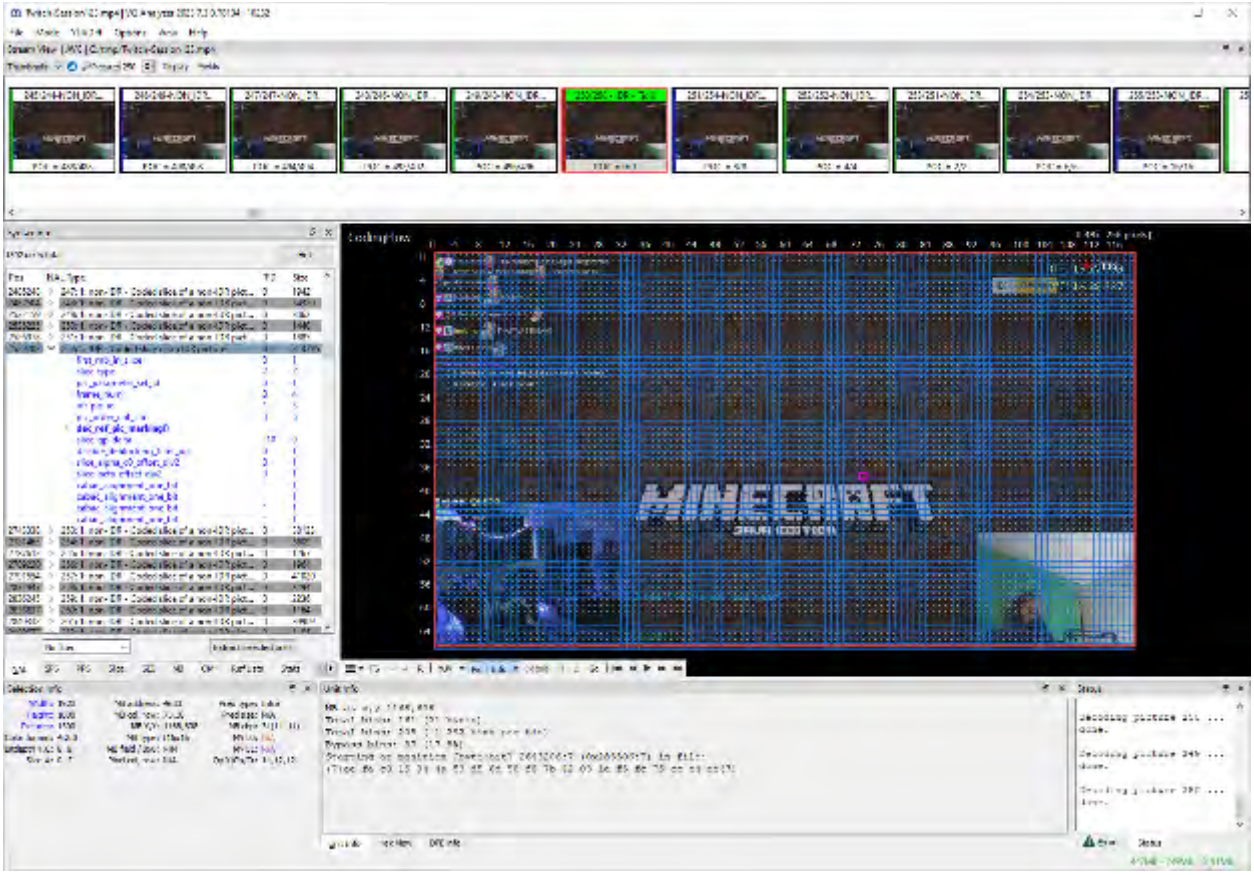
Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	5
pic_parameter_set_id	0
frame_num	1
pic_order_cnt_lsb	8
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	0
ref_pic_list_reordering()	
pred_weight_table()	
dec_ref_pic_marking()	
cabac_init_idc	0
slice_qp_delta	-10
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	0
slice_beta_offset_div2	0
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

244. When encoding Twitch.tv video, on information and belief, Amazon performs resetting the identifier value for the indicated first image frame of the independent sequence, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info

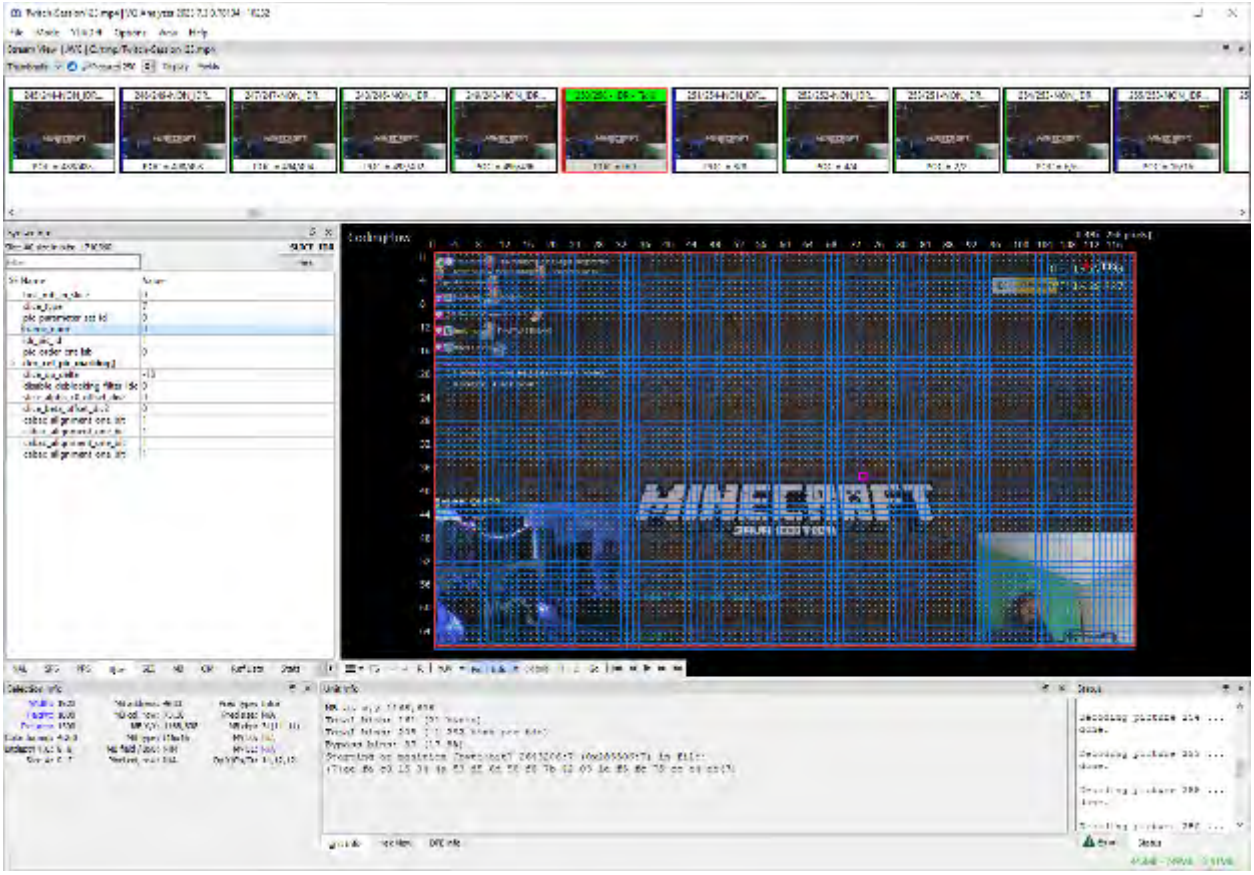
Slice #0 size in bits: 15080 SLICE_NONIDR

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	6
pic_parameter_set_id	0
frame_num	14
pic_order_cnt_lsb	48
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	0
ref_pic_list_reordering()	
cabac_init_idc	0
slice_qp_delta	-1
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	0
slice_beta_offset_div2	0
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info

Slice #0 size in bits: 1710360 SLICE_IDR

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	7
pic_parameter_set_id	0
frame_num	0
idr_pic_id	1
pic_order_cnt_lsb	0
dec_ref_pic_marking()	
slice_qp_delta	-10
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	0
slice_beta_offset_div2	0
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

C. Amazon Makes, Imports, Uses, Sells, and/or Offers for Sale Products and Services that Infringe the ’818 Patent.

245. The Accused Products infringe one or more claims of the ’818 Patent, including, for example, claim 1.

246. As just one example of infringement, on information and belief, Amazon performs a method of encoding video in a manner that is covered by claim 1 of the ’818 Patent for Amazon Prime Video content (such as trailers), as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 212.

247. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs encoding sequences of pictures into a bitstream, wherein parameters are defined in a

parameter set and each picture comprises information of one or more slices, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info

4541 units total | Hex

Pos	NAL Type	TID	Size
553	0: 7: SPS - Sequence parameter set	0	44
600	1: 8: PPS - Picture parameter set	0	5
79621	2: 5: IDR - Coded slice of an IDR picture	0	99
79724	3: 1: non-IDR - Coded slice of a non-IDR picture	0	212
79940	4: 1: non-IDR - Coded slice of a non-IDR picture	0	61
80005	5: 1: non-IDR - Coded slice of a non-IDR picture	0	19
80028	6: 1: non-IDR - Coded slice of a non-IDR picture	0	21

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

Filter | Hex

SE Name	Value
profile_idc	77
constraint_set0_flag	0
constraint_set1_flag	1
constraint_set2_flag	0
constraint_set3_flag	0
constraint_set4_flag	0
constraint_set5_flag	0
reserved_zero_2bits	0
level_idc	30
seq_parameter_set_id	0
log2_max_frame_num_minus4	0
pic_order_cnt_type	0
log2_max_pic_order_cnt_lsb_minus4	3
max_num_ref_frames	4
gaps_in_frame_num_value_allowed_flag	0
pic_width_in_mbs_minus1	44
pic_height_in_map_units_minus1	25
frame_mbs_only_flag	1
direct_8x8_inference_flag	1
frame_cropping_flag	1
frame_crop_left_offset	0
frame_crop_right_offset	0
frame_crop_top_offset	0
frame_crop_bottom_offset	6
vui_parameters_present_flag	1
vui_parameters()	
rbsp_stop_one_bit	1
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

Filter | Hex

SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	0
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	3
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	-3
deblocking_filter_control_present_flag	1
constrained_intra_pred_flag	0
redundant_pic_cnt_present_flag	0
rbsp_stop_one_bit	1
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

Slice #0 size in bits: 44544 SLICE_NONIDR

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	6
pic_parameter_set_id	0
frame_num	15
pic_order_cnt_lsb	62
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	0
num_ref_idx_l1_active_minus1	1
ref_pic_list_reordering()	
cabac_init_idc	0
slice_qp_delta	-10
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	-1
slice_beta_offset_div2	-1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

248. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs defining, in an encoder, parameter values in a sequence parameter set for a sequence of pictures, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info

4541 units total | Hex

Pos	NAL Type	TID	Size
553	0: 7: SPS - Sequence parameter set	0	44
600	1: 8: PPS - Picture parameter set	0	5
79621	2: 5: IDR - Coded slice of an IDR picture	0	99
79724	3: 1: non-IDR - Coded slice of a non-IDR picture	0	212
79940	4: 1: non-IDR - Coded slice of a non-IDR picture	0	61
80005	5: 1: non-IDR - Coded slice of a non-IDR picture	0	19
80028	6: 1: non-IDR - Coded slice of a non-IDR picture	0	21

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

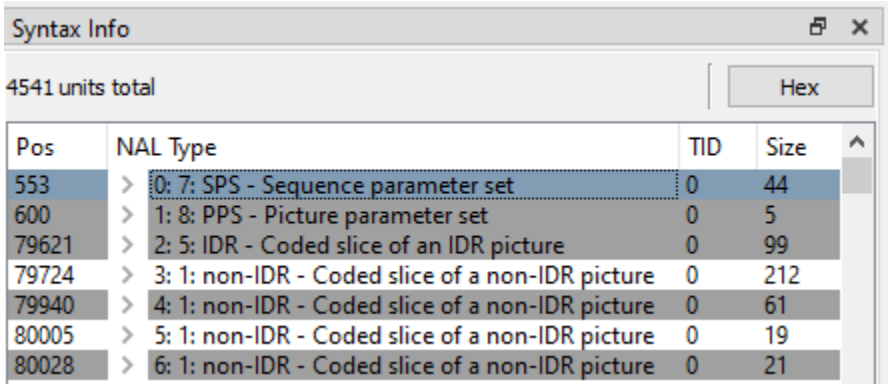
Filter | Hex

SE Name	Value
profile_idc	77
constraint_set0_flag	0
constraint_set1_flag	1
constraint_set2_flag	0
constraint_set3_flag	0
constraint_set4_flag	0
constraint_set5_flag	0
reserved_zero_2bits	0
level_idc	30
seq_parameter_set_id	0
log2_max_frame_num_minus4	0
pic_order_cnt_type	0
log2_max_pic_order_cnt_lsb_minus4	3
max_num_ref_frames	4
gaps_in_frame_num_value_allowed_flag	0
pic_width_in_mbs_minus1	44
pic_height_in_map_units_minus1	25
frame_mbs_only_flag	1
direct_8x8_inference_flag	1
frame_cropping_flag	1
frame_crop_left_offset	0
frame_crop_right_offset	0
frame_crop_top_offset	0
frame_crop_bottom_offset	6
vui_parameters_present_flag	1
vui_parameters()	
rbsp_stop_one_bit	1
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

249. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs defining, in the encoder, parameter values in a picture parameter set for a picture, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

Filter | Hex

SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	0
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	3
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	-3
deblocking_filter_control_present_flag	1
constrained_intra_pred_flag	0
redundant_pic_cnt_present_flag	0
rbsp_stop_one_bit	1
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

250. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs defining, in the encoder, at least one picture parameter value in a slice header, the picture parameter value remaining unchanged at least in all slice headers of one picture, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info

4541 units total

Hex

Pos	NAL Type	TID	Size
126233	30: 1: non-IDR - Coded slice of a non-IDR picture	0	9040
142869	31: 1: non-IDR - Coded slice of a non-IDR picture	0	5170
148043	32: 1: non-IDR - Coded slice of a non-IDR picture	0	680
148727	33: 1: non-IDR - Coded slice of a non-IDR picture	0	14378
163109	34: 1: non-IDR - Coded slice of a non-IDR picture	0	7491
170604	35: 1: non-IDR - Coded slice of a non-IDR picture	0	5568
176176	36: 1: non-IDR - Coded slice of a non-IDR picture	0	1288
177468	37: 1: non-IDR - Coded slice of a non-IDR picture	0	17139
194611	38: 1: non-IDR - Coded slice of a non-IDR picture	0	9028

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

Slice #0 size in bits: 44544 SLICE_NONIDR

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	6
pic_parameter_set_id	0
frame_num	15
pic_order_cnt_lsb	62
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	0
num_ref_idx_l1_active_minus1	1
> **ref_pic_list_reordering()**	
cabac_init_idc	0
slice_qp_delta	-10
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	-1
slice_beta_offset_div2	-1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

251. For another example, on information and belief, Amazon performs a method of encoding video in a manner that is substantially the same as described above when Amazon Prime Video contents are encoded to produce a bitstream that indicates it can be decoded by an H.265-compliant decoder, such that it is covered by claim 1 of the ’818 Patent.

Screenshots of Amazon Prime Video indicating "Codec: hevc."

252. For another example, on information and belief, Amazon performs a method of encoding a video sequence in a manner that is covered by claim 1 of the '818 Patent for Amazon.com advertisements, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 217.

253. When encoding Amazon.com advertisements, on information and belief, Amazon performs encoding sequences of pictures into a bitstream, wherein parameters are defined in a parameter set and each picture comprises information of one or more slices, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info

22858 units total

Hex

Pos	NAL Type	TID	Size
583	0: 7: SPS - Sequence parameter set	0	41
627	1: 8: PPS - Picture parameter set	0	4
216040	2: 9: AUD - Access unit delimiter	0	2
216046	3: 6: SEI - Supplemental enhancement informat...	0	11
216061	4: 6: SEI - Supplemental enhancement informat...	0	13
216078	5: 6: SEI - Supplemental enhancement informat...	0	76
216158	6: 5: IDR - Coded slice of an IDR picture	0	167
216329	7: 9: AUD - Access unit delimiter	0	2
216335	8: 6: SEI - Supplemental enhancement informat...	0	12
216351	9: 1: non-IDR - Coded slice of a non-IDR picture	0	16455
233663	10: 9: AUD - Access unit delimiter	0	2
233669	11: 6: SEI - Supplemental enhancement informa...	0	11
233684	12: 1: non-IDR - Coded slice of a non-IDR picture	0	10395
244936	13: 9: AUD - Access unit delimiter	0	2
244942	14: 6: SEI - Supplemental enhancement informa	0	12

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info | 6332

Filter | Hex

SE Name	Value
profile_idc	100
constraint_set0_flag	0
constraint_set1_flag	0
constraint_set2_flag	0
constraint_set3_flag	0
constraint_set4_flag	0
constraint_set5_flag	0
reserved_zero_2bits	0
level_idc	30
seq_parameter_set_id	0
chroma_format_idc	1
bit_depth_luma_minus8	0
bit_depth_chroma_minus8	0
qpprime_y_zero_transform_bypass_flag	0
seq_scaling_matrix_present_flag	0
log2_max_frame_num_minus4	2
pic_order_cnt_type	0
log2_max_pic_order_cnt_lsb_minus4	3
max_num_ref_frames	4
gaps_in_frame_num_value_allowed_flag	0
pic_width_in_mbs_minus1	53
pic_height_in_map_units_minus1	29
frame_mbs_only_flag	1
direct_8x8_inference_flag	1
frame_cropping_flag	1
frame_crop_left_offset	0
frame_crop_right_offset	5
frame_crop_top_offset	0
frame_crop_bottom_offset	0
vui_parameters_present_flag	1
vui_parameters()	
rbsp_stop_one_bit	1
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info | 6332

Filter | Hex

SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	1
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	2
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	0
deblocking_filter_control_present_flag	1
constrained_intra_pred_flag	0
redundant_pic_cnt_present_flag	0
transform_8x8_mode_flag	1
pic_scaling_matrix_present_flag	0
second_chroma_qp_index_offset	0
rbsp_stop_one_bit	1
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info | 6332

Slice #0 size in bits: 52384 SLICE_NONIDR

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	6
pic_parameter_set_id	0
frame_num	25
pic_order_cnt_lsb	90
delta_pic_order_cnt_bottom	0
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	1
num_ref_idx_l1_active_minus1	0
ref_pic_list_reordering()	
cabac_init_idc	0
slice_qp_delta	-14
disable_deblocking_filter_idc	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

254. When encoding Amazon.com advertisements, on information and belief, Amazon performs defining, in an encoder, parameter values in a sequence parameter set for a sequence of pictures, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info

22858 units total

Hex

Pos	NAL Type	TID	Size
583	0: 7: SPS - Sequence parameter set	0	41
627	1: 8: PPS - Picture parameter set	0	4
216040	2: 9: AUD - Access unit delimiter	0	2
216046	3: 6: SEI - Supplemental enhancement informat...	0	11
216061	4: 6: SEI - Supplemental enhancement informat...	0	13
216078	5: 6: SEI - Supplemental enhancement informat...	0	76
216158	6: 5: IDR - Coded slice of an IDR picture	0	167
216329	7: 9: AUD - Access unit delimiter	0	2
216335	8: 6: SEI - Supplemental enhancement informat...	0	12
216351	9: 1: non-IDR - Coded slice of a non-IDR picture	0	16455
233663	10: 9: AUD - Access unit delimiter	0	2
233669	11: 6: SEI - Supplemental enhancement informa...	0	11
233684	12: 1: non-IDR - Coded slice of a non-IDR picture	0	10395
244936	13: 9: AUD - Access unit delimiter	0	2
244942	14: 6: SEI - Supplemental enhancement informa	0	12

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info | 6332

Filter | Hex

SE Name	Value
profile_idc	100
constraint_set0_flag	0
constraint_set1_flag	0
constraint_set2_flag	0
constraint_set3_flag	0
constraint_set4_flag	0
constraint_set5_flag	0
reserved_zero_2bits	0
level_idc	30
seq_parameter_set_id	0
chroma_format_idc	1
bit_depth_luma_minus8	0
bit_depth_chroma_minus8	0
qpprime_y_zero_transform_bypass_flag	0
seq_scaling_matrix_present_flag	0
log2_max_frame_num_minus4	2
pic_order_cnt_type	0
log2_max_pic_order_cnt_lsb_minus4	3
max_num_ref_frames	4
gaps_in_frame_num_value_allowed_flag	0
pic_width_in_mbs_minus1	53
pic_height_in_map_units_minus1	29
frame_mbs_only_flag	1
direct_8x8_inference_flag	1
frame_cropping_flag	1
frame_crop_left_offset	0
frame_crop_right_offset	5
frame_crop_top_offset	0
frame_crop_bottom_offset	0
vui_parameters_present_flag	1
vui_parameters()	
rbsp_stop_one_bit	1
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

255. When encoding Amazon.com advertisements, on information and belief, Amazon performs defining, in the encoder, parameter values in a picture parameter set for a picture, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info | 6332

22858 units total

Hex

Pos	NAL Type	TID	Size
583	0: 7: SPS - Sequence parameter set	0	41
627	1: 8: PPS - Picture parameter set	0	4
216040	2: 9: AUD - Access unit delimiter	0	2
216046	3: 6: SEI - Supplemental enhancement informat...	0	11
216061	4: 6: SEI - Supplemental enhancement informat...	0	13
216078	5: 6: SEI - Supplemental enhancement informat...	0	76
216158	6: 5: IDR - Coded slice of an IDR picture	0	167
216329	7: 9: AUD - Access unit delimiter	0	2
216335	8: 6: SEI - Supplemental enhancement informat...	0	12
216351	9: 1: non-IDR - Coded slice of a non-IDR picture	0	16455
233663	10: 9: AUD - Access unit delimiter	0	2
233669	11: 6: SEI - Supplemental enhancement informa...	0	11
233684	12: 1: non-IDR - Coded slice of a non-IDR picture	0	10395
244936	13: 9: AUD - Access unit delimiter	0	2

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info | 6332

Filter

Hex

SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	1
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	2
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	0
deblocking_filter_control_present_flag	1
constrained_intra_pred_flag	0
redundant_pic_cnt_present_flag	0
transform_8x8_mode_flag	1
pic_scaling_matrix_present_flag	0
second_chroma_qp_index_offset	0
rbsp_stop_one_bit	1
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

256. When encoding Amazon.com advertisements, on information and belief, Amazon performs defining, in the encoder, at least one picture parameter value in a slice header, the picture parameter value remaining unchanged at least in all slice headers of one picture, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info | 6332

22858 units total

Hex

Pos	NAL Type	TID	Size
9753418	1350: 6: SEI - Supplemental enhancement infor...	0	11
9753433	1351: 1: non-IDR - Coded slice of a non-IDR pic...	0	12504
9766844	1352: 9: AUD - Access unit delimiter	0	2
9766850	1353: 6: SEI - Supplemental enhancement infor...	0	12
9766866	1354: 1: non-IDR - Coded slice of a non-IDR pic...	0	6548
9774212	1355: 9: AUD - Access unit delimiter	0	2
9774218	1356: 6: SEI - Supplemental enhancement infor...	0	11
9774233	1357: 6: SEI - Supplemental enhancement infor...	0	11

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info | 6332

Slice #0 size in bits: 52384 SLICE_NONIDR

Filter

Hex

SE Name	Value
first_mb_in_slice	0
slice_type	6
pic_parameter_set_id	0
frame_num	25
pic_order_cnt_lsb	90
delta_pic_order_cnt_bottom	0
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	1
num_ref_idx_l1_active_minus1	0
ref_pic_list_reordering()	
cabac_init_idc	0
slice_qp_delta	-14
disable_deblocking_filter_idc	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

257. For another example, on information and belief, Amazon performs a method of encoding video in a manner that is substantially the same as described above when Amazon.com videos are encoded to produce a bitstream that indicates it can be decoded by an H.265-compliant decoder, such that it is covered by claim 1 of the ’818 Patent.

258. For another example, on information and belief, Amazon performs a method of encoding a video sequence in a manner that is covered by claim 1 of the ’818 Patent for Twitch.tv video, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 222.

259. When encoding Twitch.tv video, on information and belief, Amazon performs encoding sequences of pictures into a bitstream, wherein parameters are defined in a parameter set and each picture comprises information of one or more slices, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info

757 units total

Hex

Pos	NAL Type	TID	Size
975	0: 9: AUD - Access unit delimiter	0	3
981	1: 7: SPS - Sequence parameter set	0	38
1022	2: 8: PPS - Picture parameter set	0	6
1031	3: 6: SEI - Supplemental enhancement informat...	0	11
1045	4: 6: SEI - Supplemental enhancement informat...	0	8
1056	5: 7: SPS - Sequence parameter set	0	38
1097	6: 8: PPS - Picture parameter set	0	6
1106	7: 5: IDR - Coded slice of an IDR picture	0	40229
42214	8: 5: IDR - Coded slice of an IDR picture	0	34133
77094	9: 5: IDR - Coded slice of an IDR picture	0	28957
106682	10: 5: IDR - Coded slice of an IDR picture	0	54862
162766	11: 9: AUD - Access unit delimiter	0	3
162772	12: 6: SEI - Supplemental enhancement informa...	0	7
162782	13: 1: non-IDR - Coded slice of a non-IDR picture	0	3257
166114	14: 1: non-IDR - Coded slice of a non-IDR picture	0	2919
169100	15: 1: non-IDR - Coded slice of a non-IDR picture	0	3047
172218	16: 1: non-IDR - Coded slice of a non-IDR picture	0	3905
176984	17: 9: AUD - Access unit delimiter	0	3
176990	18: 6: SEI - Supplemental enhancement informa...	0	7
177000	19: 1: non-IDR - Coded slice of a non-IDR picture	0	1714
178753	20: 1: non-IDR - Coded slice of a non-IDR picture	0	1708

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info

Filter | Hex

SE Name	Value
profile_idc	100
constraint_set0_flag	0
constraint_set1_flag	0
constraint_set2_flag	0
constraint_set3_flag	0
constraint_set4_flag	0
constraint_set5_flag	0
reserved_zero_2bits	0
level_idc	42
seq_parameter_set_id	0
chroma_format_idc	1
bit_depth_luma_minus8	0
bit_depth_chroma_minus8	0
qpprime_y_zero_transform_bypass_flag	0
seq_scaling_matrix_present_flag	0
log2_max_frame_num_minus4	0
pic_order_cnt_type	0
log2_max_pic_order_cnt_lsb_minus4	1
max_num_ref_frames	5
gaps_in_frame_num_value_allowed_flag	0
pic_width_in_mbs_minus1	119
pic_height_in_map_units_minus1	67
frame_mbs_only_flag	1
direct_8x8_inference_flag	1
frame_cropping_flag	1
frame_crop_left_offset	0
frame_crop_right_offset	0
frame_crop_top_offset	0
frame_crop_bottom_offset	4
vui_parameters_present_flag	1
vui_parameters()	
rbsp_stop_one_bit	1
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info

Filter | Hex

SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	0
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	4
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	-3
deblocking_filter_control_present_flag	1
constrained_intra_pred_flag	0
redundant_pic_cnt_present_flag	0
transform_8x8_mode_flag	1
pic_scaling_matrix_present_flag	0
second_chroma_qp_index_offset	-3
rbsp_stop_one_bit	1

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info | 10232

Slice #1 size in bits: 13664 SLICE_NONIDR

Filter Hex

SE Name	Value
first_mb_in_slice	2040
slice_type	6
pic_parameter_set_id	0
frame_num	2
pic_order_cnt_lsb	2
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	0
num_ref_idx_l1_active_minus1	0
ref_pic_list_reordering()	
cabac_init_idc	0
slice_qp_delta	6
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	-1
slice_beta_offset_div2	-1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

260. When encoding Twitch.tv video, on information and belief, Amazon performs defining, in an encoder, parameter values in a sequence parameter set for a sequence of pictures, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info

757 units total

Hex

Pos	NAL Type	TID	Size
975	0: 9: AUD - Access unit delimiter	0	3
981	1: 7: SPS - Sequence parameter set	0	38
1022	2: 8: PPS - Picture parameter set	0	6
1031	3: 6: SEI - Supplemental enhancement informat...	0	11
1045	4: 6: SEI - Supplemental enhancement informat...	0	8
1056	5: 7: SPS - Sequence parameter set	0	38
1097	6: 8: PPS - Picture parameter set	0	6
1106	7: 5: IDR - Coded slice of an IDR picture	0	40229
42214	8: 5: IDR - Coded slice of an IDR picture	0	34133
77094	9: 5: IDR - Coded slice of an IDR picture	0	28957
106682	10: 5: IDR - Coded slice of an IDR picture	0	54862
162766	11: 9: AUD - Access unit delimiter	0	3
162772	12: 6: SEI - Supplemental enhancement informa...	0	7
162782	13: 1: non-IDR - Coded slice of a non-IDR picture	0	3257
166114	14: 1: non-IDR - Coded slice of a non-IDR picture	0	2919
169100	15: 1: non-IDR - Coded slice of a non-IDR picture	0	3047
172218	16: 1: non-IDR - Coded slice of a non-IDR picture	0	3905
176984	17: 9: AUD - Access unit delimiter	0	3
176990	18: 6: SEI - Supplemental enhancement informa...	0	7
177000	19: 1: non-IDR - Coded slice of a non-IDR picture	0	1714
178753	20: 1: non-IDR - Coded slice of a non-IDR picture	0	1708

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info

Filter | Hex

SE Name	Value
profile_idc	100
constraint_set0_flag	0
constraint_set1_flag	0
constraint_set2_flag	0
constraint_set3_flag	0
constraint_set4_flag	0
constraint_set5_flag	0
reserved_zero_2bits	0
level_idc	42
seq_parameter_set_id	0
chroma_format_idc	1
bit_depth_luma_minus8	0
bit_depth_chroma_minus8	0
qpprime_y_zero_transform_bypass_flag	0
seq_scaling_matrix_present_flag	0
log2_max_frame_num_minus4	0
pic_order_cnt_type	0
log2_max_pic_order_cnt_lsb_minus4	1
max_num_ref_frames	5
gaps_in_frame_num_value_allowed_flag	0
pic_width_in_mbs_minus1	119
pic_height_in_map_units_minus1	67
frame_mbs_only_flag	1
direct_8x8_inference_flag	1
frame_cropping_flag	1
frame_crop_left_offset	0
frame_crop_right_offset	0
frame_crop_top_offset	0
frame_crop_bottom_offset	4
vui_parameters_present_flag	1
vui_parameters()	
rbsp_stop_one_bit	1
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

261. When encoding Twitch.tv video, on information and belief, Amazon performs defining, in the encoder, parameter values in a picture parameter set for a picture, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info

757 units total

Hex

Pos	NAL Type	TID	Size
975	0: 9: AUD - Access unit delimiter	0	3
981	1: 7: SPS - Sequence parameter set	0	38
1022	2: 8: PPS - Picture parameter set	0	6
1031	3: 6: SEI - Supplemental enhancement informat...	0	11
1045	4: 6: SEI - Supplemental enhancement informat...	0	8
1056	5: 7: SPS - Sequence parameter set	0	38
1097	6: 8: PPS - Picture parameter set	0	6
1106	7: 5: IDR - Coded slice of an IDR picture	0	40229
42214	8: 5: IDR - Coded slice of an IDR picture	0	34133
77094	9: 5: IDR - Coded slice of an IDR picture	0	28957
106682	10: 5: IDR - Coded slice of an IDR picture	0	54862
162766	11: 9: AUD - Access unit delimiter	0	3
162772	12: 6: SEI - Supplemental enhancement informa...	0	7
162782	13: 1: non-IDR - Coded slice of a non-IDR picture	0	3257
166114	14: 1: non-IDR - Coded slice of a non-IDR picture	0	2919
169100	15: 1: non-IDR - Coded slice of a non-IDR picture	0	3047
172218	16: 1: non-IDR - Coded slice of a non-IDR picture	0	3905
176984	17: 9: AUD - Access unit delimiter	0	3
176990	18: 6: SEI - Supplemental enhancement informa...	0	7
177000	19: 1: non-IDR - Coded slice of a non-IDR picture	0	1714
178753	20: 1: non-IDR - Coded slice of a non-IDR picture	0	1708

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info

Filter | Hex

SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	0
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	4
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	-3
deblocking_filter_control_present_flag	1
constrained_intra_pred_flag	0
redundant_pic_cnt_present_flag	0
transform_8x8_mode_flag	1
pic_scaling_matrix_present_flag	0
second_chroma_qp_index_offset	-3
rbsp_stop_one_bit	1

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

262. When encoding Twitch.tv video, on information and belief, Amazon performs defining, in the encoder, at least one picture parameter value in a slice header, the picture parameter value remaining unchanged at least in all slice headers of one picture, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info

757 units total | Hex

Pos	NAL Type	TID	Size
166114	14: 1: non-IDR - Coded slice of a non-IDR picture	0	2919
169100	15: 1: non-IDR - Coded slice of a non-IDR picture	0	3047
172218	16: 1: non-IDR - Coded slice of a non-IDR picture	0	3905
176984	17: 9: AUD - Access unit delimiter	0	3
176990	18: 6: SEI - Supplemental enhancement informa...	0	7
177000	19: 1: non-IDR - Coded slice of a non-IDR picture	0	1714
178753	20: 1: non-IDR - Coded slice of a non-IDR picture	0	1708

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info | 10232

Slice #0 size in bits: 13712 SLICE_NONIDR

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	6
pic_parameter_set_id	0
frame_num	2
pic_order_cnt_lsb	2
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	0
num_ref_idx_l1_active_minus1	0
ref_pic_list_reordering()	
cabac_init_idc	0
slice_qp_delta	6
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	-1
slice_beta_offset_div2	-1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

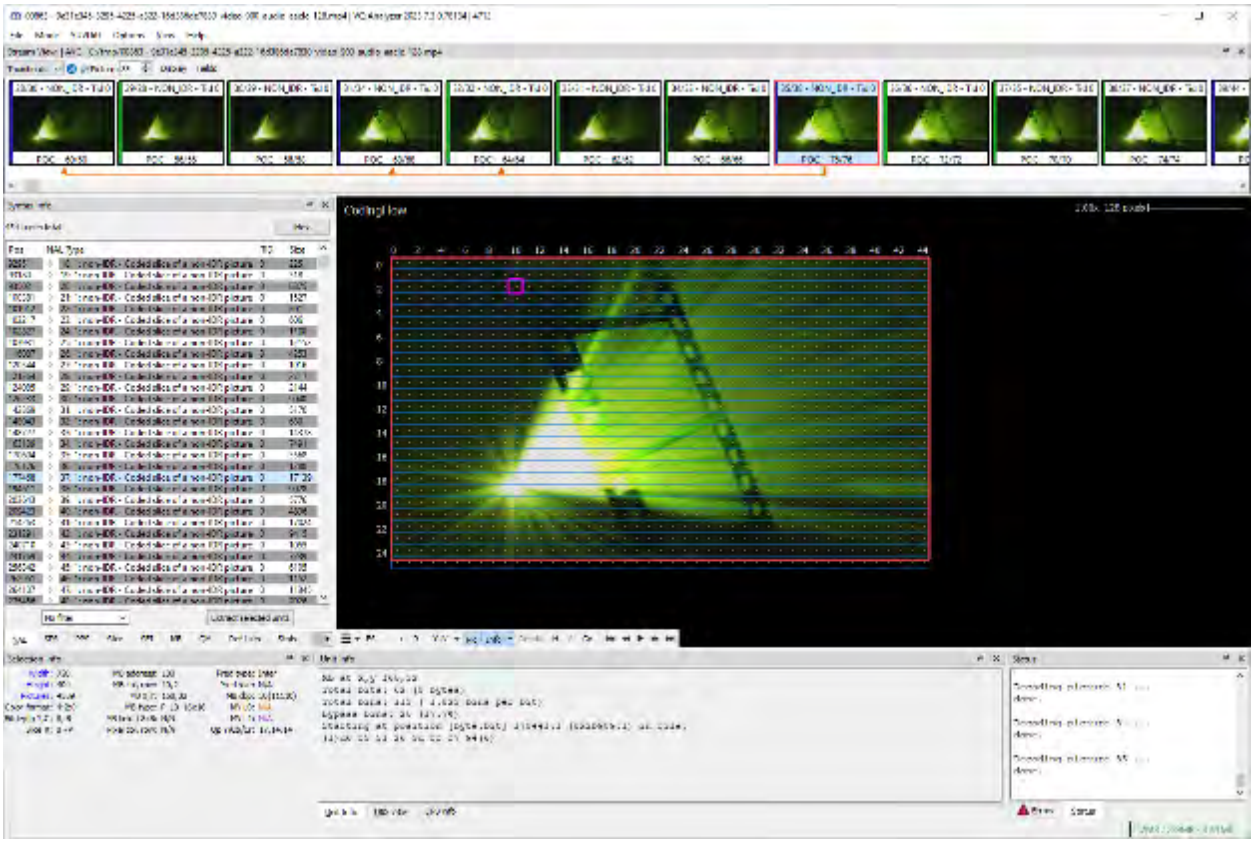
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

D. Amazon Makes, Imports, Uses, Sells, and/or Offers for Sale Products and Services that Infringe the ’469 Patent

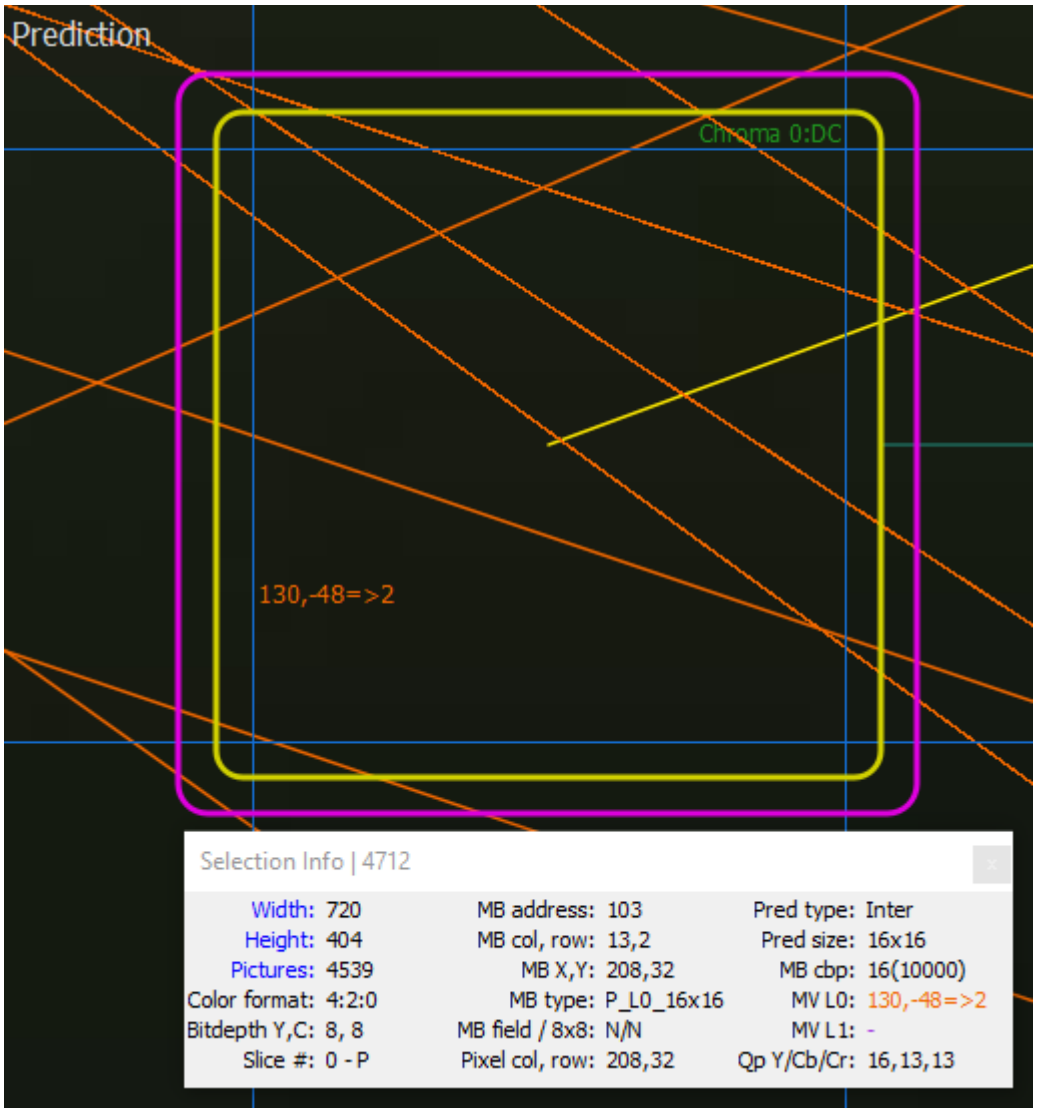
263. The Accused Products infringe one or more claims of the ’469 Patent, including, for example, claim 1.

264. As just one example of infringement, on information and belief, Amazon performs a method of interpolation in video coding in a manner that is covered by claim 1 of the ’469 Patent for Amazon Prime Video content (such as trailers), as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 212.

265. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs interpolation in video coding in which an image comprising pixels arranged in rows and columns and represented by values having a specified dynamic range, the pixels in the rows residing at unit horizontal locations and the pixels in the columns residing at unit vertical locations, is interpolated to generate values for sub-pixels at fractional horizontal and vertical locations, the fractional horizontal and vertical locations being defined according to $½^x$, where x is a positive integer having a maximum value N, as demonstrated in the screenshots below using VQ Analyzer software.

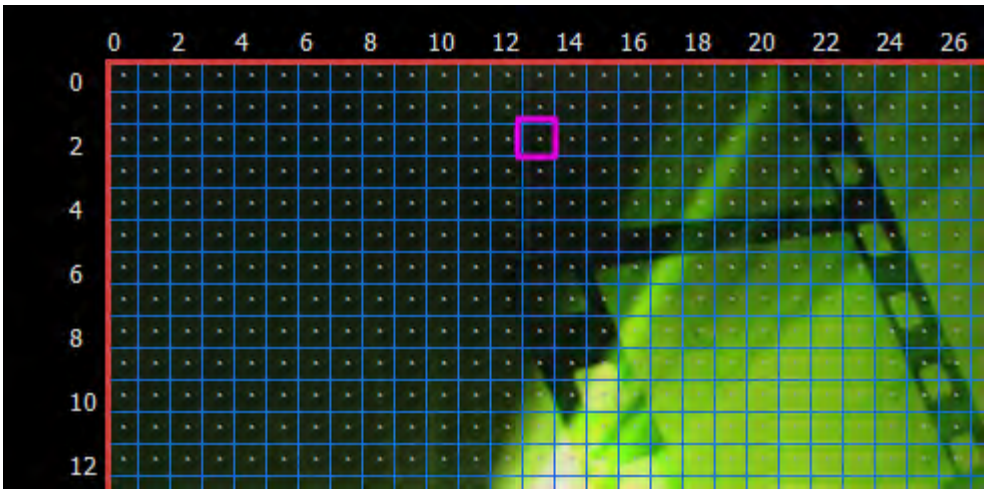


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

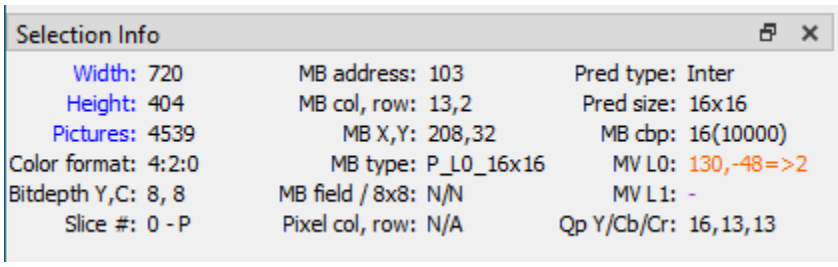


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

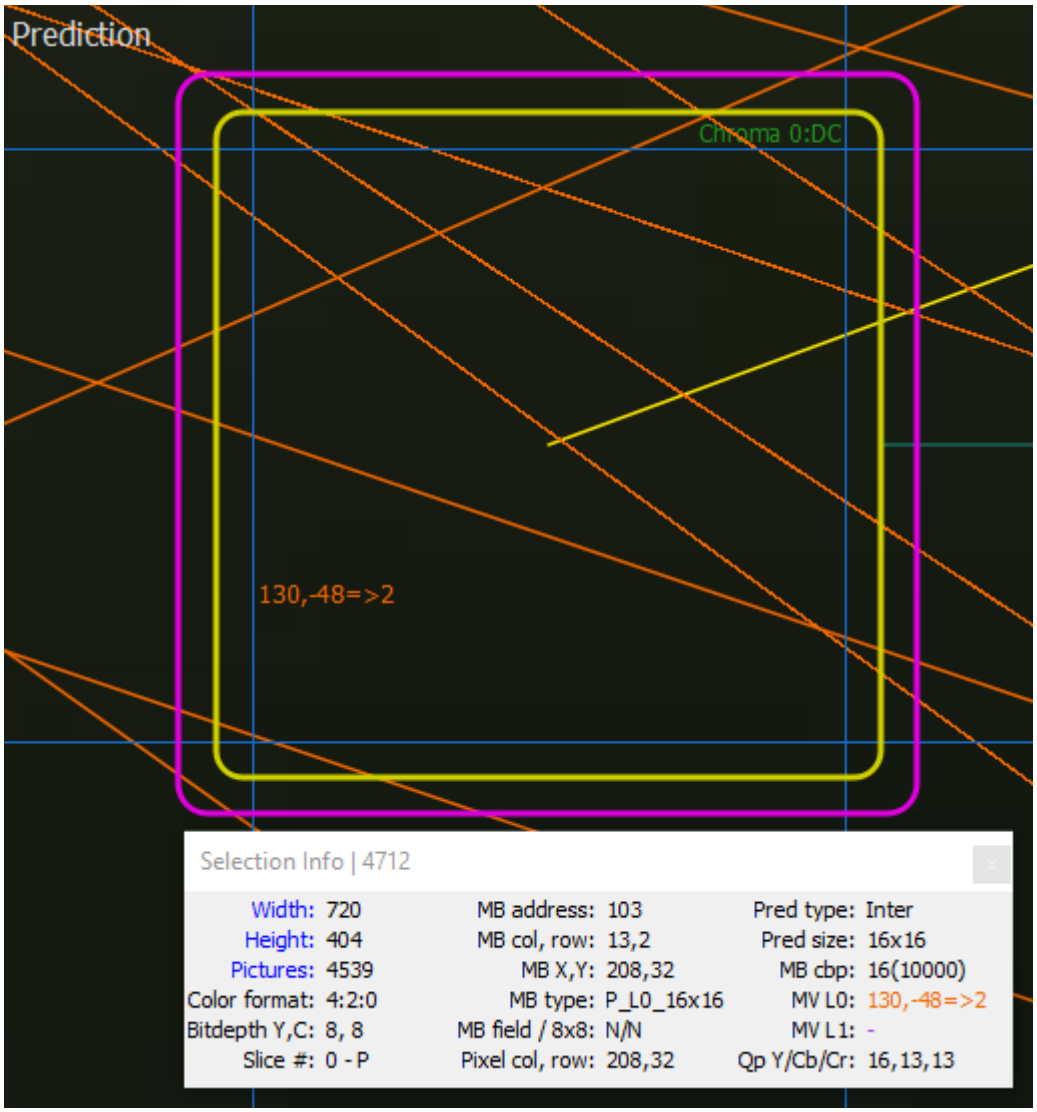
266. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs when values for sub-pixels at $½^{N-1}$ unit horizontal and unit vertical locations, and unit horizontal and $½^{N-1}$ unit vertical locations are required, interpolating such values directly using weighted sums of pixels residing at unit horizontal and unit vertical locations, as demonstrated in the screenshots below using VQ Analyzer software, where the 130, -48 motion vector corresponds to $½^{N-1}$ unit horizontal and unit vertical locations.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

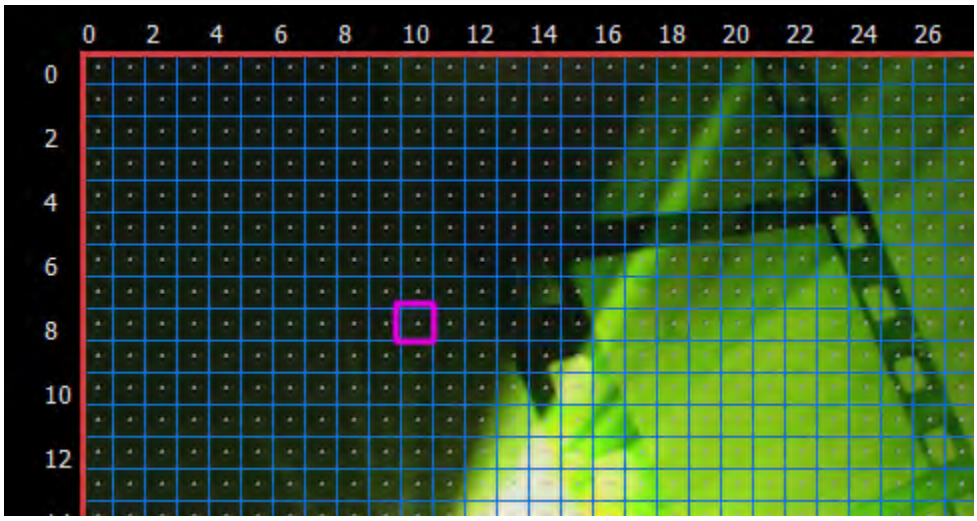


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

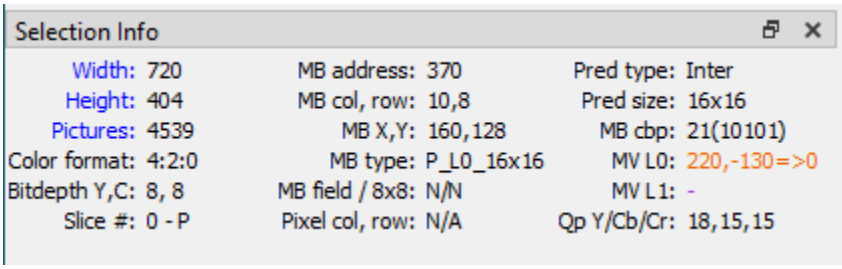


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

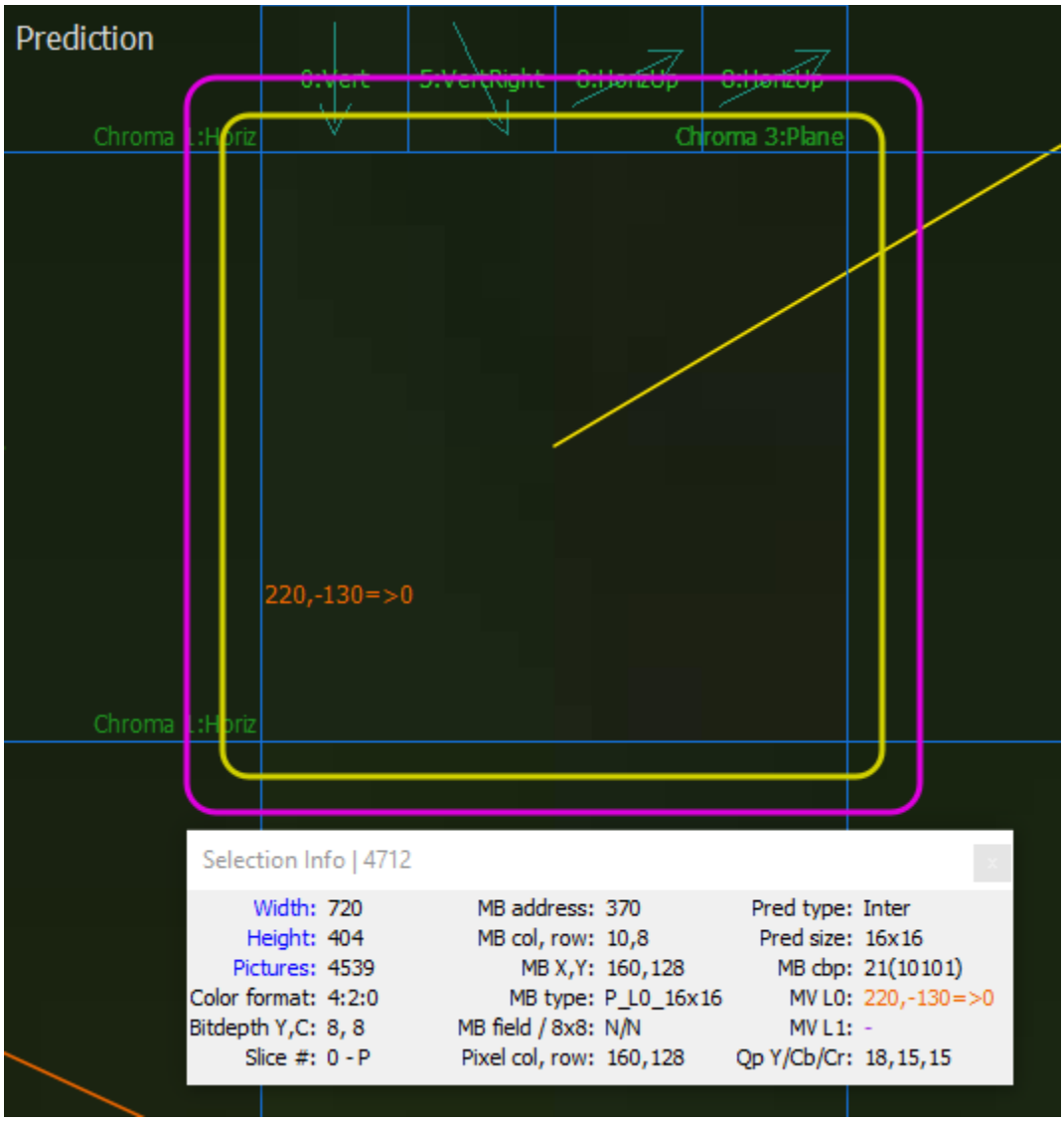
267. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs when values for sub-pixels at $½^{N-1}$ unit horizontal and unit vertical locations, and unit horizontal and $½^{N-1}$ unit vertical locations are required, interpolating such values directly using weighted sums of pixels residing at unit horizontal and unit vertical locations, as demonstrated in the screenshots below using VQ Analyzer software, where the 220, -130 motion vector corresponds to unit horizontal and $½^{N-1}$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



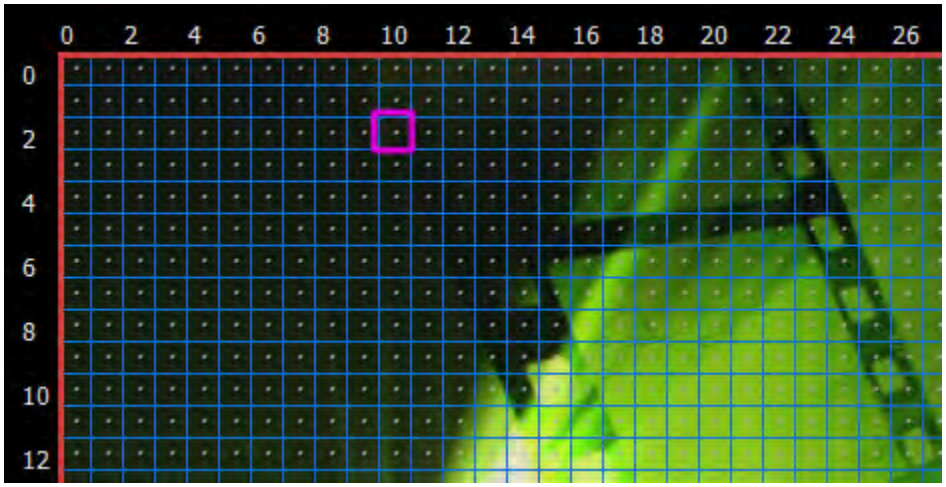
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



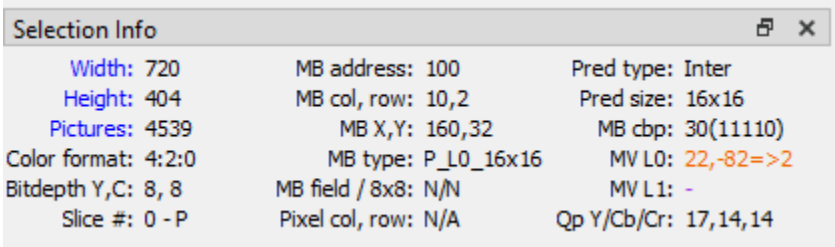
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

268. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs when values for sub-pixels at $½^{N-1}$ unit horizontal and $½^{N-1}$ unit vertical locations are required, interpolating such values directly using a choice of a first weighted sum of values for sub-pixels residing at $½^{N-1}$ unit horizontal and unit vertical locations and a second weighted sum of values for sub-pixels residing at unit horizontal and $½^{N-1}$ unit vertical locations, the first and second weighted sums of values being calculated according to step (a), as demonstrated in the

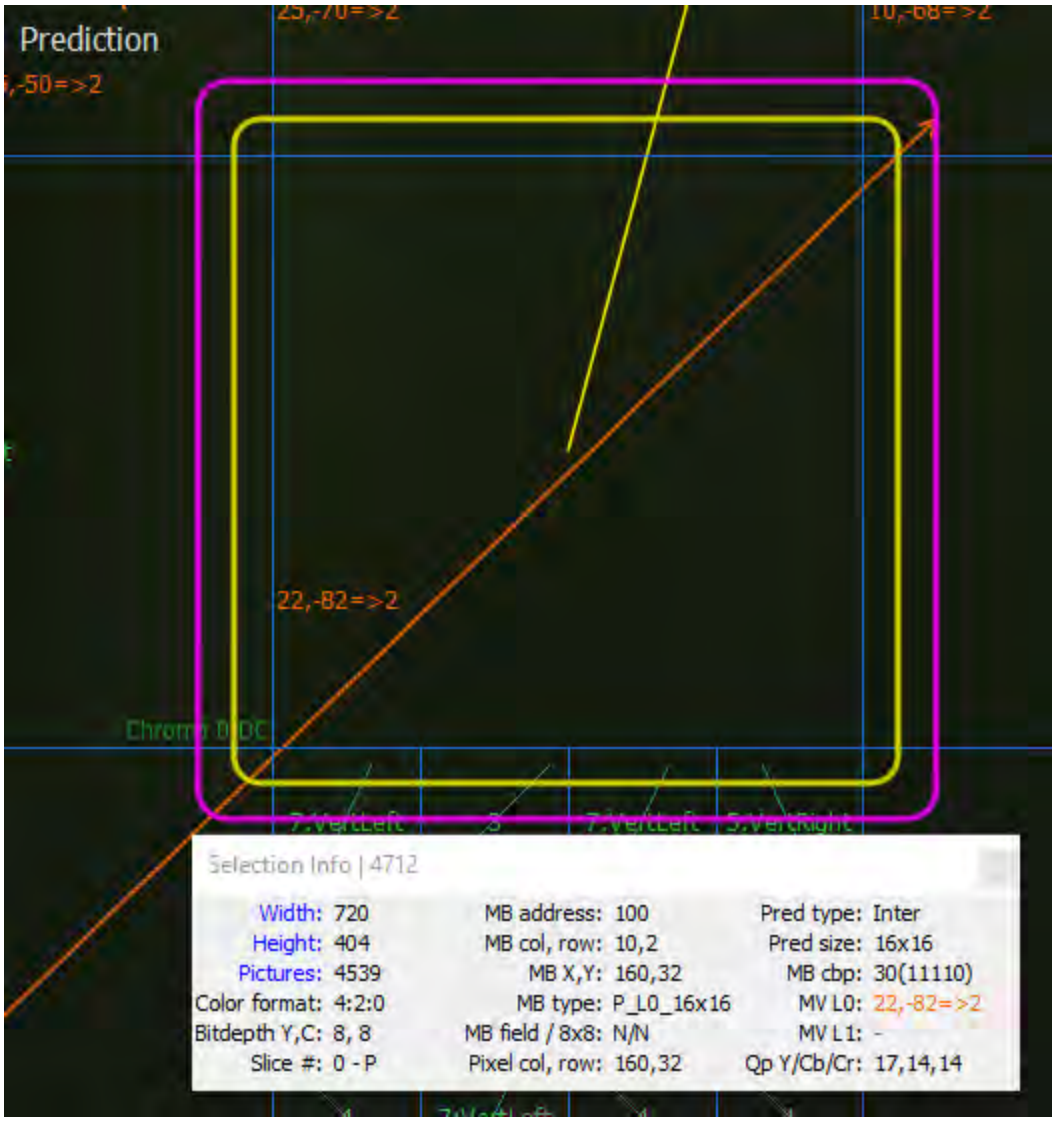
screenshots below using VQ Analyzer software, where the 22, -82 motion vector corresponds to $½^{N-1}$ unit horizontal and $½^{N-1}$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



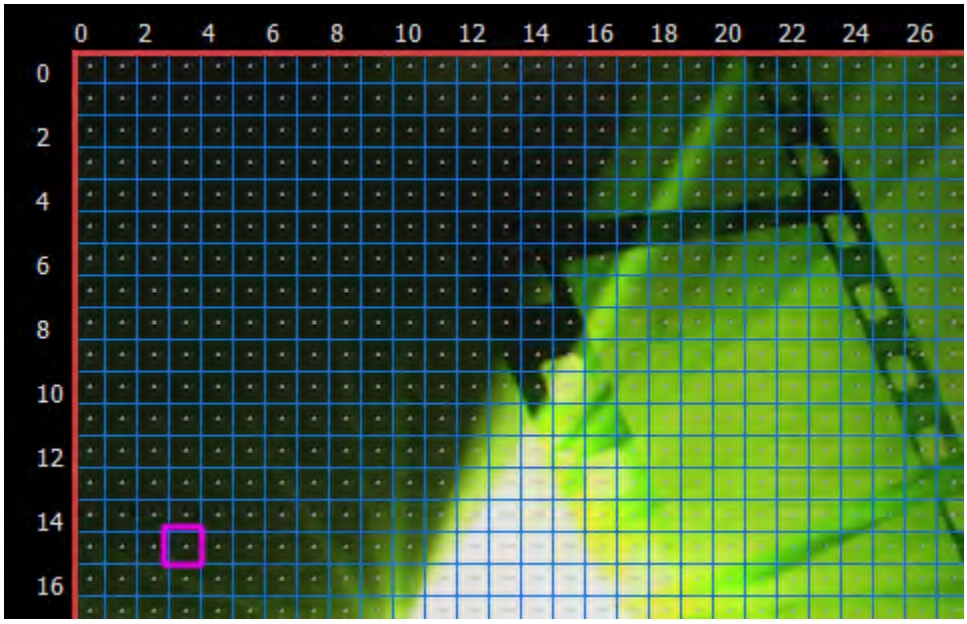
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



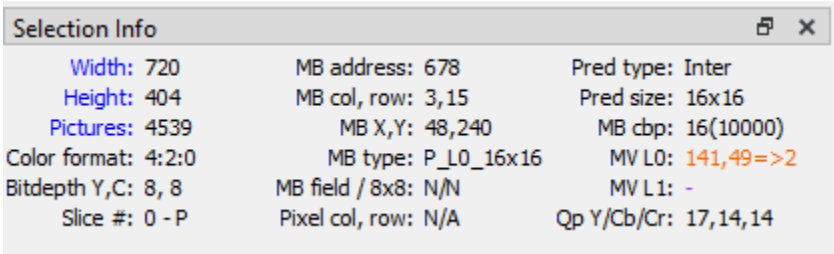
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

269. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs when a value for a sub-pixel situated at a $½^N$ unit horizontal and $½^N$ unit vertical location is required, interpolating such a value by taking a weighted average of the value of a first sub-pixel or pixel situated at a $½^{N-m}$ unit horizontal and $½^{N-n}$ unit vertical location and the value of a second sub-pixel or pixel located at a $½^{N-p}$ unit horizontal and $½^{N-q}$ unit vertical location, variables m, n, p and q taking integer values in the range 1 to N such that the first and second sub-pixels or pixels are located diagonally with respect to the sub-pixel at $½^N$ unit horizontal and $½^N$

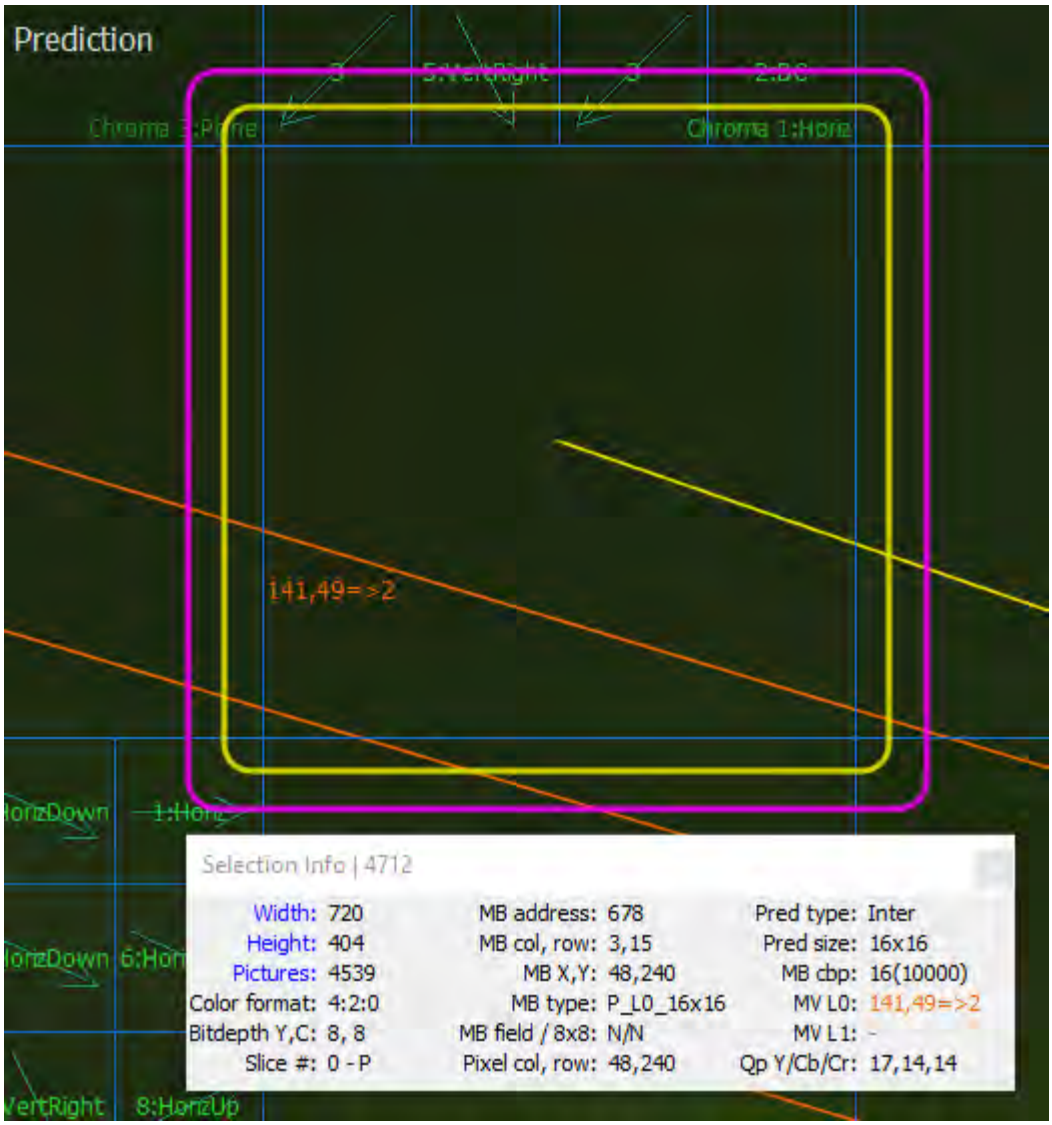
vertical location, as demonstrated in the screenshots below using VQ Analyzer software, where the 141, 49 motion vector corresponds to $½^N$ unit horizontal and $½^N$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

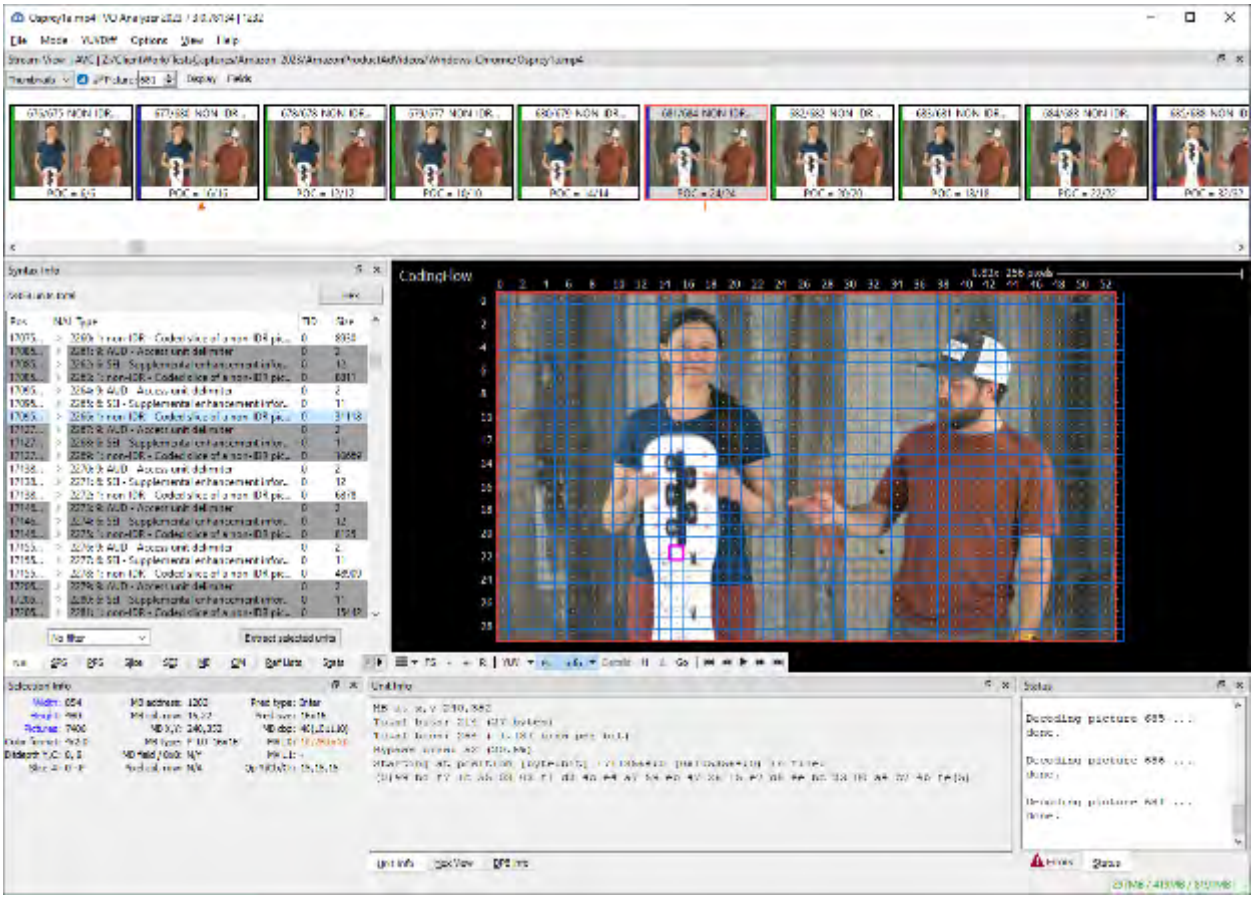


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

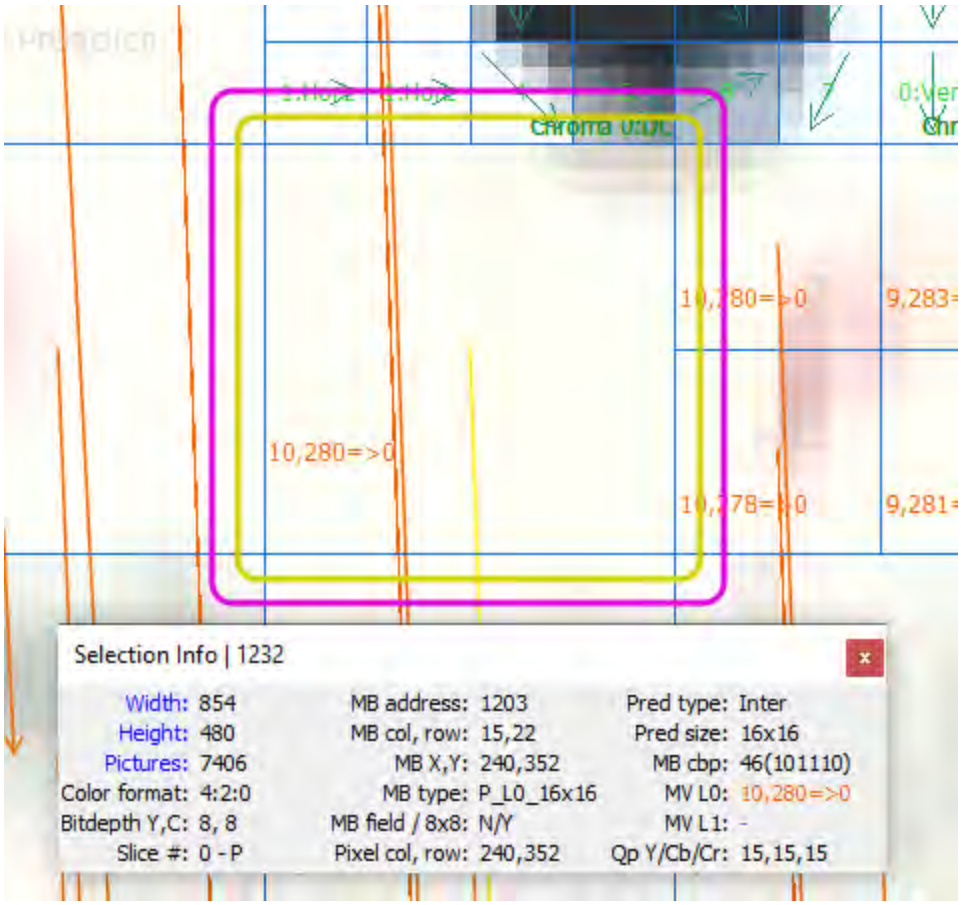
270. For another example, on information and belief, Amazon performs a method of interpolation in video coding in a manner that is covered by claim 1 of the ’469 Patent for Amazon.com advertisements, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 217.

271. When encoding Amazon.com advertisements, on information and belief, Amazon performs interpolation in video coding in which an image comprising pixels arranged in rows and

columns and represented by values having a specified dynamic range, the pixels in the rows residing at unit horizontal locations and the pixels in the columns residing at unit vertical locations, is interpolated to generate values for sub-pixels at fractional horizontal and vertical locations, the fractional horizontal and vertical locations being defined according to $½^x$, where x is a positive integer having a maximum value N, as demonstrated in the screenshots below using VQ Analyzer software.

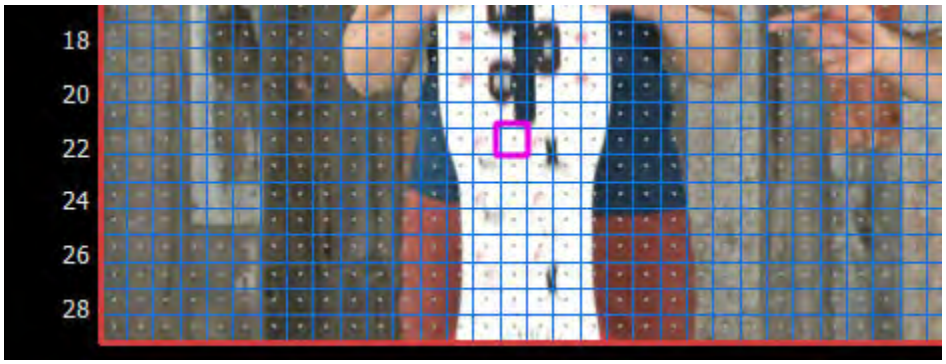


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

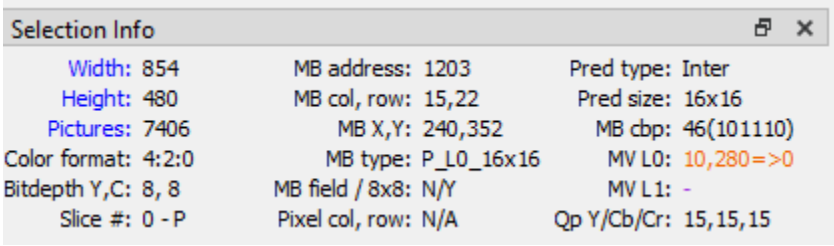


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

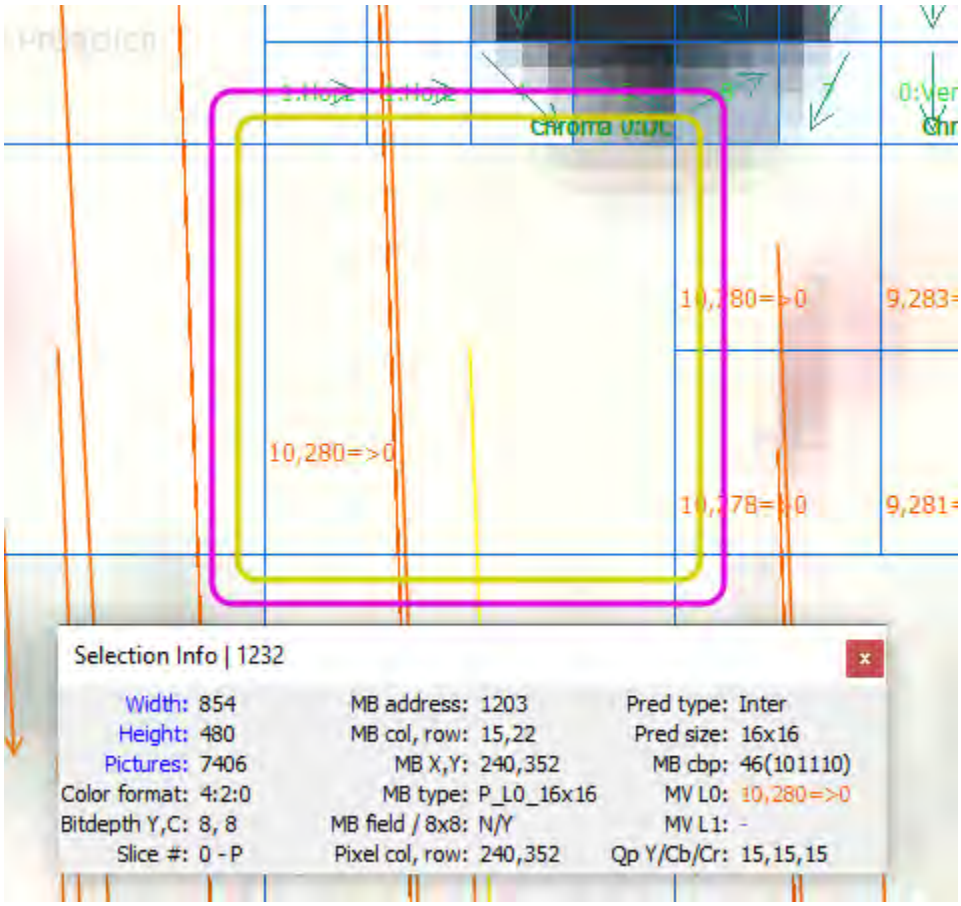
272. When encoding Amazon.com advertisements, on information and belief, Amazon performs when values for sub-pixels at $½^{N-1}$ unit horizontal and unit vertical locations, and unit horizontal and $½^{N-1}$ unit vertical locations are required, interpolating such values directly using weighted sums of pixels residing at unit horizontal and unit vertical locations, as demonstrated in the screenshots below using VQ Analyzer software, where the 10, 280 motion vector corresponds to $½^{N-1}$ unit horizontal and unit vertical locations.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

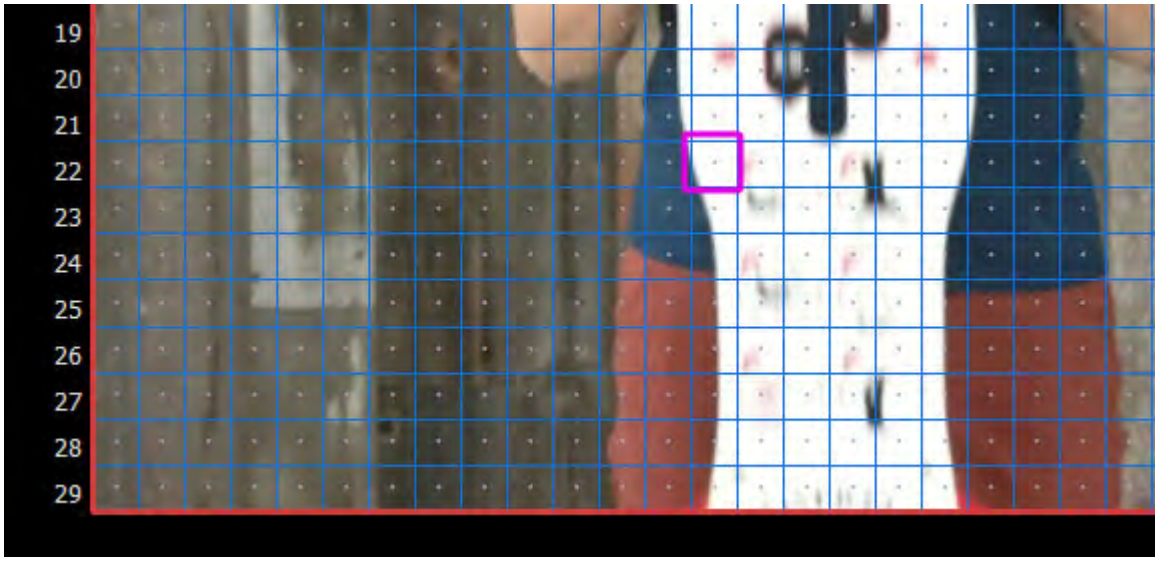


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

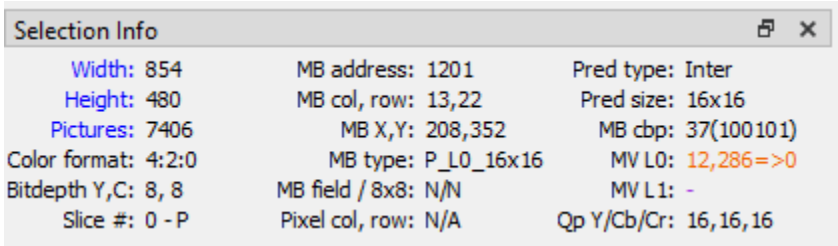


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

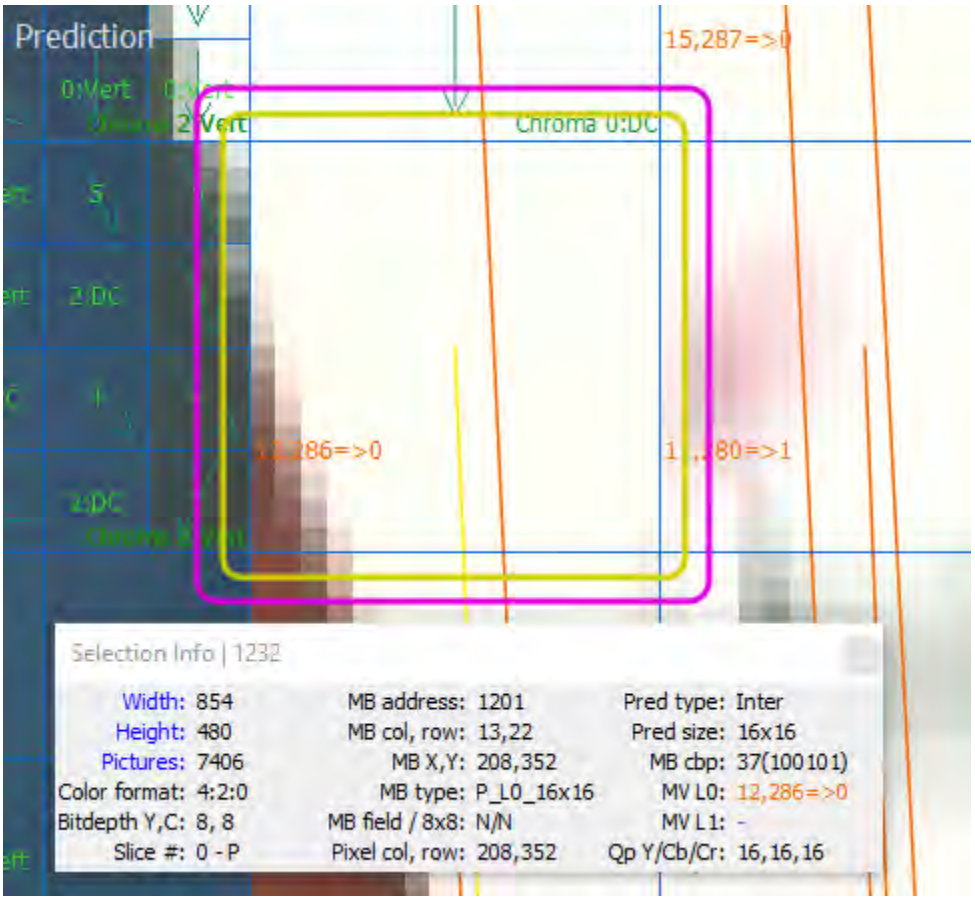
273. When encoding Amazon.com advertisements, on information and belief, Amazon performs when values for sub-pixels at ½N−1 unit horizontal and unit vertical locations, and unit horizontal and ½N−1 unit vertical locations are required, interpolating such values directly using weighted sums of pixels residing at unit horizontal and unit vertical locations, as demonstrated in the screenshots below using VQ Analyzer software, where the 12, 286 motion vector corresponds to unit horizontal and $½^{N-1}$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

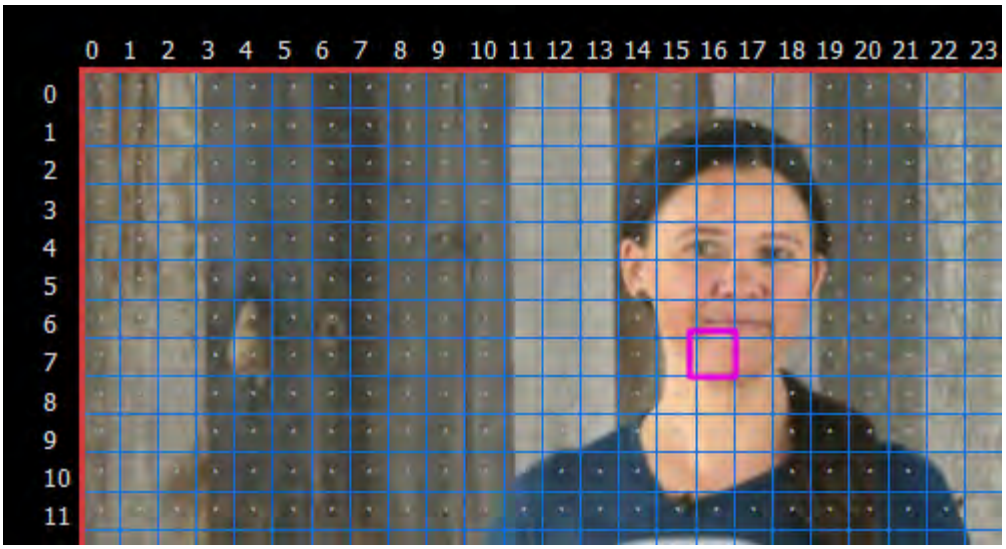


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

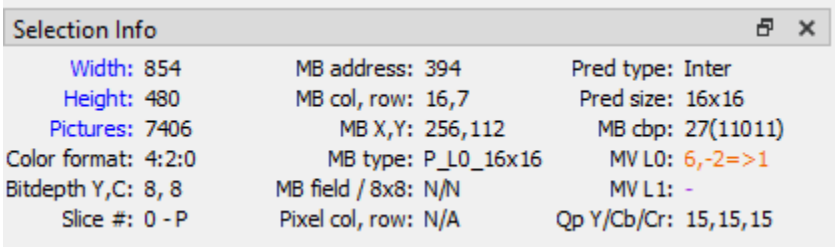


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

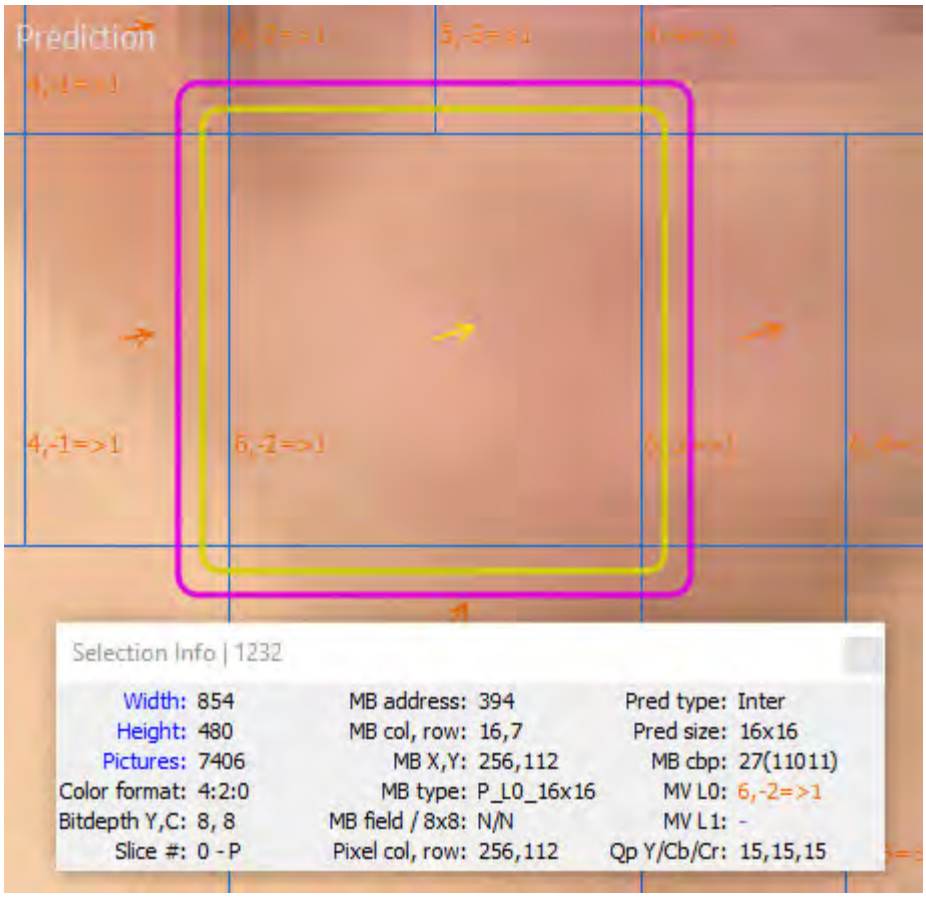
274. When encoding Amazon.com advertisements, on information and belief, Amazon performs when values for sub-pixels at $½^{N-1}$ unit horizontal and $½^{N-1}$ unit vertical locations are required, interpolating such values directly using a choice of a first weighted sum of values for sub-pixels residing at $½^{N-1}$ unit horizontal and unit vertical locations and a second weighted sum of values for sub-pixels residing at unit horizontal and $½^{N-1}$ unit vertical locations, the first and second weighted sums of values being calculated according to step (a), as demonstrated in the screenshots below using VQ Analyzer software, where the 6, -2 motion vector corresponds to $½^{N-1}$ unit horizontal and $½^{N-1}$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

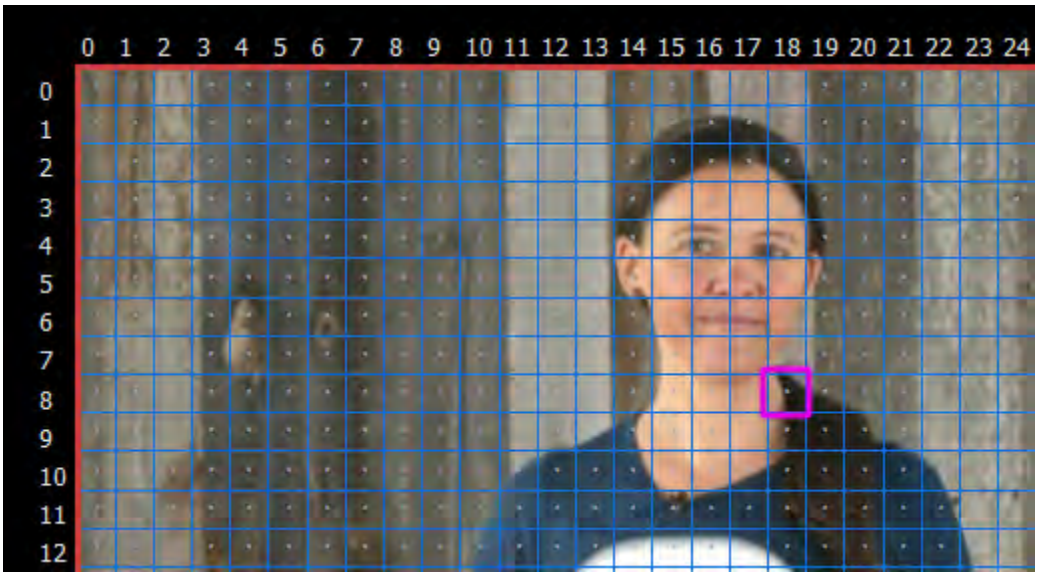


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

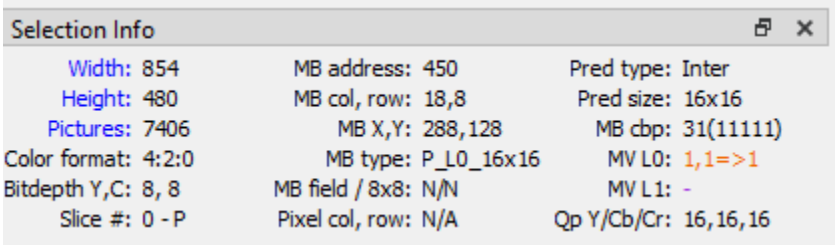


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

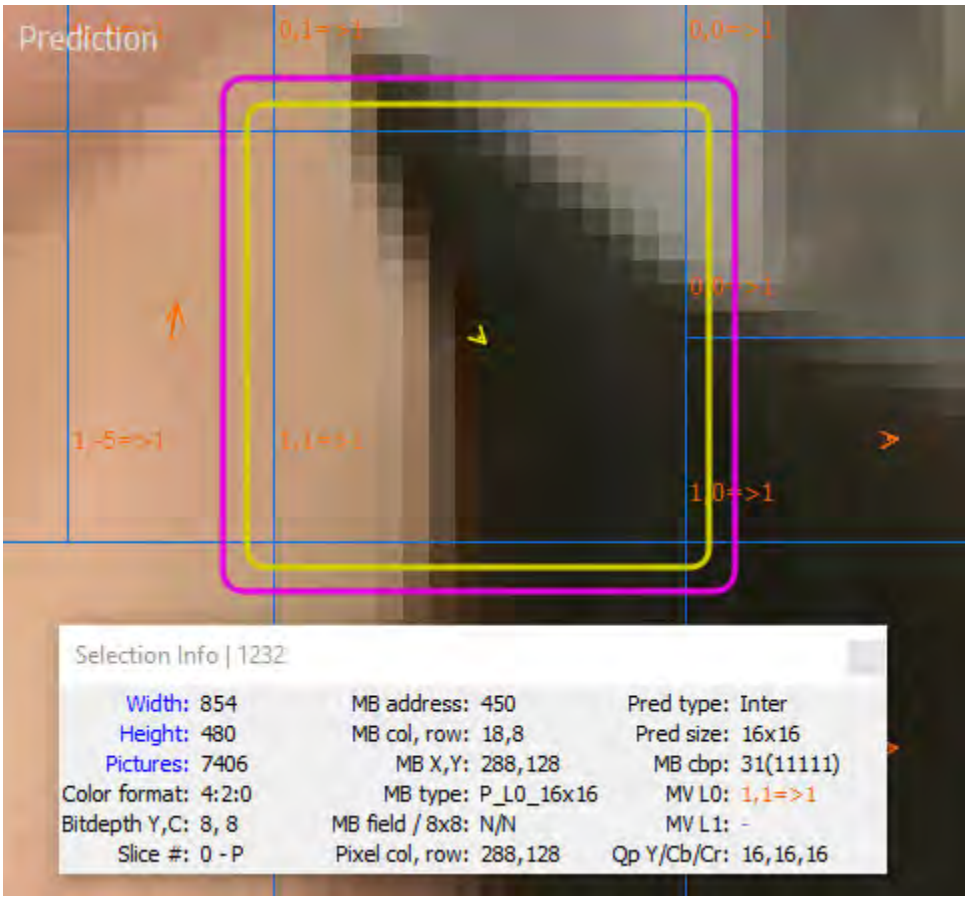
275. When encoding Amazon.com advertisements, on information and belief, Amazon performs when a value for a sub-pixel situated at a $½^N$ unit horizontal and $½^N$ unit vertical location is required, interpolating such a value by taking a weighted average of the value of a first sub-pixel or pixel situated at a $½^{N-m}$ unit horizontal and $½^{N-n}$ unit vertical location and the value of a second sub-pixel or pixel located at a $½^{N-p}$ unit horizontal and $½^{N-q}$ unit vertical location, variables m, n, p and q taking integer values in the range 1 to N such that the first and second sub-pixels or pixels are located diagonally with respect to the sub-pixel at $½^N$ unit horizontal and $½^N$ vertical locations, as demonstrated in the screenshots below using VQ Analyzer software, where the 1, 1 motion vector corresponds to $½^N$ unit horizontal and $½^N$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

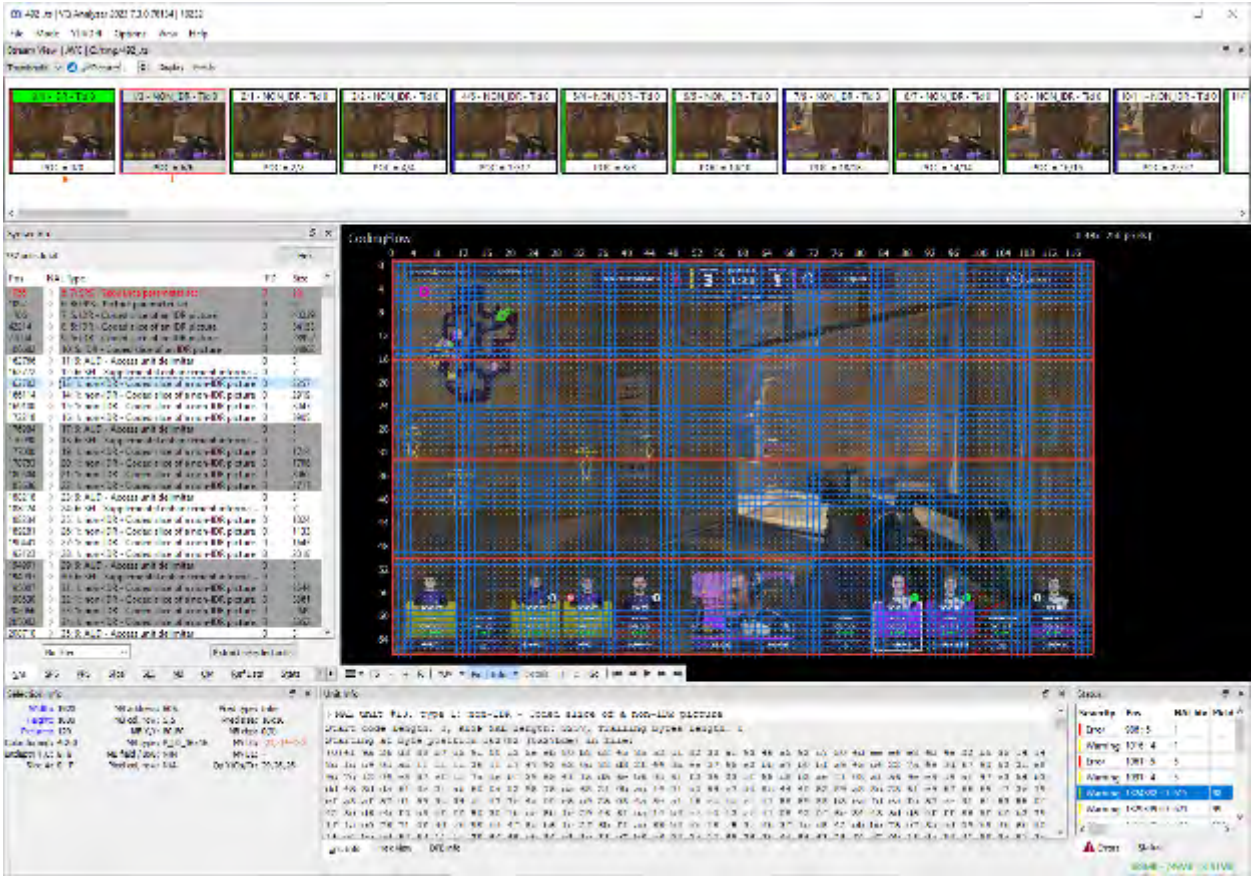


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

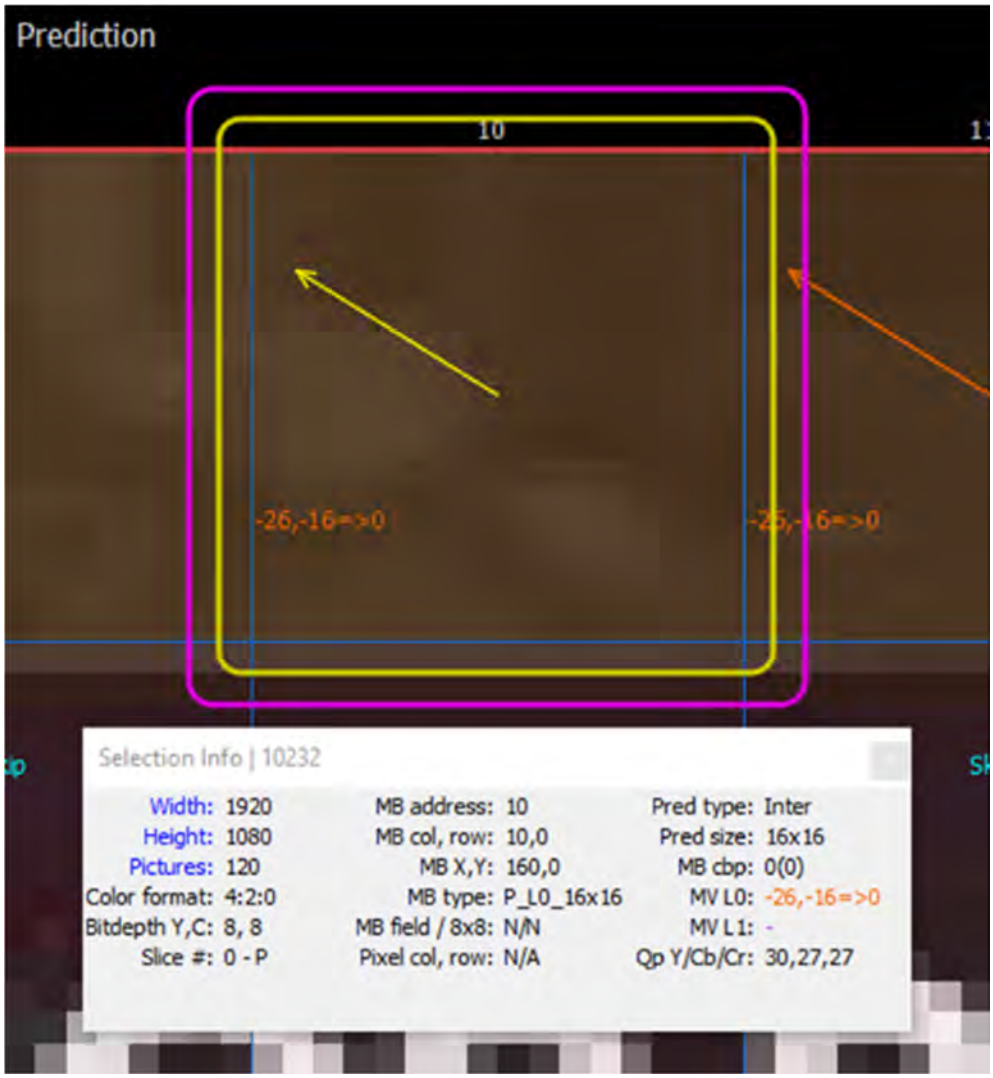
276. For another example, on information and belief, Amazon performs a method of interpolation in video coding in a manner that is covered by claim 1 of the ’469 Patent for Twitch.tv video, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 222.

277. When encoding Twitch.tv video, on information and belief, Amazon performs interpolation in video coding in which an image comprising pixels arranged in rows and columns and represented by values having a specified dynamic range, the pixels in the rows residing at unit horizontal locations and the pixels in the columns residing at unit vertical locations, is interpolated to generate values for sub-pixels at fractional horizontal and vertical locations, the fractional horizontal and vertical locations being defined according to $½^x$, where x is a positive

integer having a maximum value N, as demonstrated in the screenshots below using VQ Analyzer software.

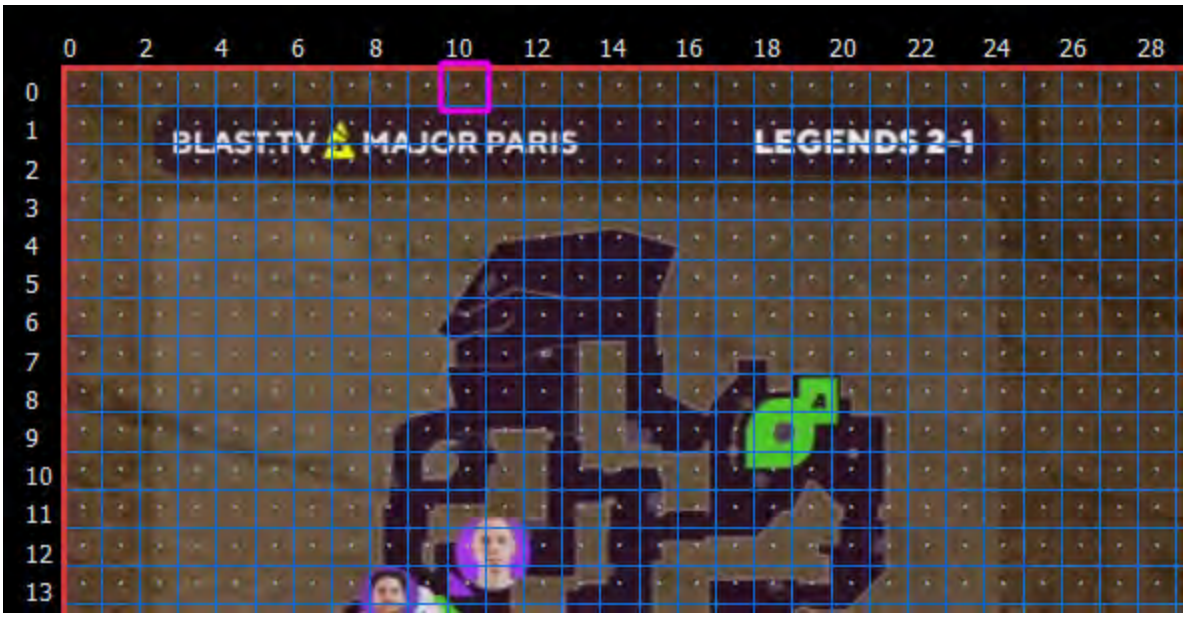


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

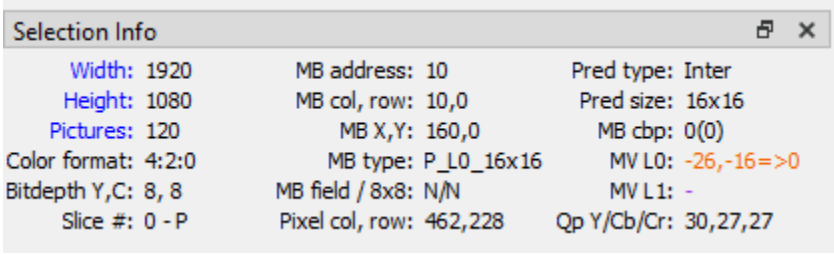


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

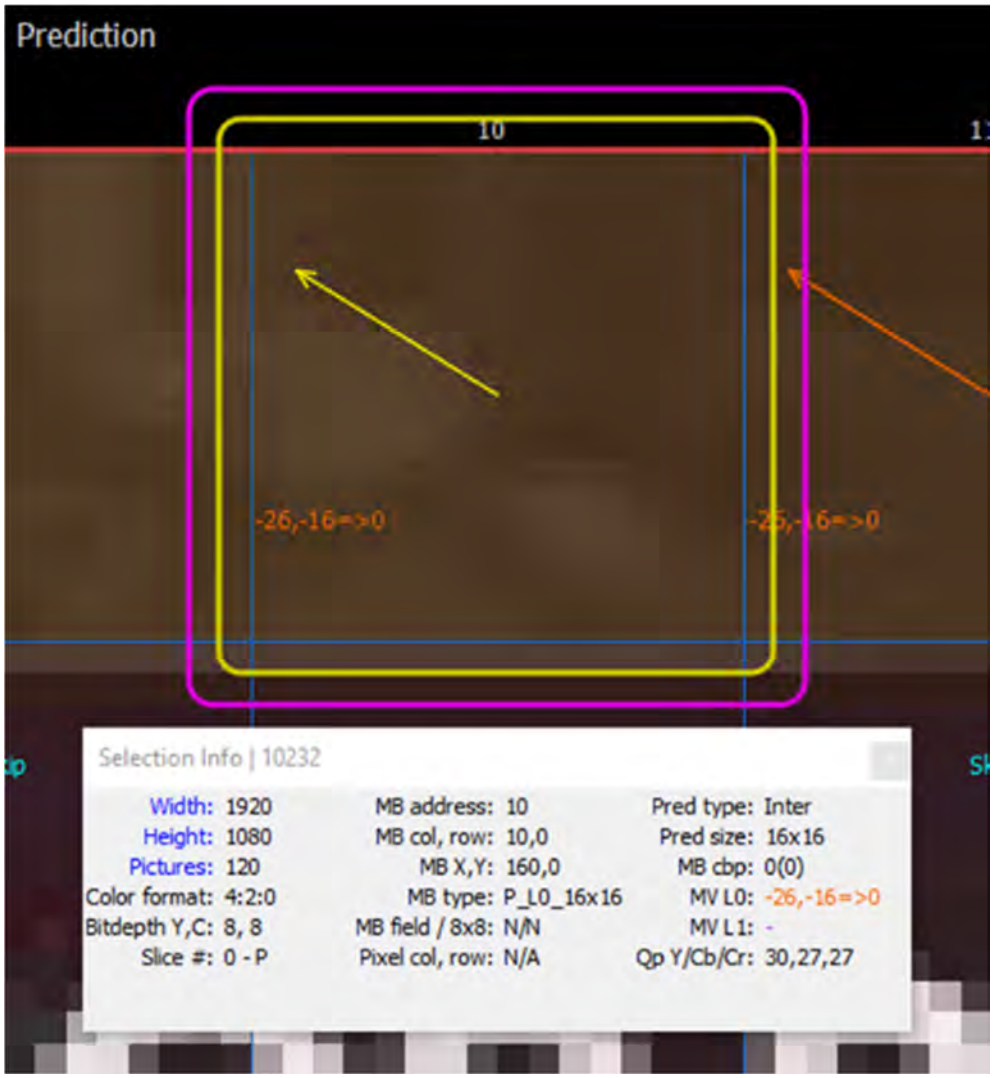
278. When encoding Twitch.tv video, on information and belief, Amazon performs when values for sub-pixels at $½^{N-1}$ unit horizontal and unit vertical locations, and unit horizontal and $½^{N-1}$ unit vertical locations are required, interpolating such values directly using weighted sums of pixels residing at unit horizontal and unit vertical locations, as demonstrated in the screenshots below using VQ Analyzer software, where the -26, -16 motion vector corresponds to $½^{N-1}$ unit horizontal and unit vertical locations.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

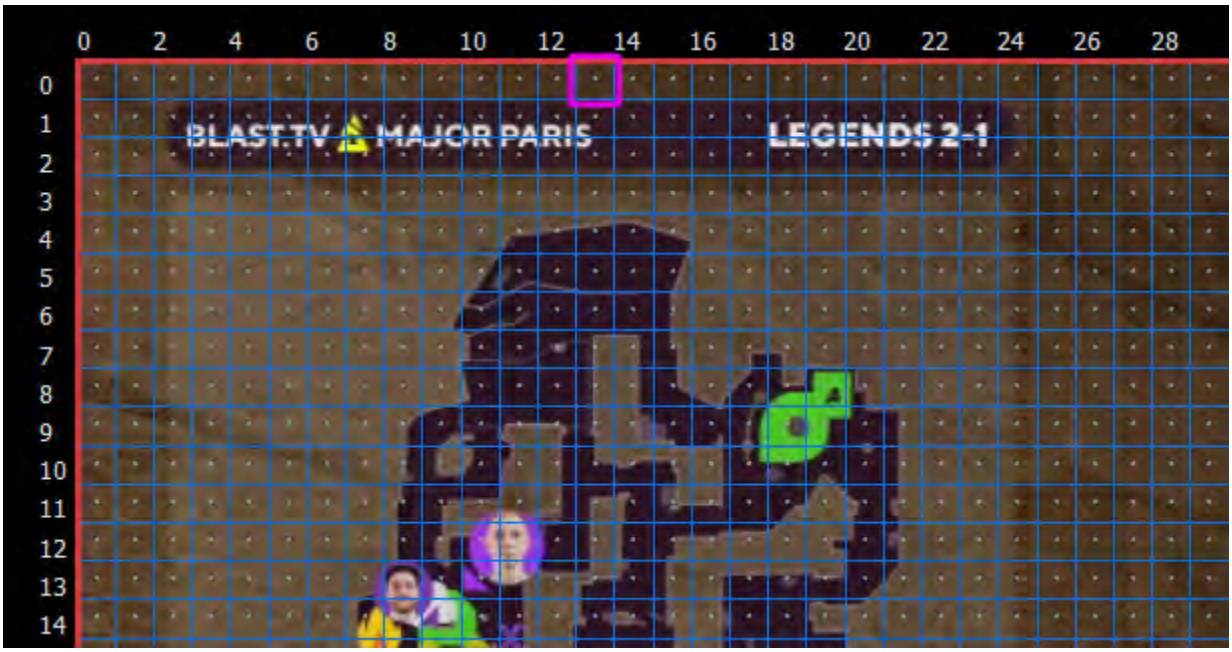


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

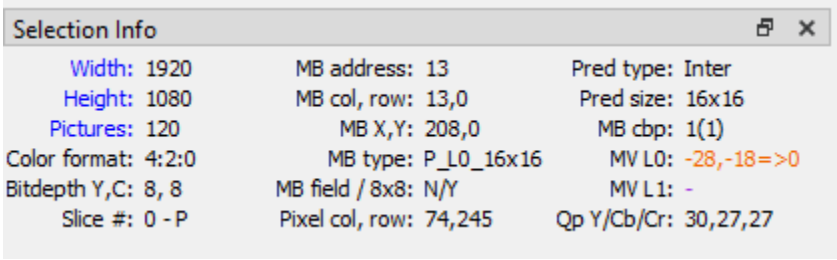


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

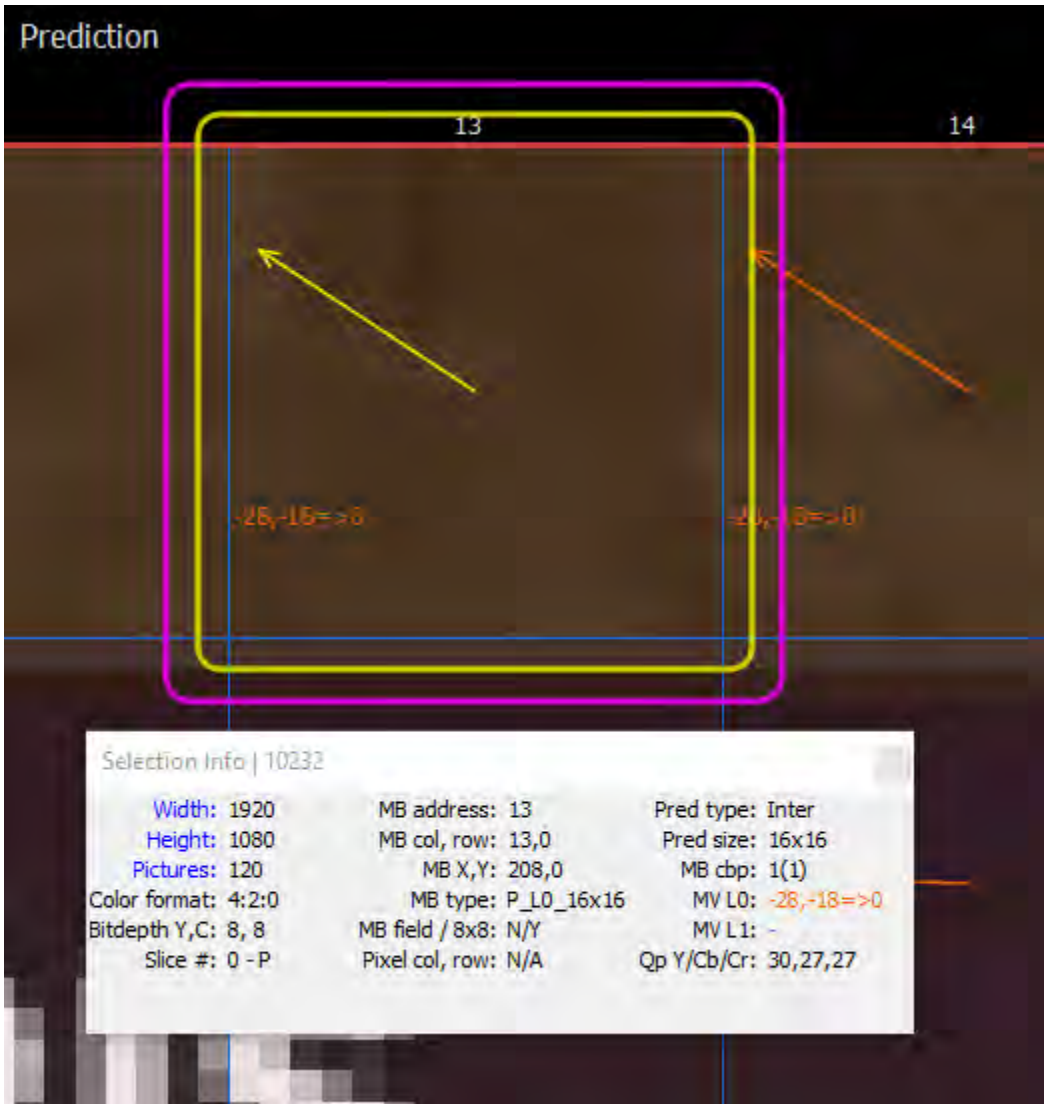
279. When encoding Twitch.tv video, on information and belief, Amazon performs when values for sub-pixels at $½^{N-1}$ unit horizontal and unit vertical locations, and unit horizontal and $½^{N-1}$ unit vertical locations are required, interpolating such values directly using weighted sums of pixels residing at unit horizontal and unit vertical locations, as demonstrated in the screenshots below using VQ Analyzer software, where the -28, -18 motion vector corresponds to unit horizontal and $½^{N-1}$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



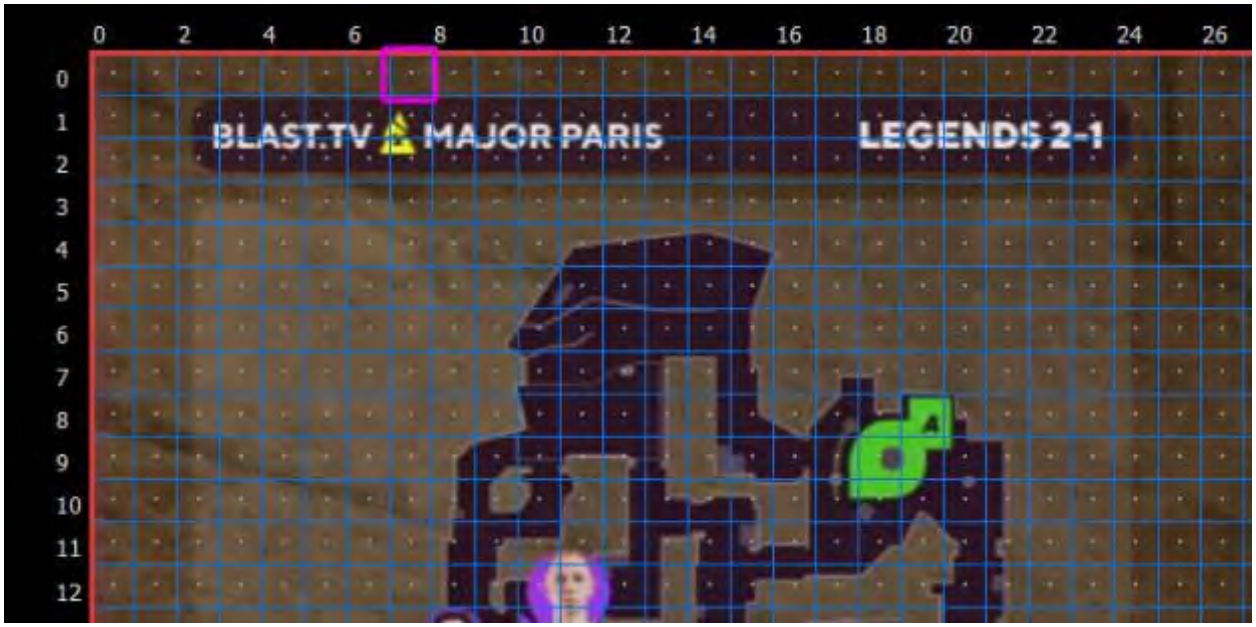
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



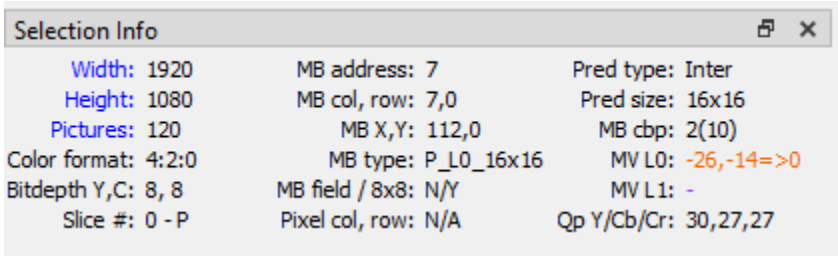
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

280. When encoding Twitch.tv video, on information and belief, Amazon performs when values for sub-pixels at $½^{N-1}$ unit horizontal and $½^{N-1}$ unit vertical locations are required, interpolating such values directly using a choice of a first weighted sum of values for sub-pixels residing at $½^{N-1}$ unit horizontal and unit vertical locations and a second weighted sum of values for sub-pixels residing at unit horizontal and $½^{N-1}$ unit vertical locations, the first and second weighted sums of values being calculated according to step (a), as demonstrated in the screenshots

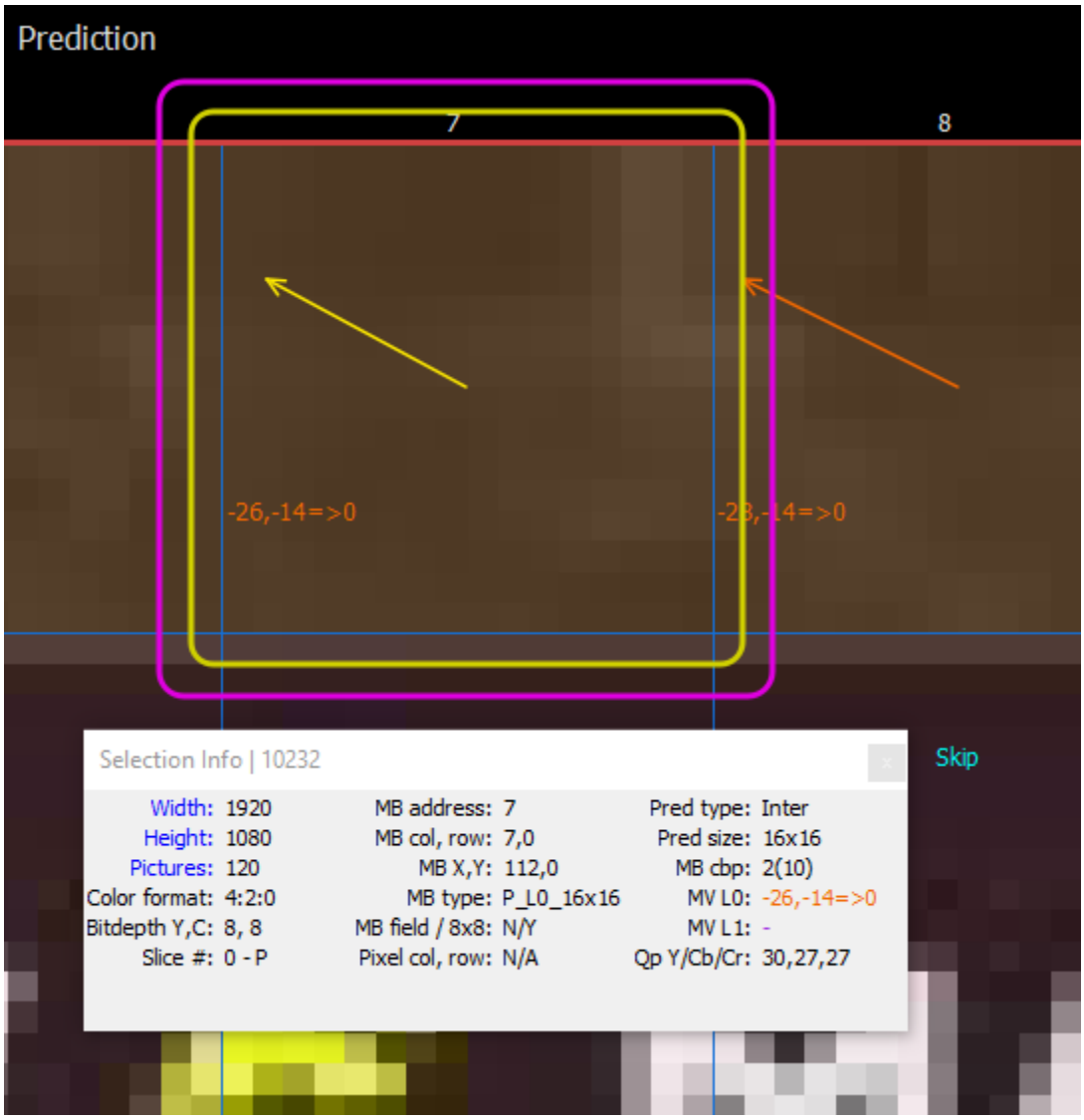
below using VQ Analyzer software, where the -26, -14 motion vector corresponds to $½^{N-1}$ unit horizontal and $½^{N-1}$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



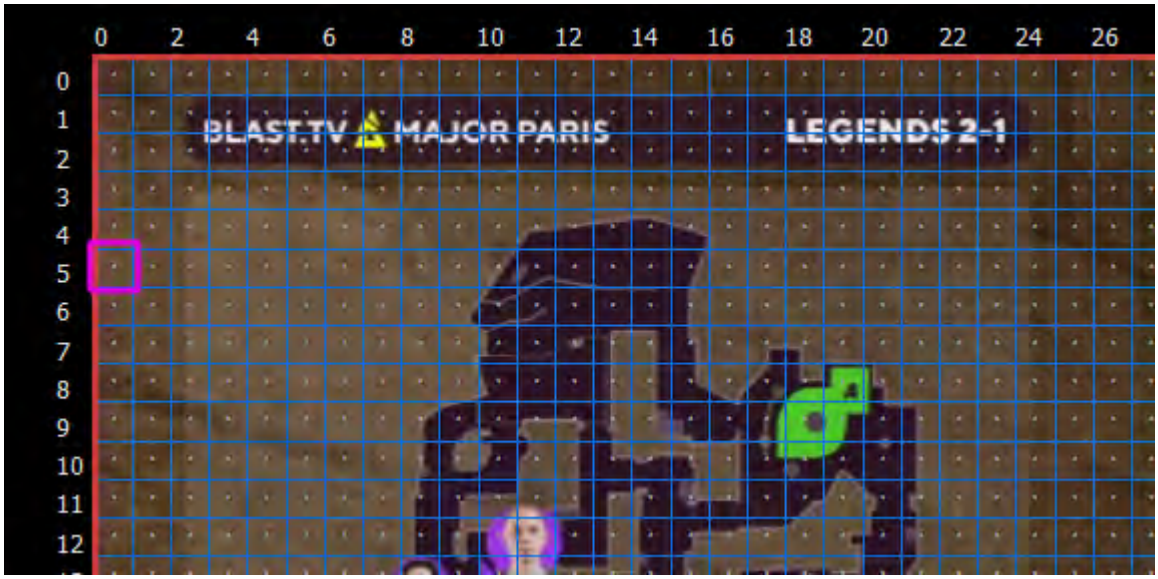
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



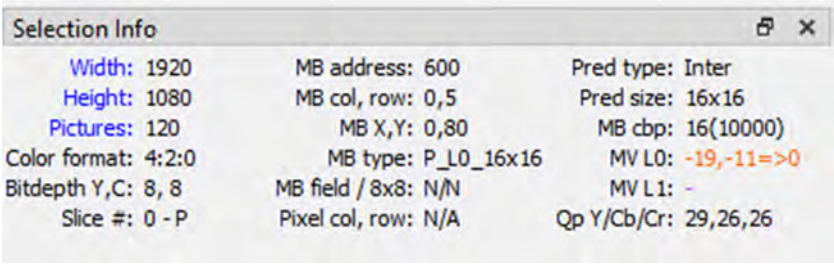
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

281. When encoding Twitch.tv video, on information and belief, Amazon performs when a value for a sub-pixel situated at a $½^N$ unit horizontal and $½^N$ unit vertical location is required, interpolating such a value by taking a weighted average of the value of a first sub-pixel or pixel situated at a $½^{N-m}$ unit horizontal and $½^{N-n}$ unit vertical location and the value of a second sub-pixel or pixel located at a $½^{N-p}$ unit horizontal and $½^{N-q}$ unit vertical location, variables m, n, p and q taking integer values in the range 1 to N such that the first and second sub-pixels or pixels are located diagonally with respect to the sub-pixel at $½^N$ unit horizontal and $½^N$ vertical locations,

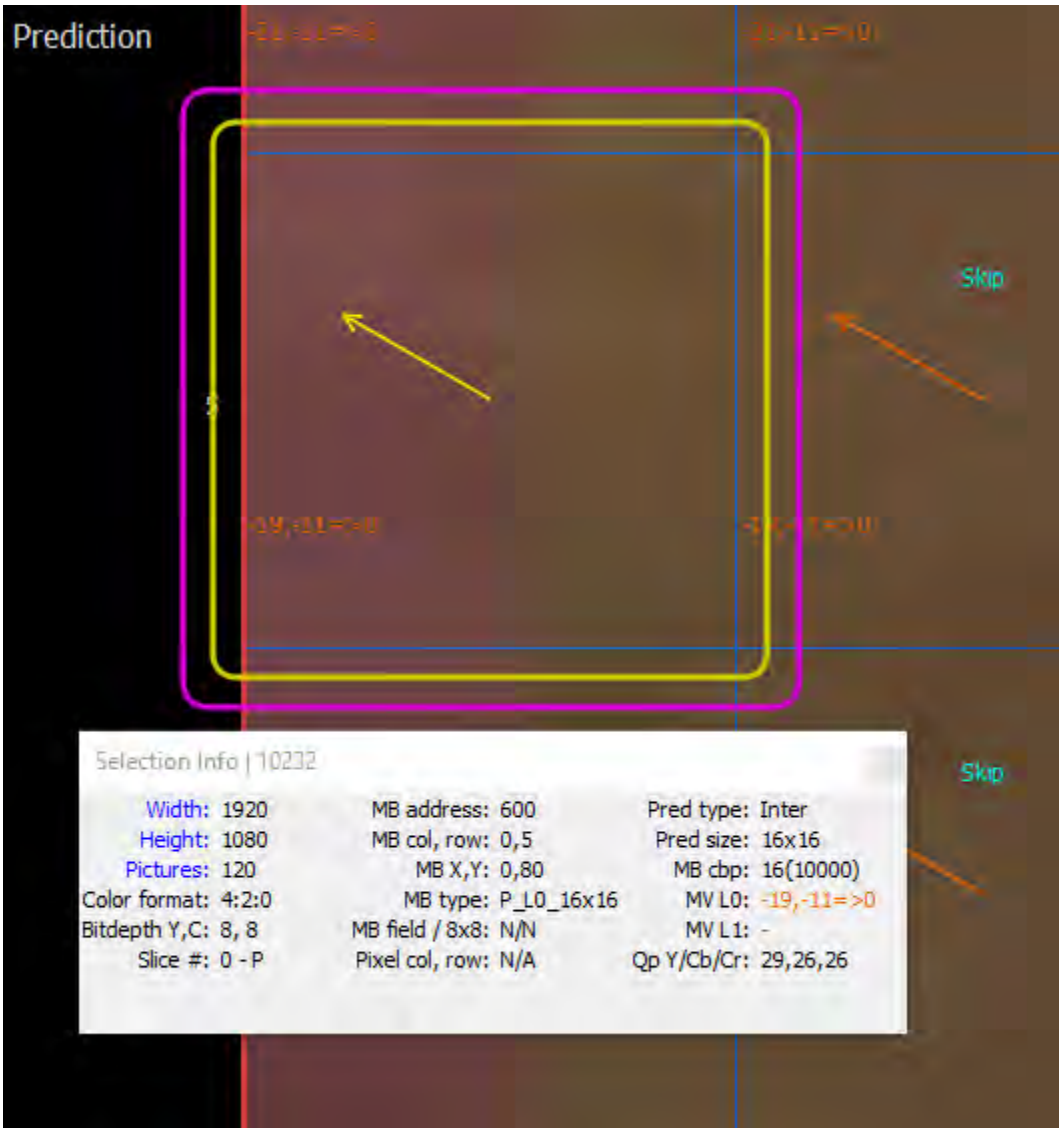
as demonstrated in the screenshots below using VQ Analyzer software, where the -19, -11 motion vector corresponds to $½^N$ unit horizontal and $½^N$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

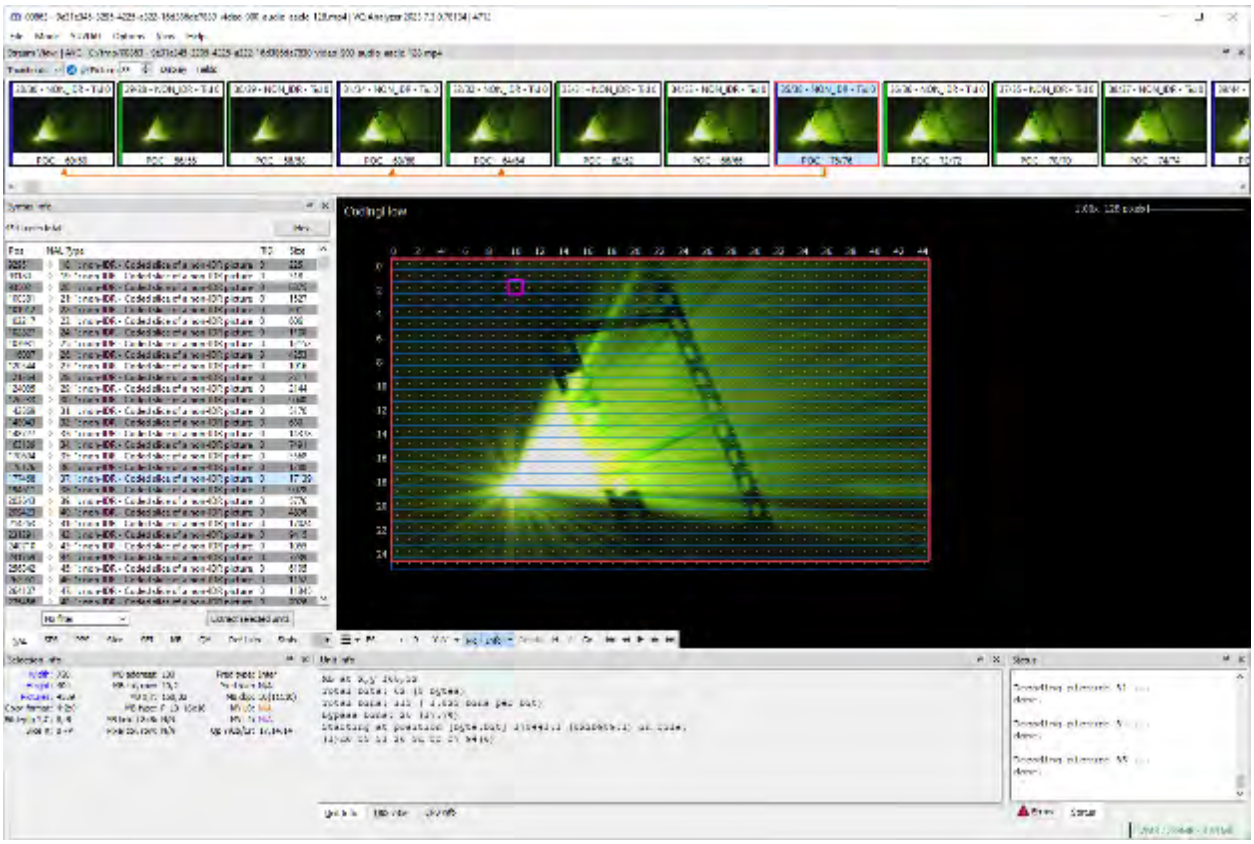
E. Amazon Makes, Imports, Uses, Sells, and/or Offers for Sale Products and Services that Infringe the ’599 Patent

282. The Accused Products infringe one or more claims of the ’599 Patent, including, for example, claim 1.

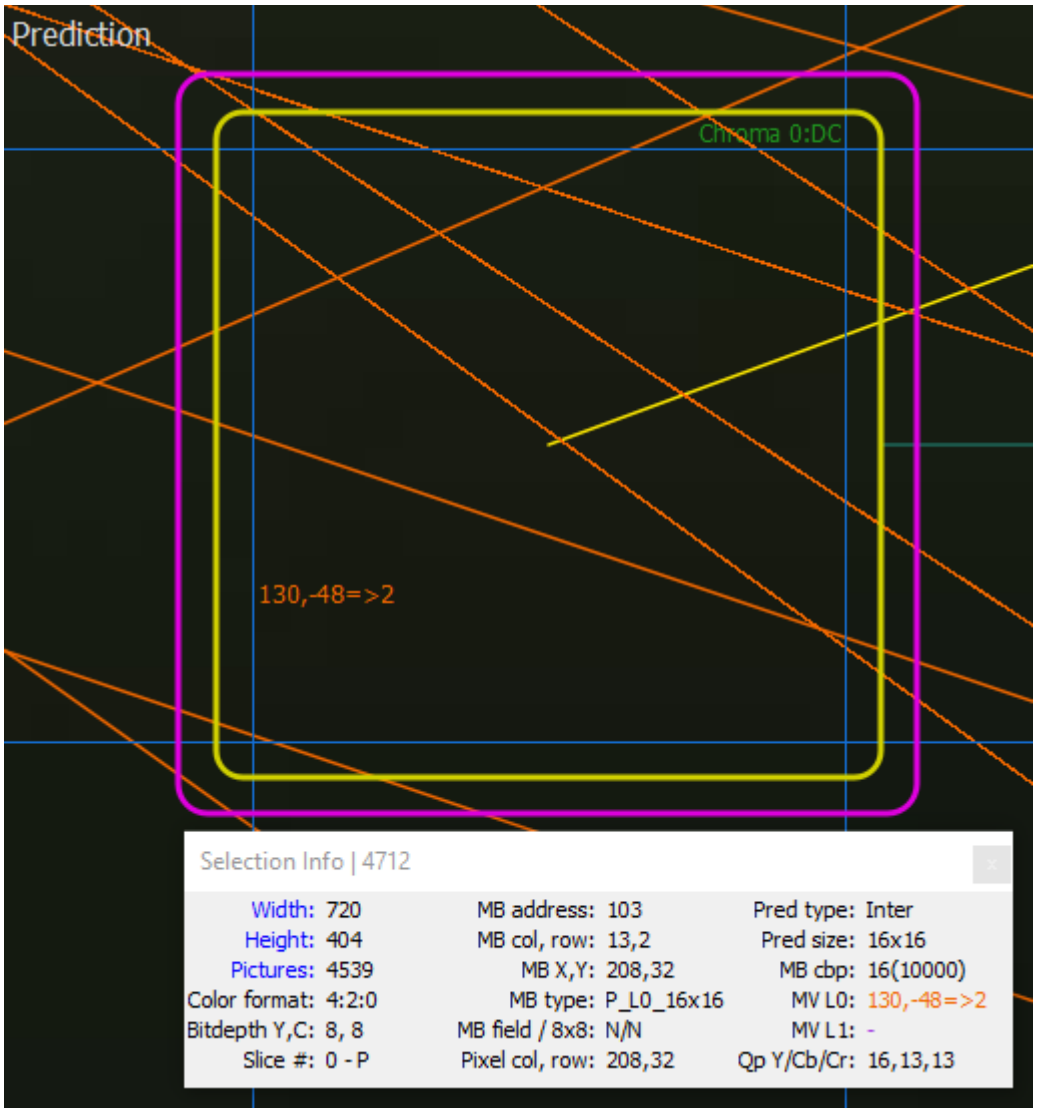
283. As just one example of infringement, on information and belief, Amazon performs a method of sub-pixel value interpolation in video coding in a manner that is covered by claim 1 of the ’599 Patent for Amazon Prime Video content (such as trailers), as demonstrated in the

screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 212.

284. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs sub-pixel value interpolation to determine values for sub-pixels situated within a rectangular bounded region defined by four corner pixels with no intermediate pixels between the corners, the pixels and sub-pixels being arranged in rows and columns, the pixel and sub-pixel locations being representable mathematically within the rectangular bounded region using the co-ordinate notation $K/2^N$, $L/2^N$, K and L being positive integers having respective values between zero and 2^N, N being a positive integer greater than one and representing a particular degree of sub-pixel value interpolation, as demonstrated in the screenshots below using VQ Analyzer software.



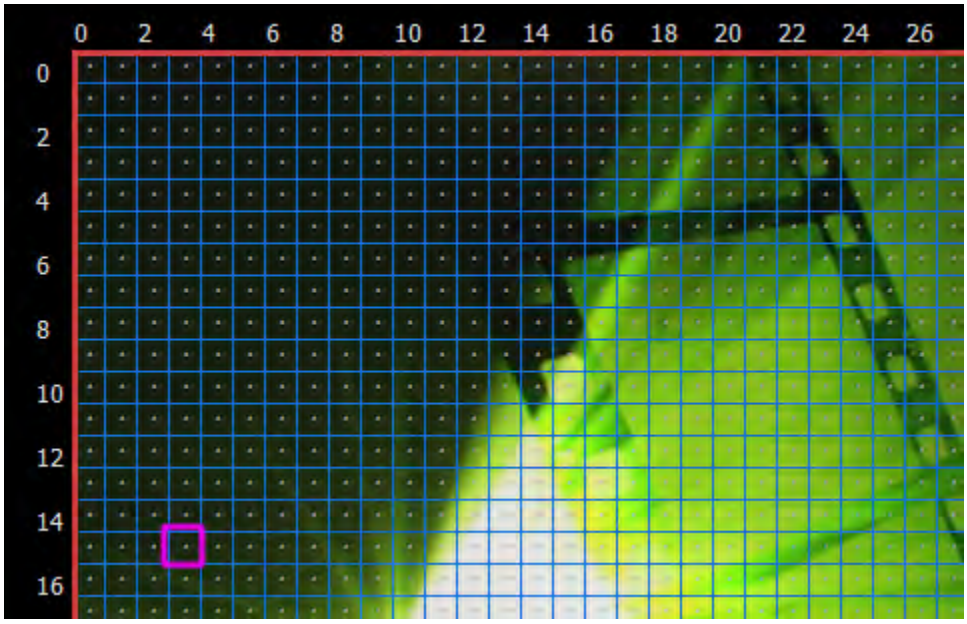
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



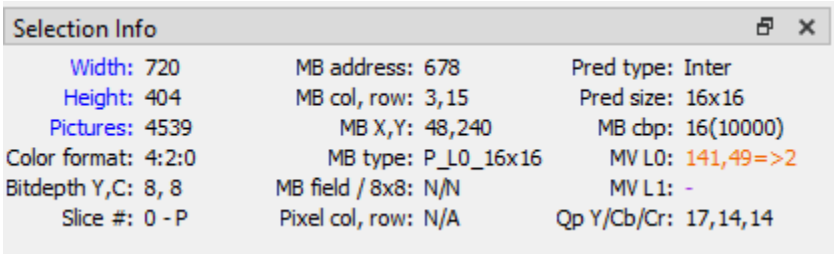
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

285. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs a method of interpolating a sub-pixel value for a sub-pixel having co-ordinates with odd values of both K and L, according to a predetermined choice of a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates 1/2, 1/2, and a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of both K and L, including zero, situated within a quadrant of the rectangular bounded region defined by corner pixels having co-ordinates 1/2, 1/2 and the nearest

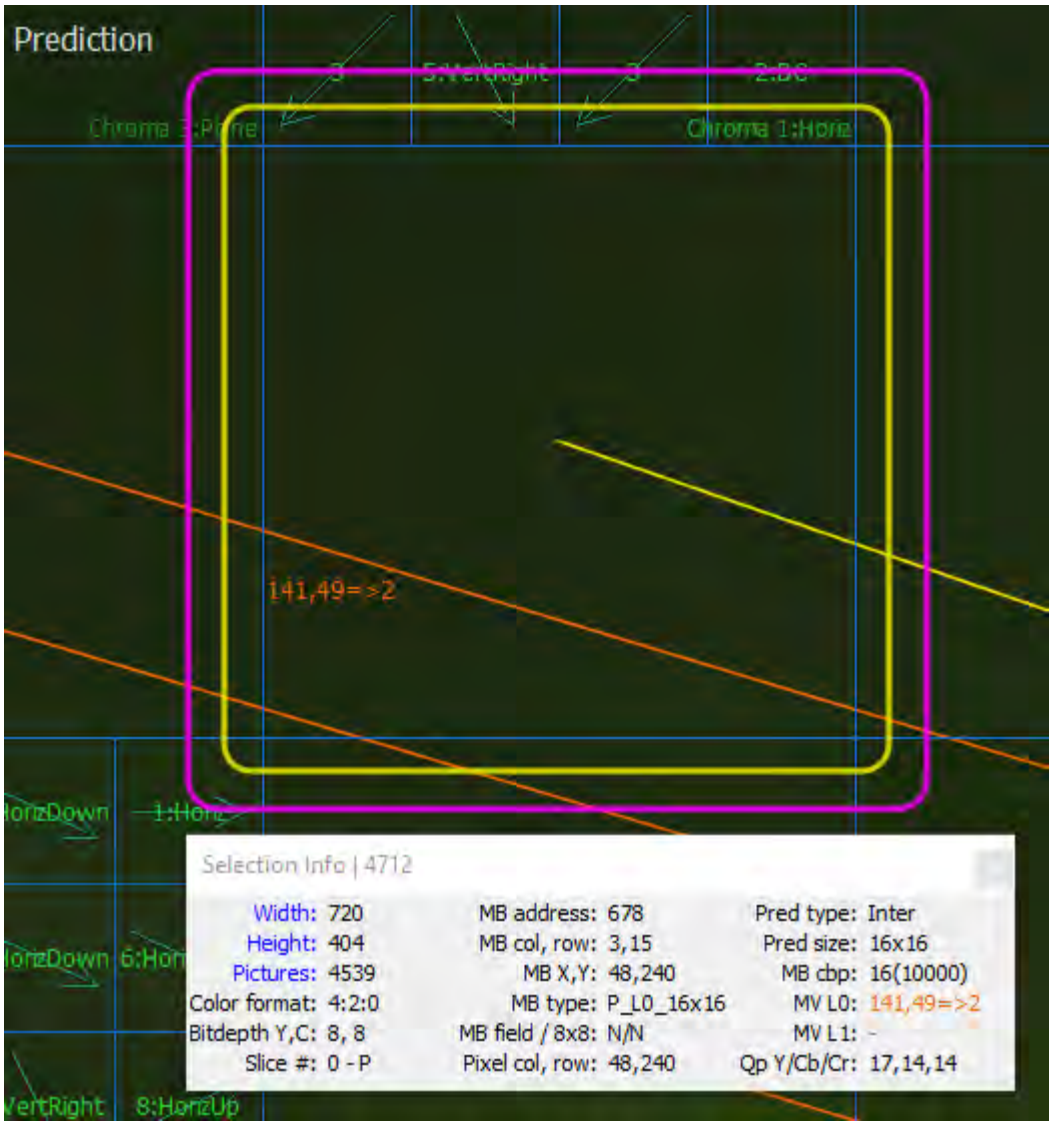
neighbouring pixel, as demonstrated in the screenshots below using VQ Analyzer software, where the 141, 49 motion vector corresponds to $K/2^N$ unit horizontal and $L/2^N$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

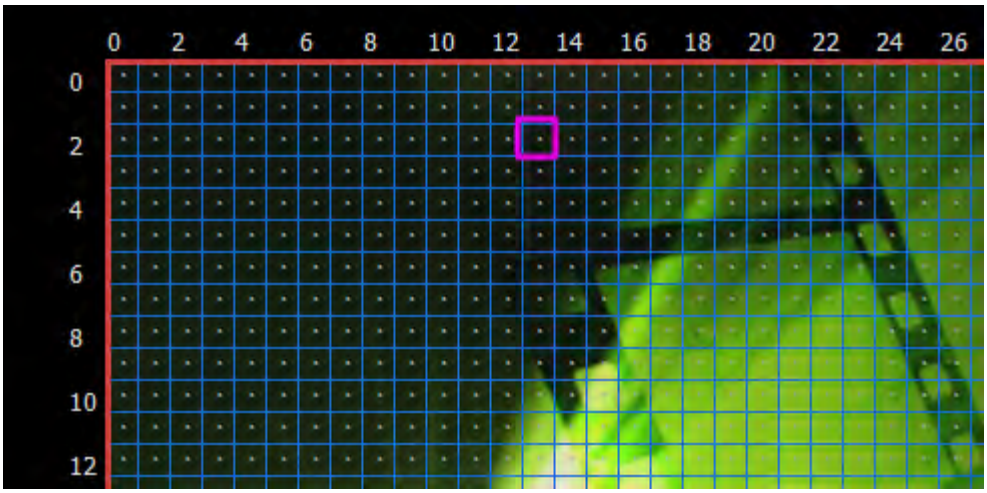


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

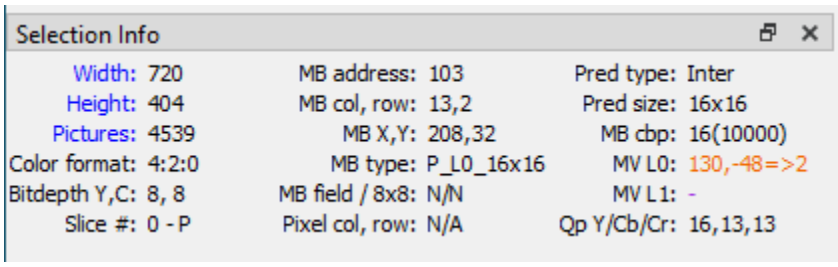


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

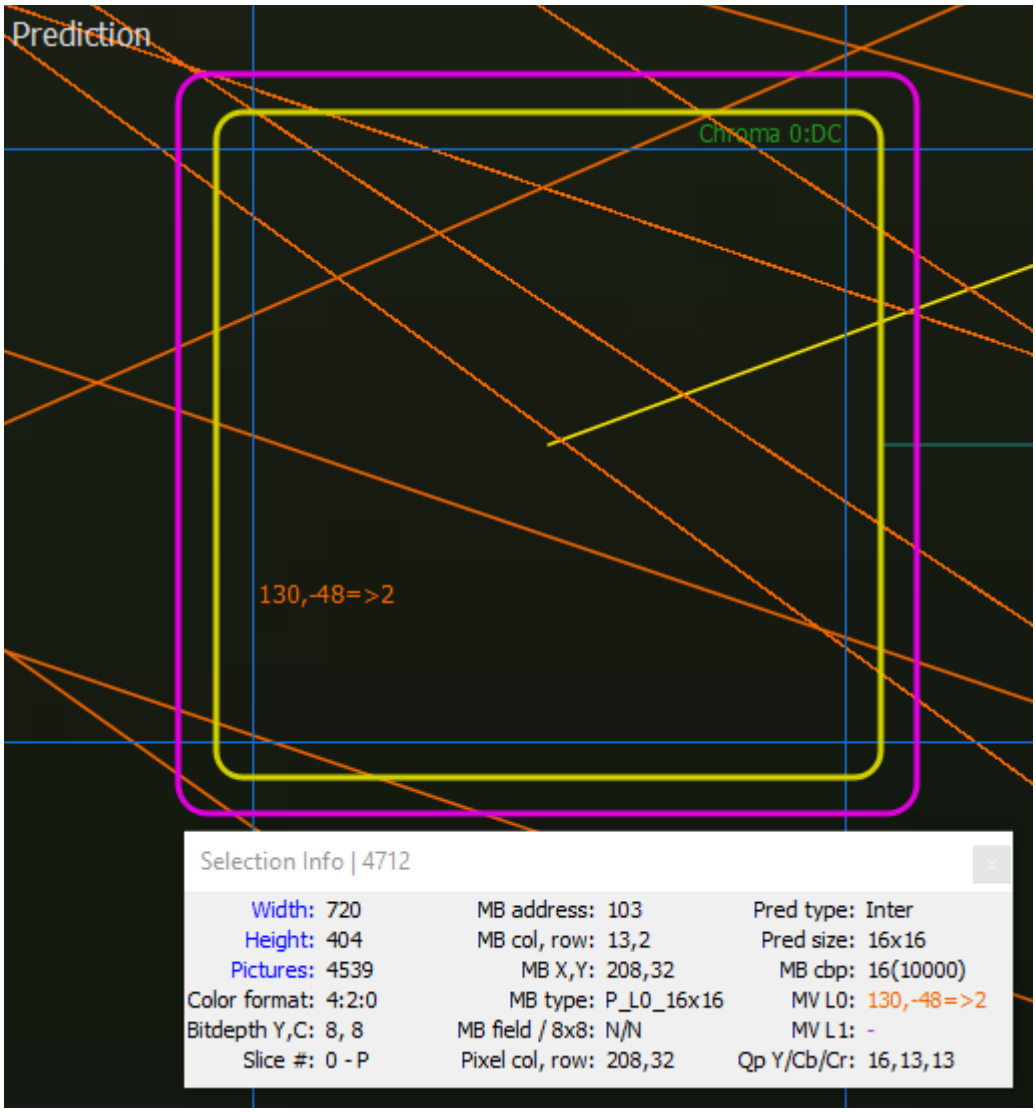
286. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs a method of interpolating sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of both K and L, using weighted sums of the values of pixels located in rows and columns respectively, as demonstrated in the screenshots below using VQ Analyzer software, where the 130, -48 motion vector corresponds to $K/2^N$ unit horizontal and unit vertical locations.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

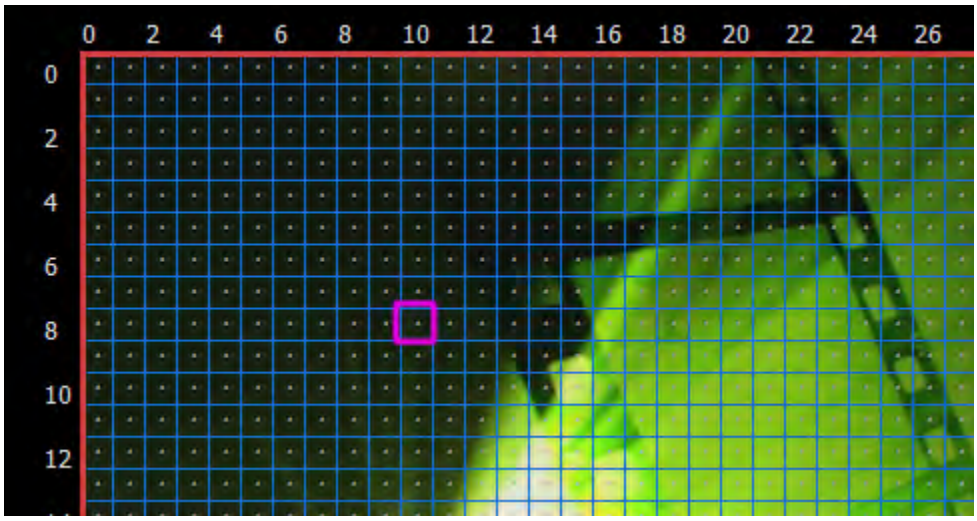


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

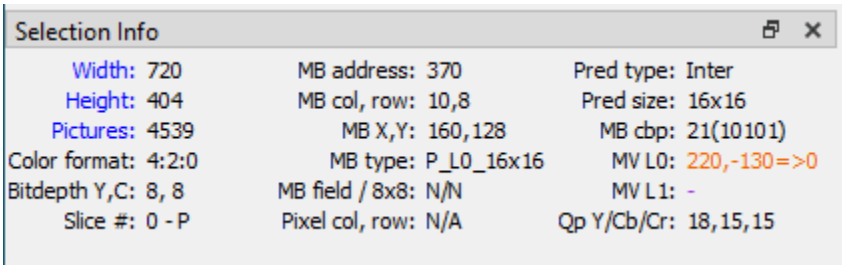


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

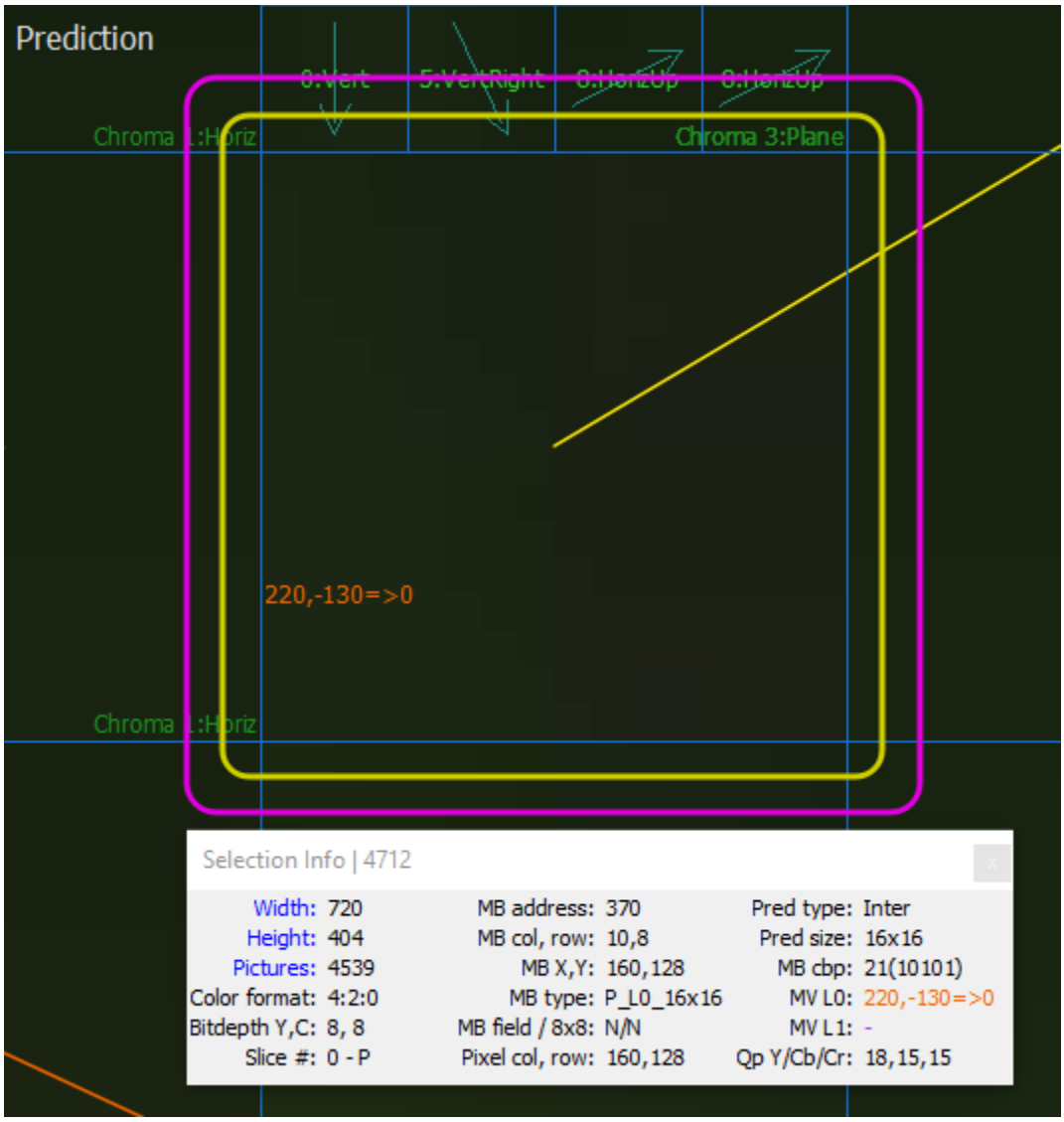
287. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs a method of interpolating sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of both K and L, using weighted sums of the values of pixels located in rows and columns respectively, as demonstrated in the screenshots below using VQ Analyzer software, where the 220, -130 motion vector corresponds to unit horizontal and $L/2^N$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



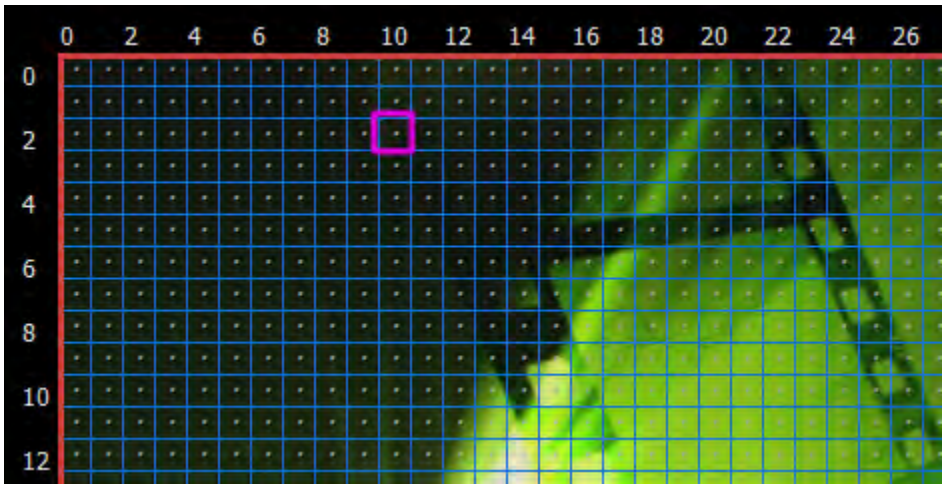
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



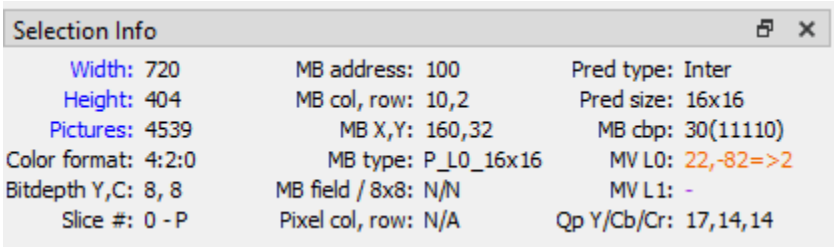
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

288. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs a method of interpolating sub-pixel values for sub-pixels having co-ordinates with even values of both K and L, used in the interpolation of sub-pixel values for the sub-pixels having co-ordinates with odd values of both K and L, using a predetermined choice of either a weighted sum of the values of sub-pixels having co-ordinates with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the values of sub-pixels having co-ordinates

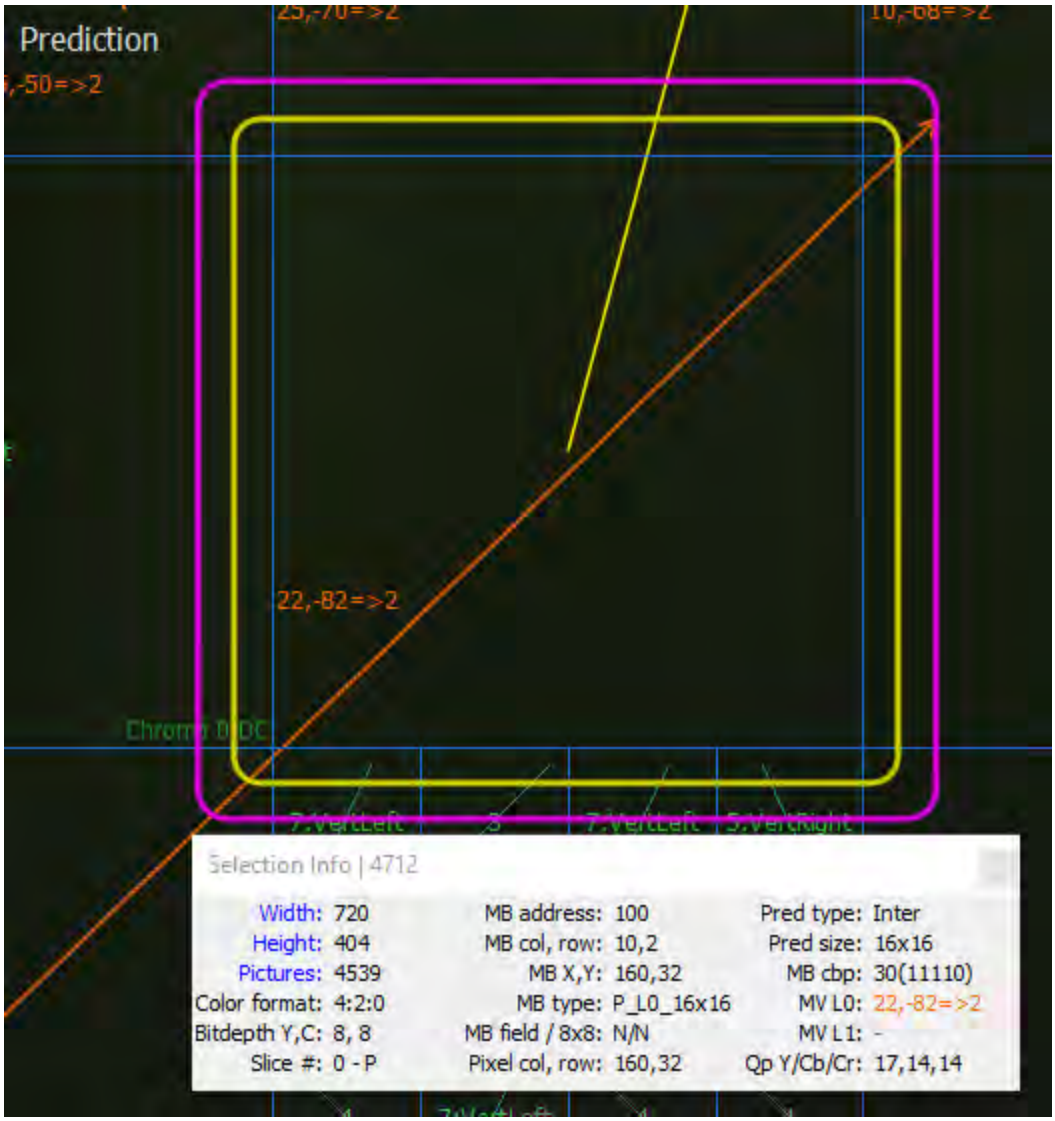
with K equal to zero and L equal to an even value and the values of sub-pixels having corresponding co-ordinates in immediately adjacent bounded rectangular regions, as demonstrated in the screenshots below using VQ Analyzer software, where the 22, -82 motion vector corresponds to $K/2^N$ unit horizontal and $L/2^N$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

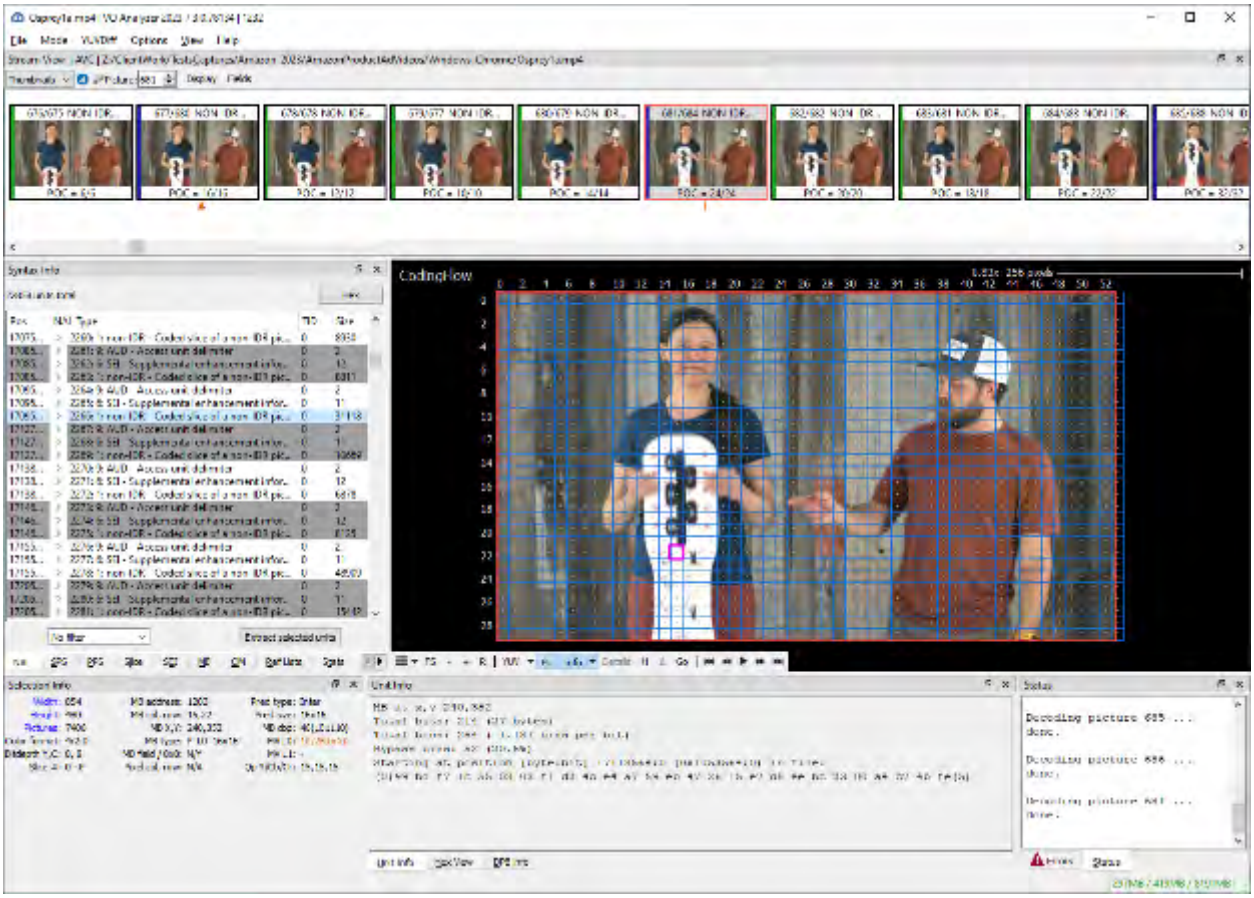


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

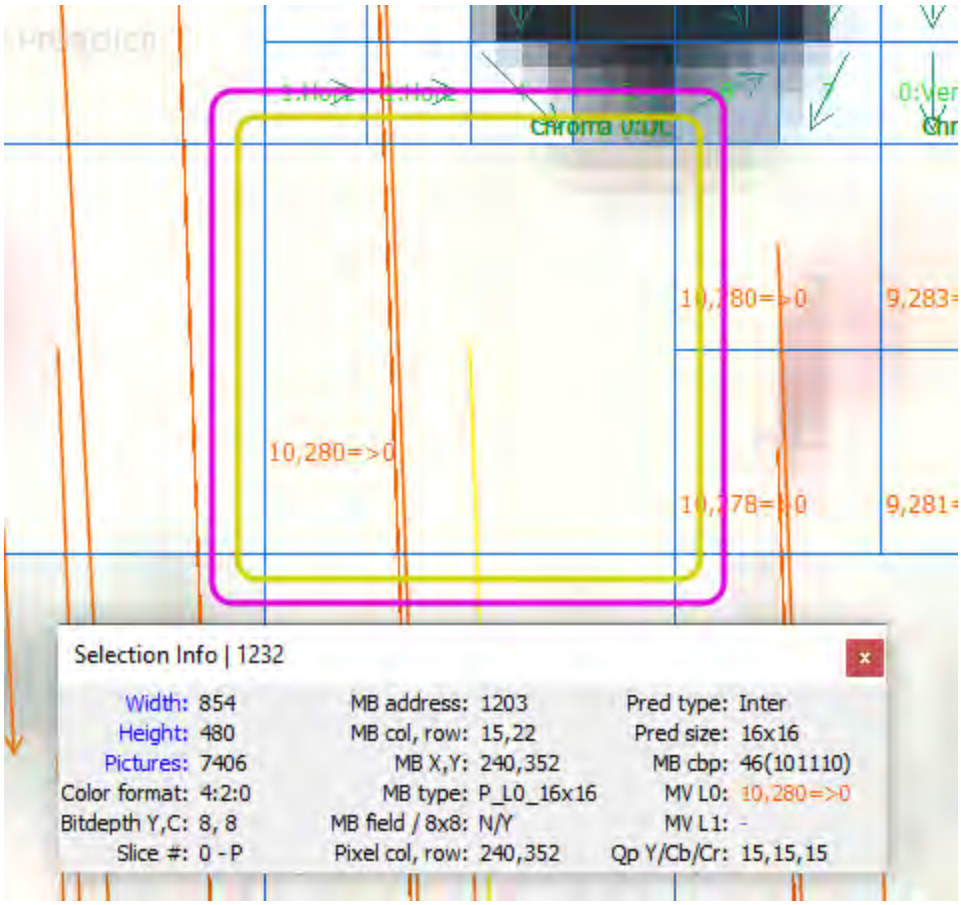
289. For another example, on information and belief, Amazon performs a method of sub-pixel value interpolation in video coding in a manner that is covered by claim 1 of the '599 Patent for Amazon.com advertisements, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 217.

290. When encoding Amazon.com advertisements, on information and belief, Amazon performs sub-pixel value interpolation to determine values for sub-pixels situated within a

rectangular bounded region defined by four corner pixels with no intermediate pixels between the corners, the pixels and sub-pixels being arranged in rows and columns, the pixel and sub-pixel locations being representable mathematically within the rectangular bounded region using the co-ordinate notation $K/2^N$, $L/2^N$, K and L being positive integers having respective values between zero and 2^N, N being a positive integer greater than one and representing a particular degree of sub-pixel value interpolation, as demonstrated in the screenshots below using VQ Analyzer software.

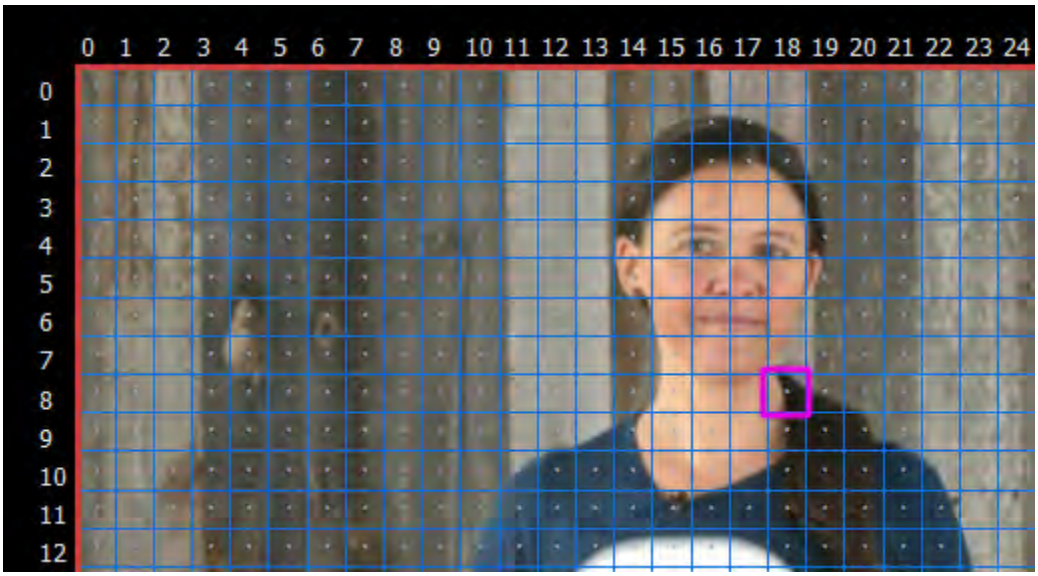


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

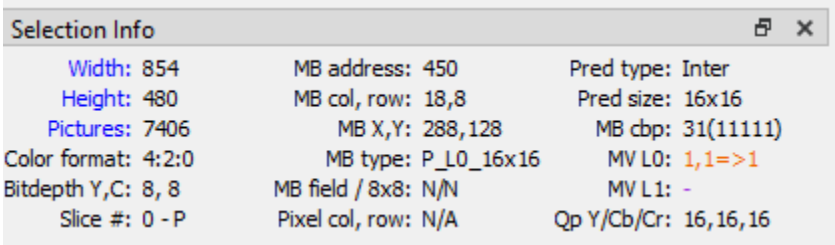


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

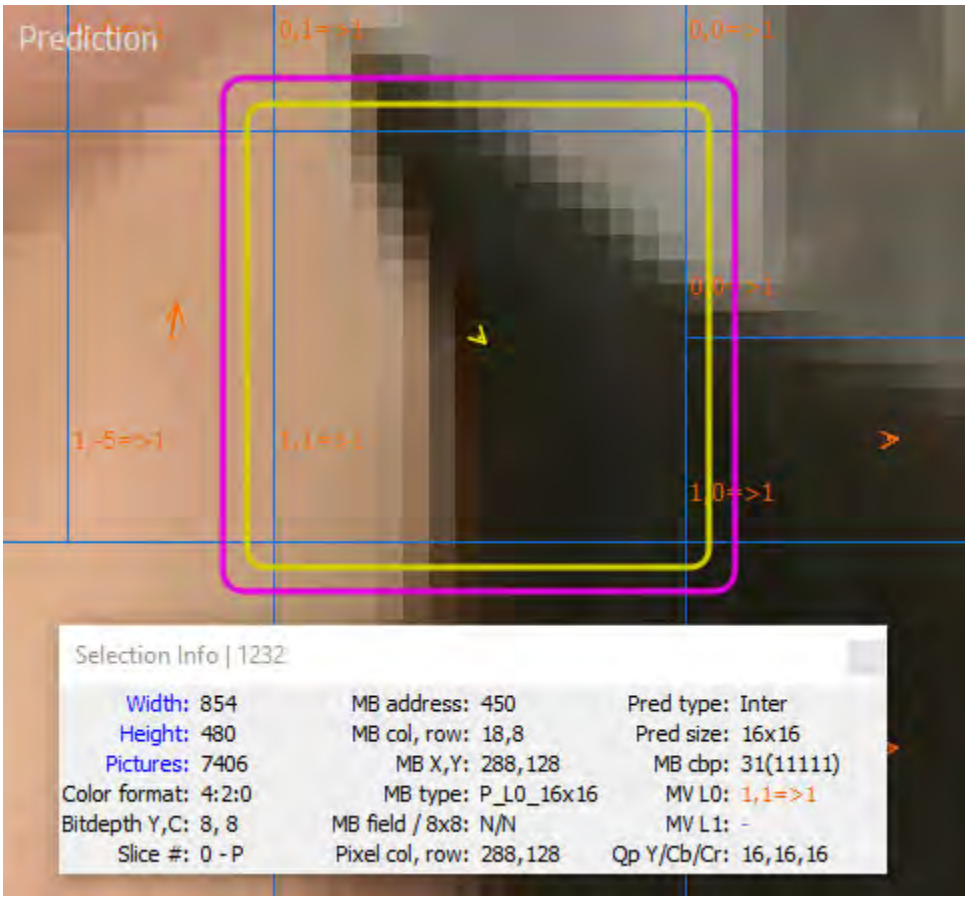
291. When encoding Amazon.com advertisements, on information and belief, Amazon performs a method of interpolating a sub-pixel value for a sub-pixel having co-ordinates with odd values of both K and L, according to a predetermined choice of a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates 1/2, 1/2, and a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of both K and L, including zero, situated within a quadrant of the rectangular bounded region defined by corner pixels having co-ordinates 1/2, 1/2 and the nearest neighbouring pixel, as demonstrated in the screenshots below using VQ Analyzer software, where the 1, 1 motion vector corresponds to $K/2^N$ unit horizontal and $L/2^N$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

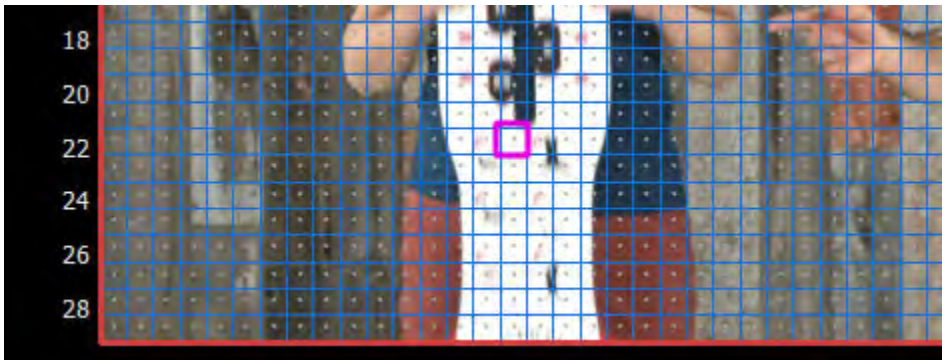


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

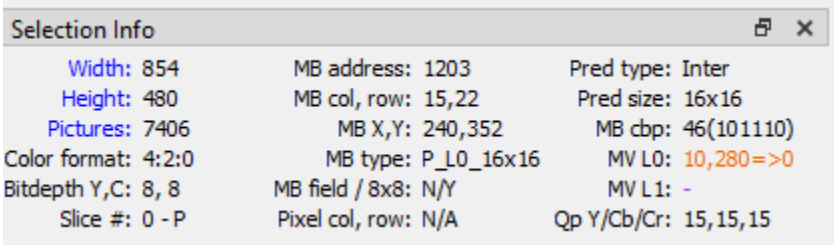


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

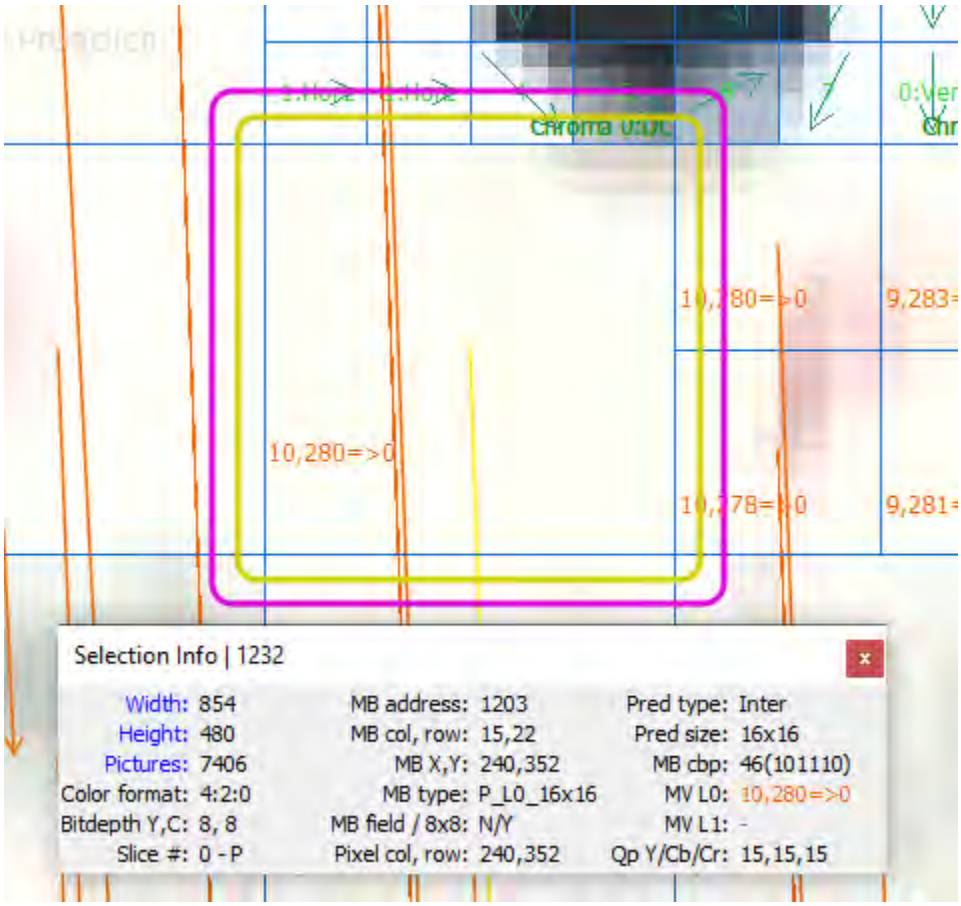
292. When encoding Amazon.com advertisements, on information and belief, Amazon performs a method of interpolating sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of both K and L, using weighted sums of the values of pixels located in rows and columns respectively, as demonstrated in the screenshots below using VQ Analyzer software, where the 10, 280 motion vector corresponds to $K/2^N$ unit horizontal and unit vertical locations.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

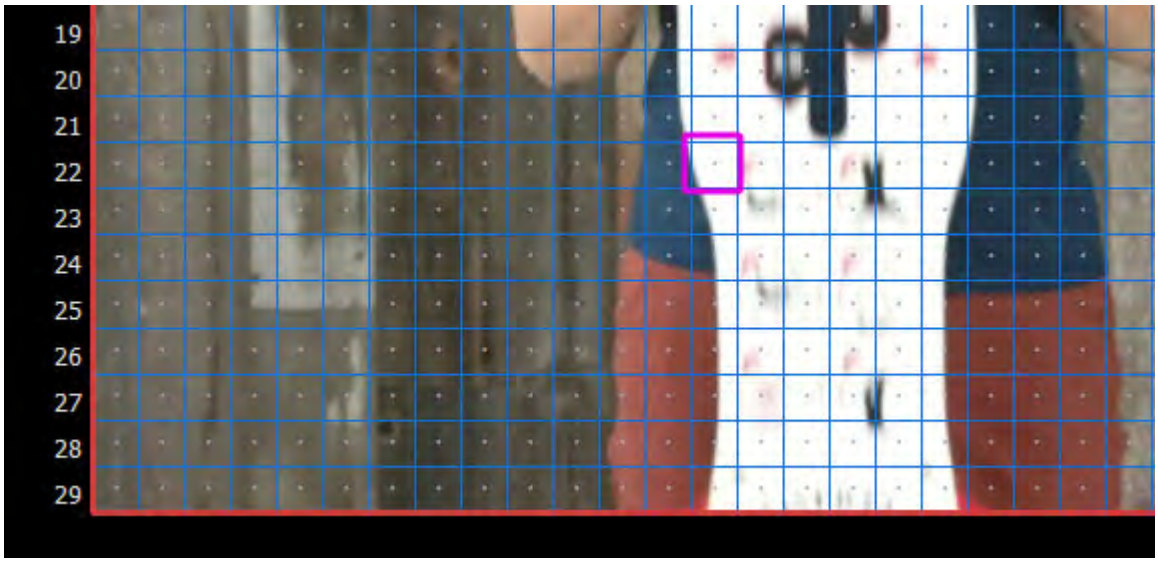


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

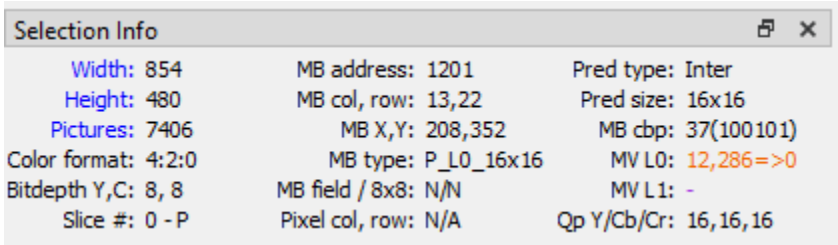


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

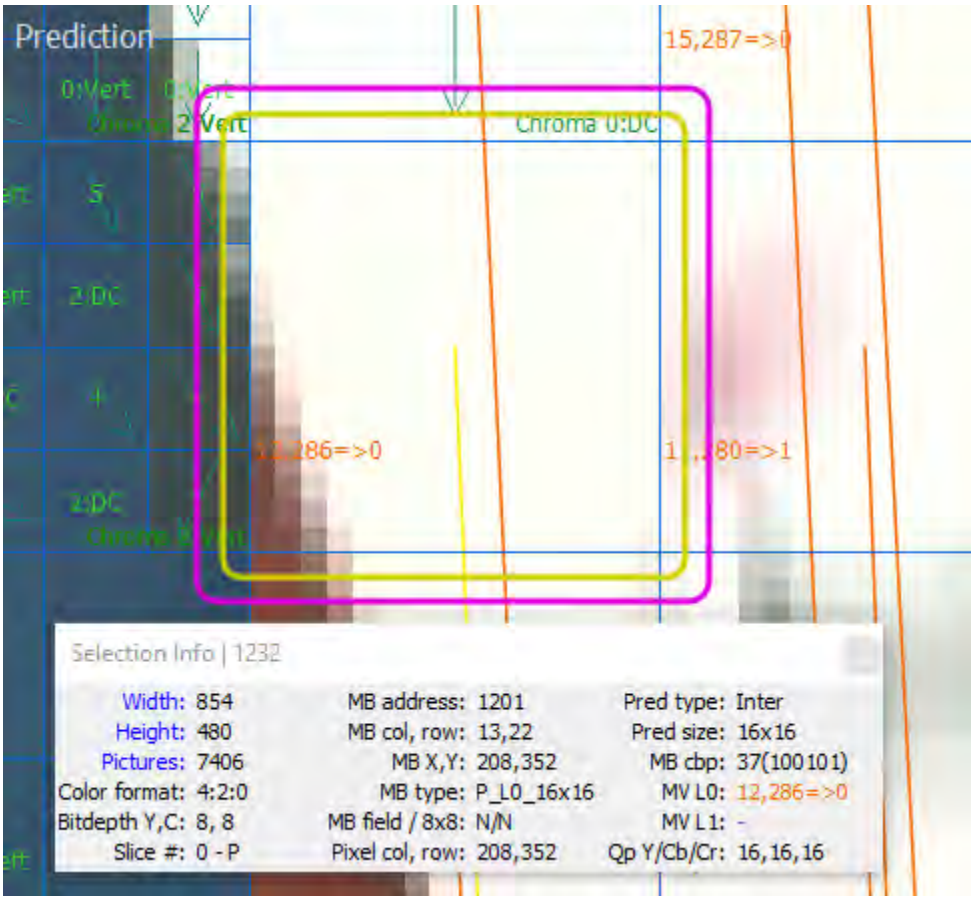
293. When encoding Amazon.com advertisements, on information and belief, Amazon performs a method of interpolating sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of both K and L, using weighted sums of the values of pixels located in rows and columns respectively, as demonstrated in the screenshots below using VQ Analyzer software, where the 12, 286 motion vector corresponds to unit horizontal and $L/2^N$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



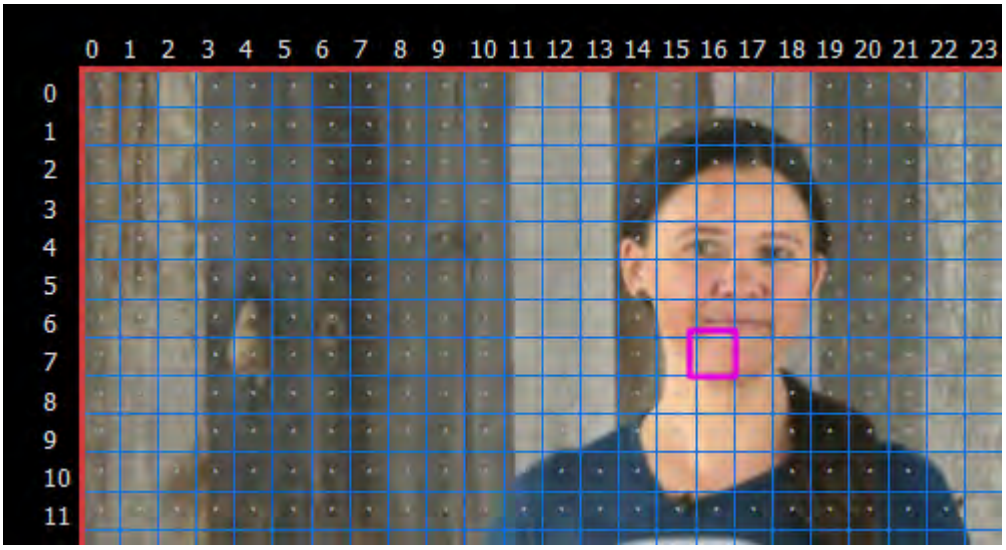
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



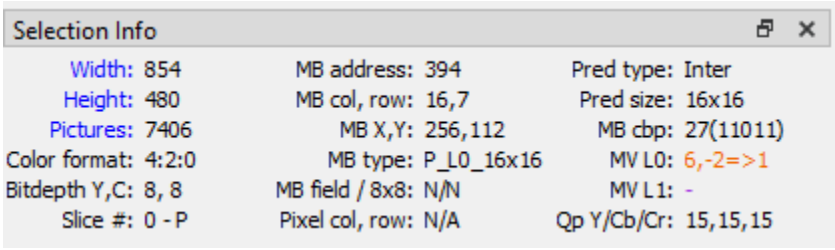
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

294. When encoding Amazon.com advertisements, on information and belief, Amazon performs a method of interpolating sub-pixel values for sub-pixels having co-ordinates with even values of both K and L, used in the interpolation of sub-pixel values for the sub-pixels having co-ordinates with odd values of both K and L, using a predetermined choice of either a weighted sum of the values of sub-pixels having co-ordinates with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the values of sub-pixels having co-ordinates with K equal to zero and L equal to an even value and the values of sub-pixels having corresponding co-ordinates in immediately adjacent bounded rectangular regions, as

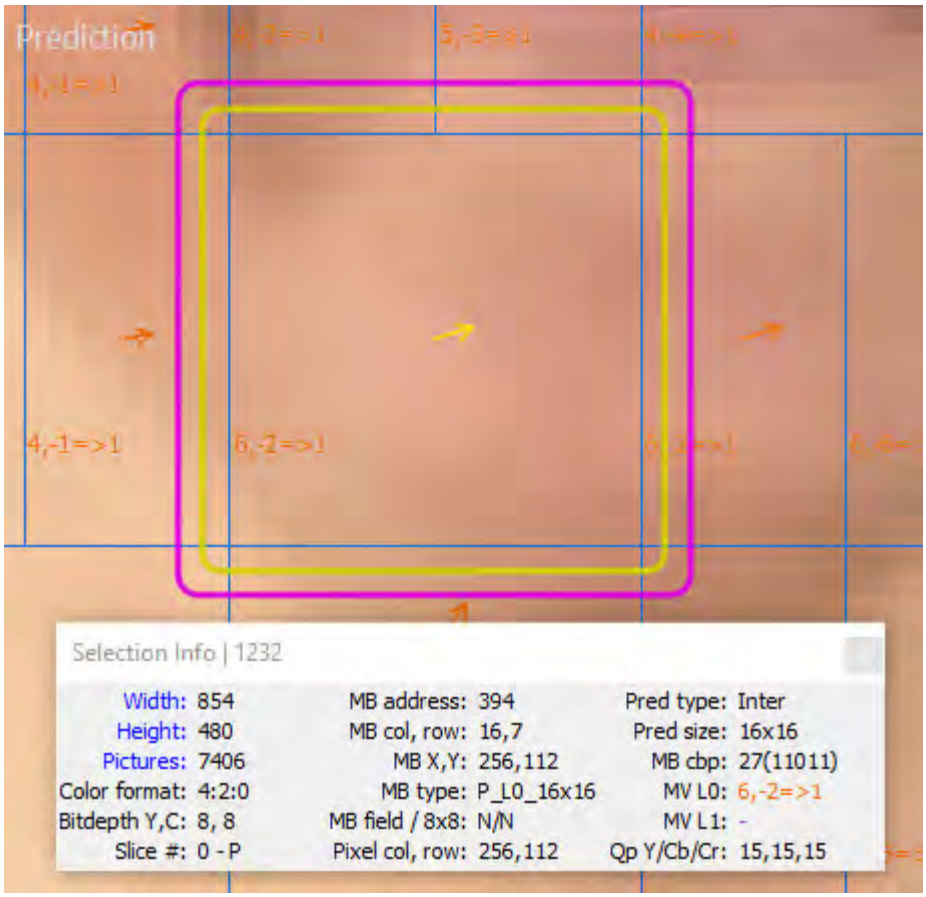
demonstrated in the screenshots below using VQ Analyzer software, where the 6, -2 motion vector corresponds to $K/2^N$ unit horizontal and $L/2^N$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

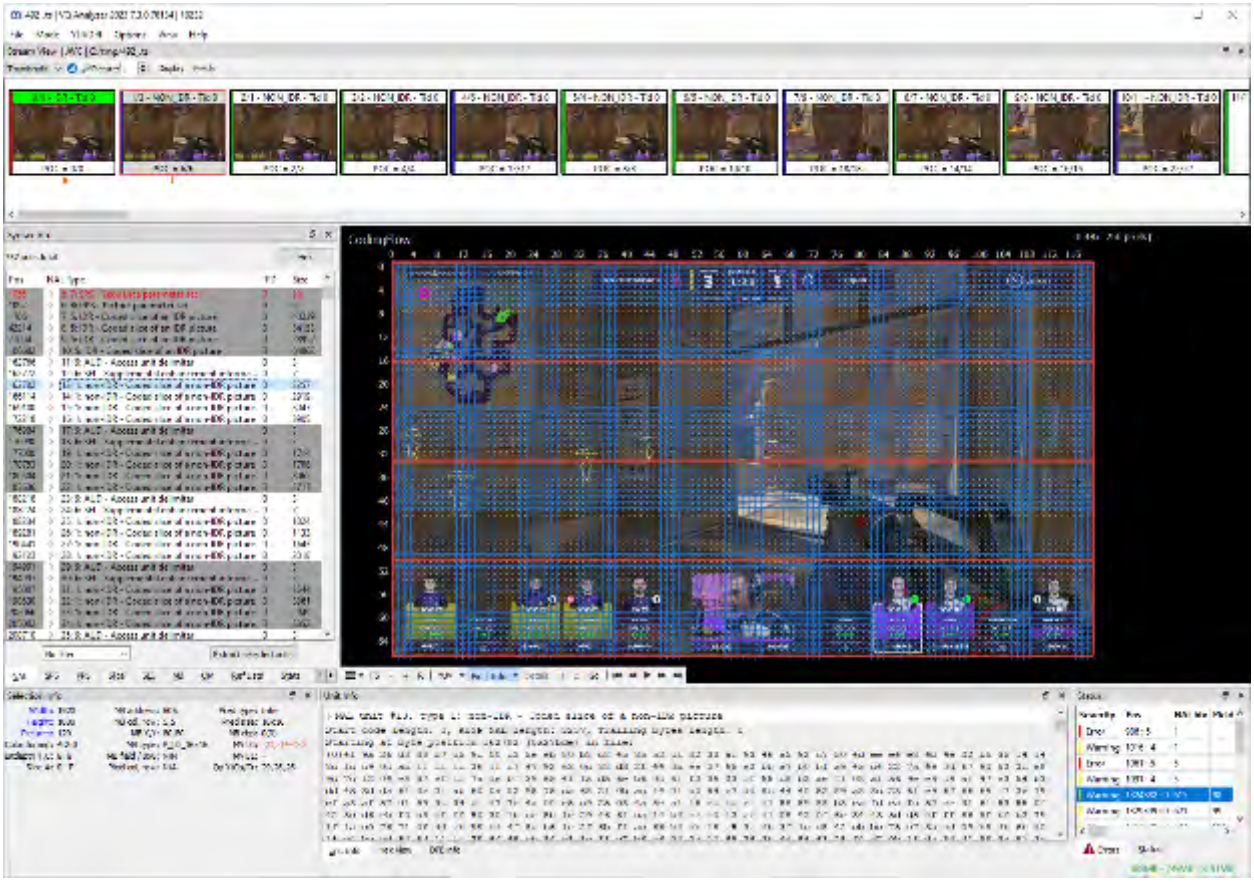


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

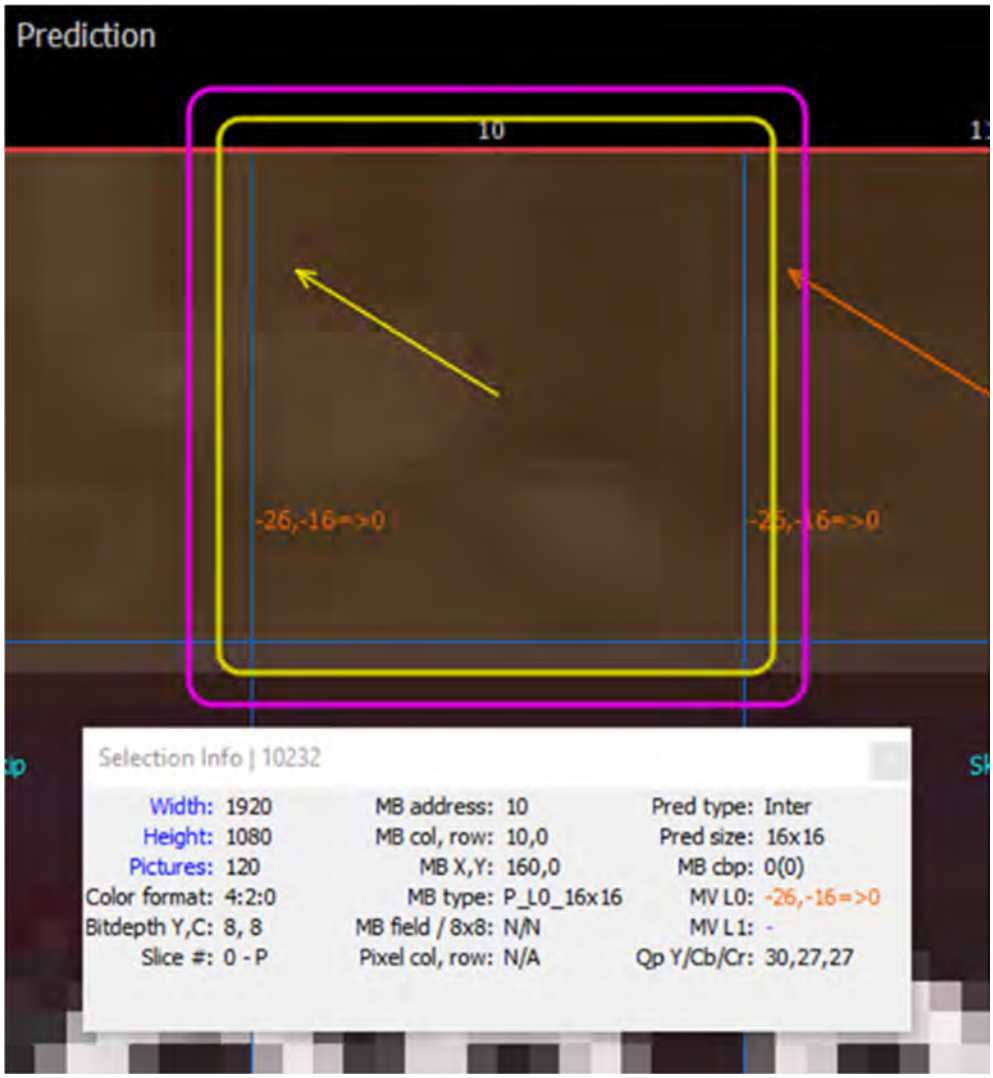
295. For another example, on information and belief, Amazon performs a method of sub-pixel value interpolation in video coding in a manner that is covered by claim 1 of the ’599 Patent for Twitch.tv video, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 222.

296. When encoding Twitch.tv video, on information and belief, Amazon performs sub-pixel value interpolation to determine values for sub-pixels situated within a rectangular bounded region defined by four corner pixels with no intermediate pixels between the corners, the pixels and sub-pixels being arranged in rows and columns, the pixel and sub-pixel locations being representable mathematically within the rectangular bounded region using the co-ordinate

notation $K/2^N$, $L/2^N$, K and L being positive integers having respective values between zero and 2^N, N being a positive integer greater than one and representing a particular degree of sub-pixel value interpolation, as demonstrated in the screenshots below using VQ Analyzer software.



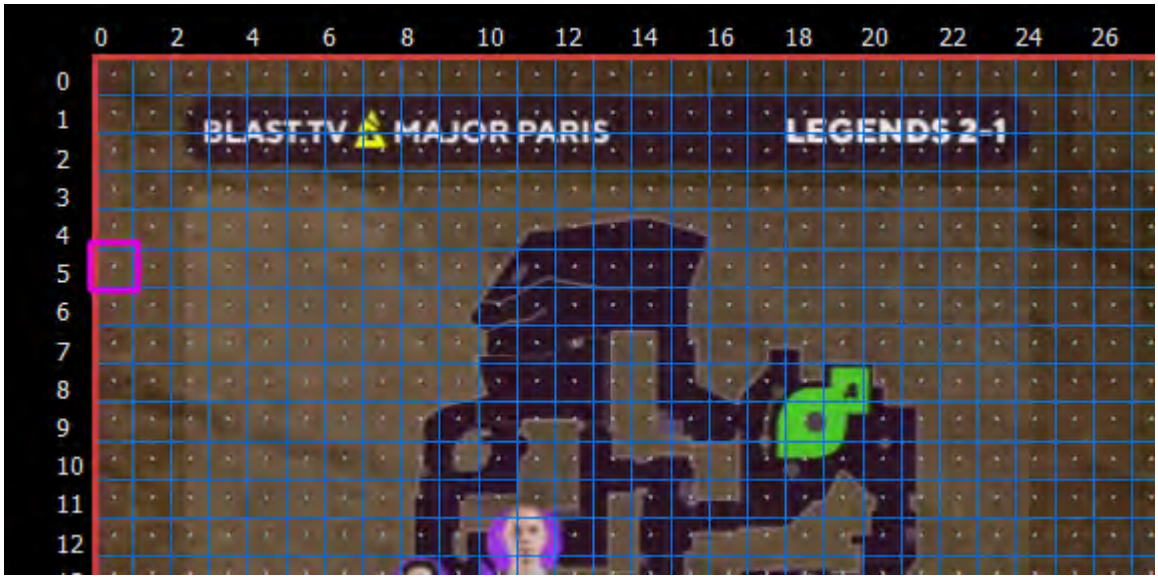
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



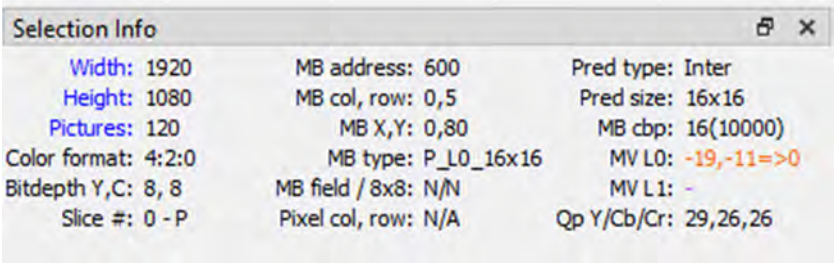
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

297. When encoding Twitch.tv video, on information and belief, Amazon performs a method of interpolating a sub-pixel value for a sub-pixel having co-ordinates with odd values of both K and L, according to a predetermined choice of a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates 1/2, 1/2, and a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of both K and L, including zero, situated within a quadrant of the rectangular bounded region defined by corner pixels having co-ordinates 1/2, 1/2 and the nearest neighbouring pixel,

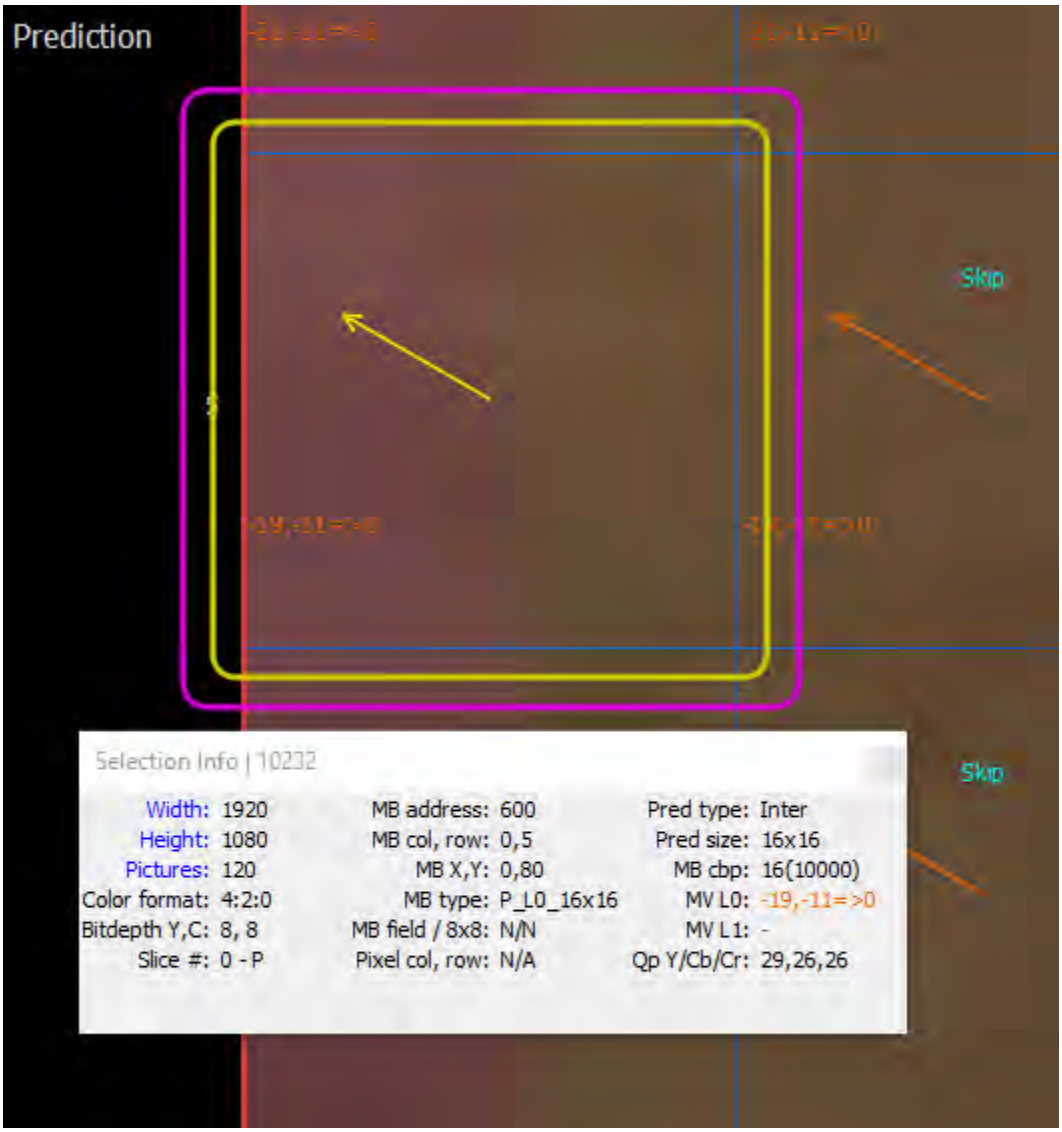
as demonstrated in the screenshots below using VQ Analyzer software, where the -19, -11 motion vector corresponds to $K/2^N$ unit horizontal and $L/2^N$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

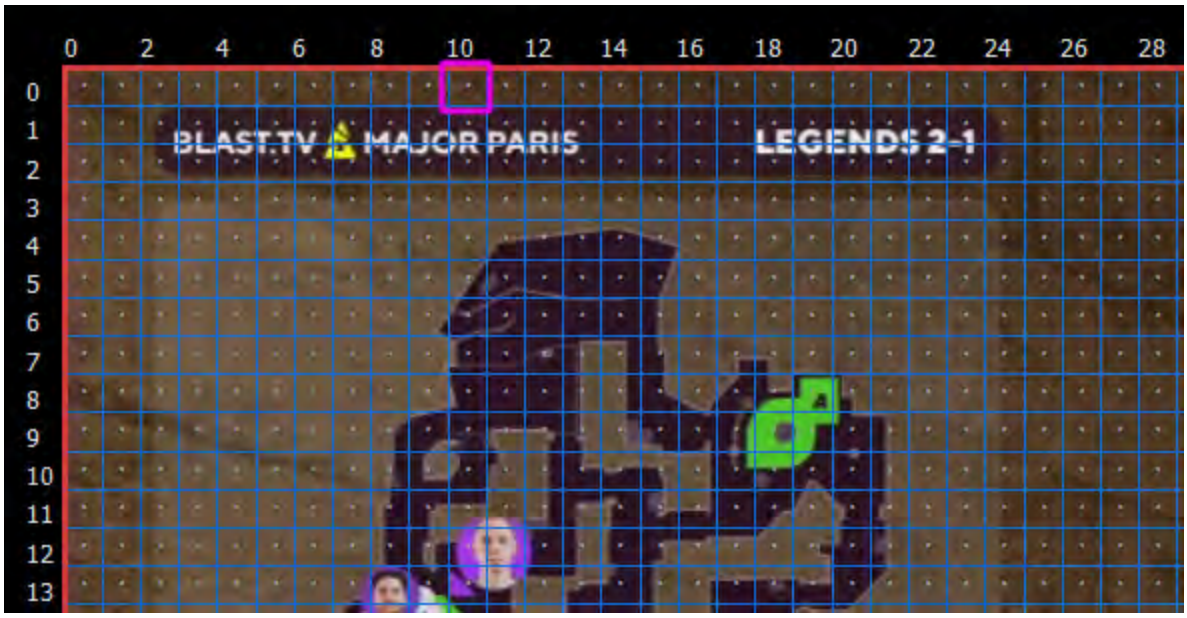


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

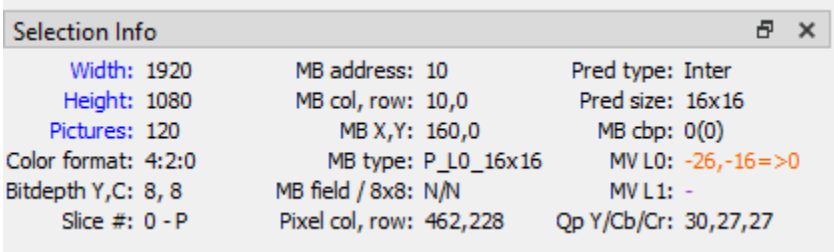


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

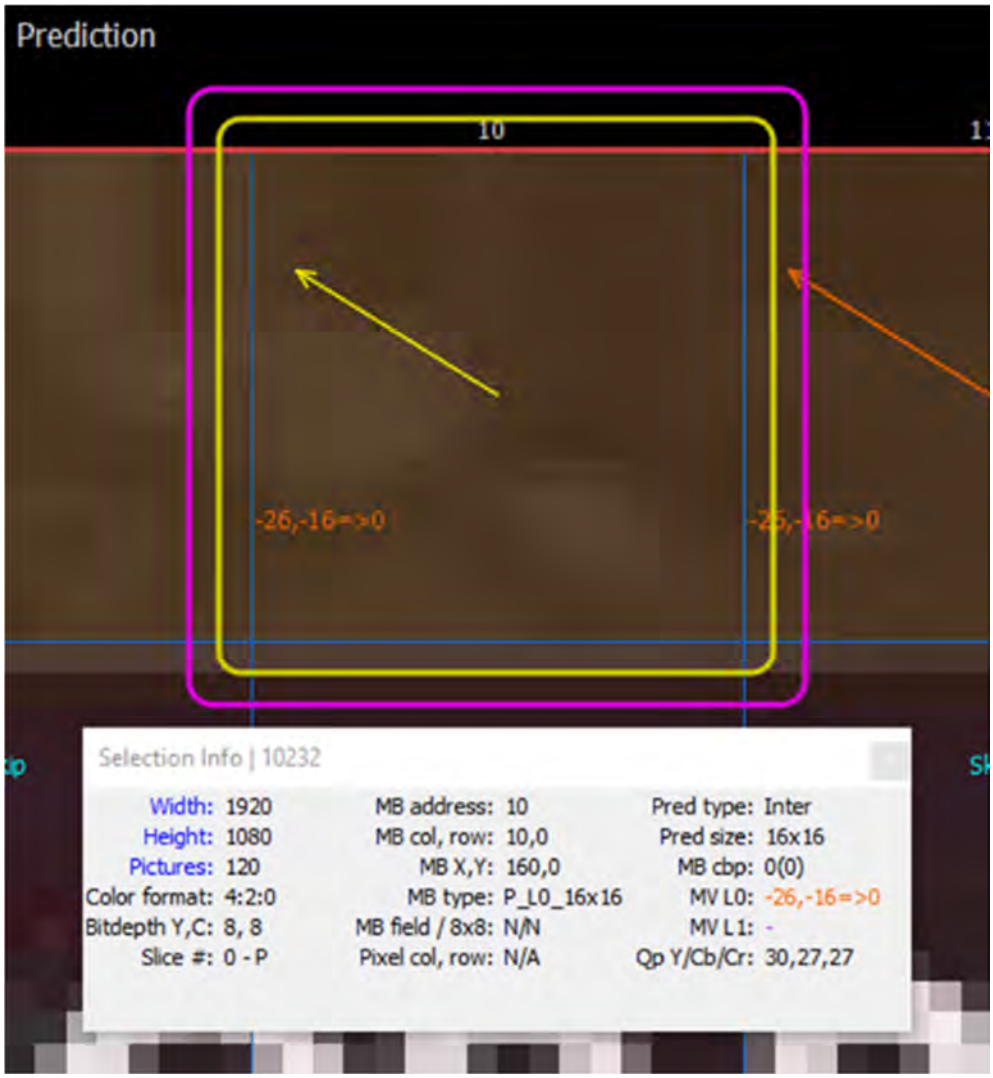
298. When encoding Twitch.tv video, on information and belief, Amazon performs a method of interpolating sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of both K and L, using weighted sums of the values of pixels located in rows and columns respectively, as demonstrated in the screenshots below using VQ Analyzer software, where the -26, -16 motion vector corresponds to $K/2^N$ unit horizontal and unit vertical locations.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

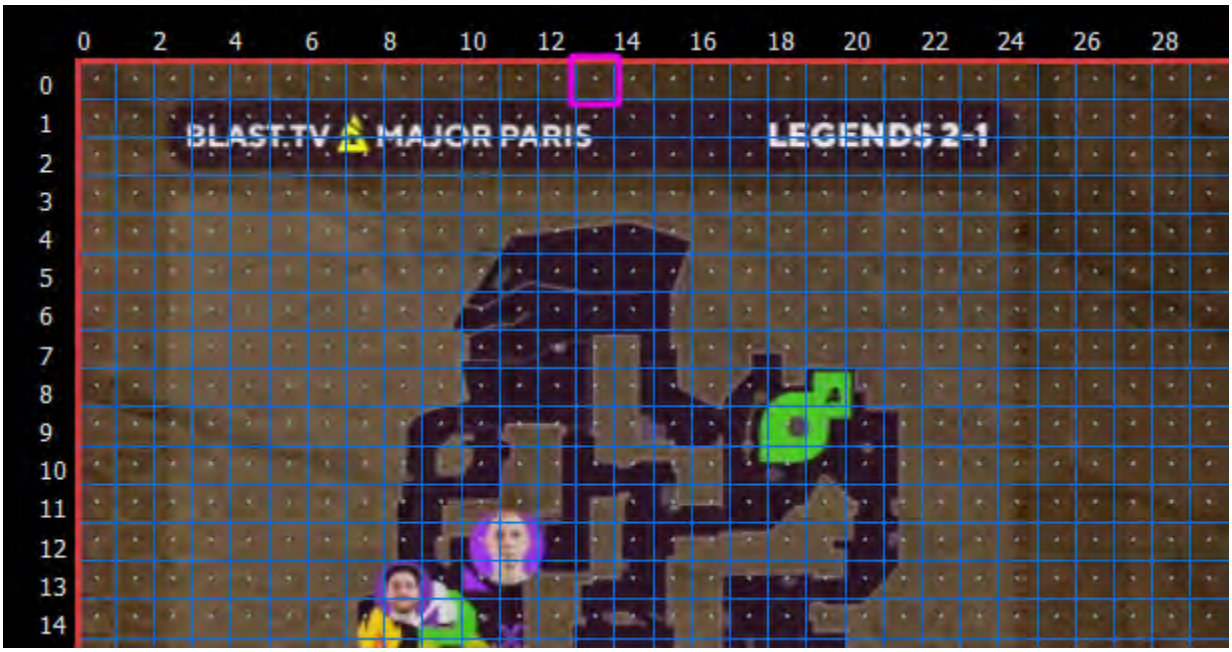


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

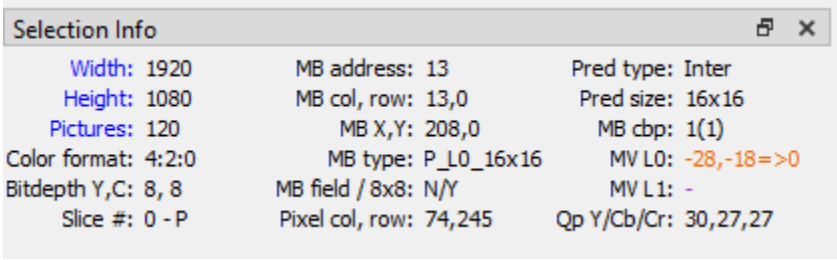


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

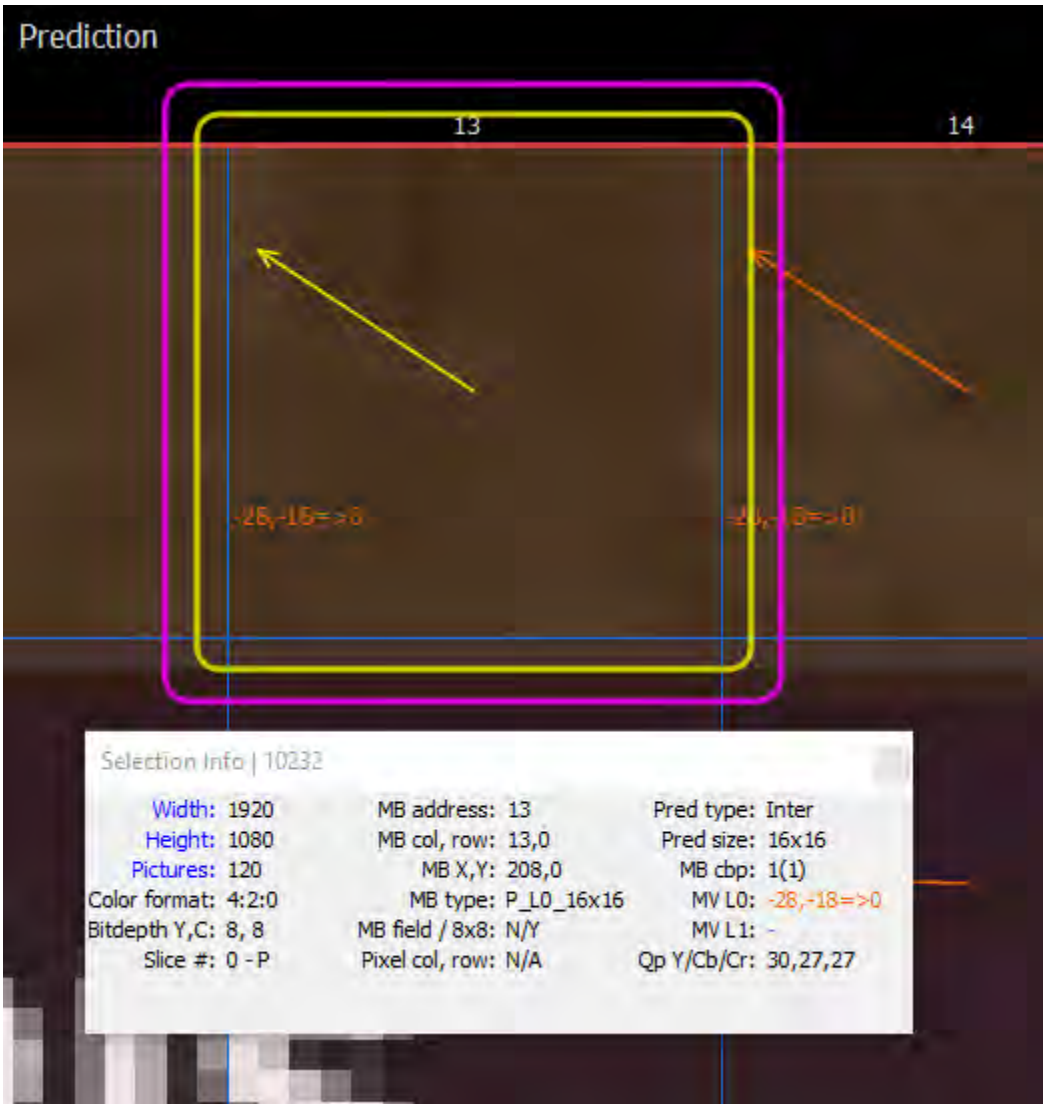
299. When encoding Twitch.tv video, on information and belief, Amazon performs a method of interpolating sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of both K and L, using weighted sums of the values of pixels located in rows and columns respectively, as demonstrated in the screenshots below using VQ Analyzer software, where the -28, -18 motion vector corresponds to unit horizontal and $L/2^N$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

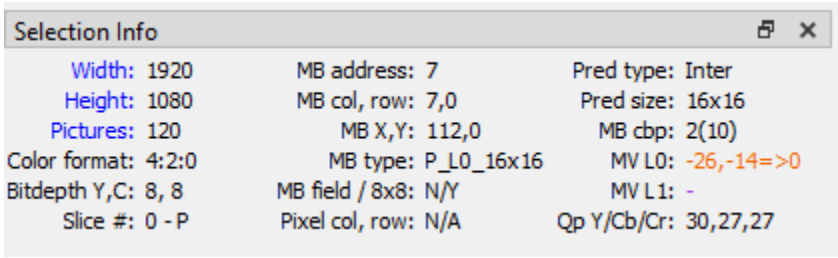


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

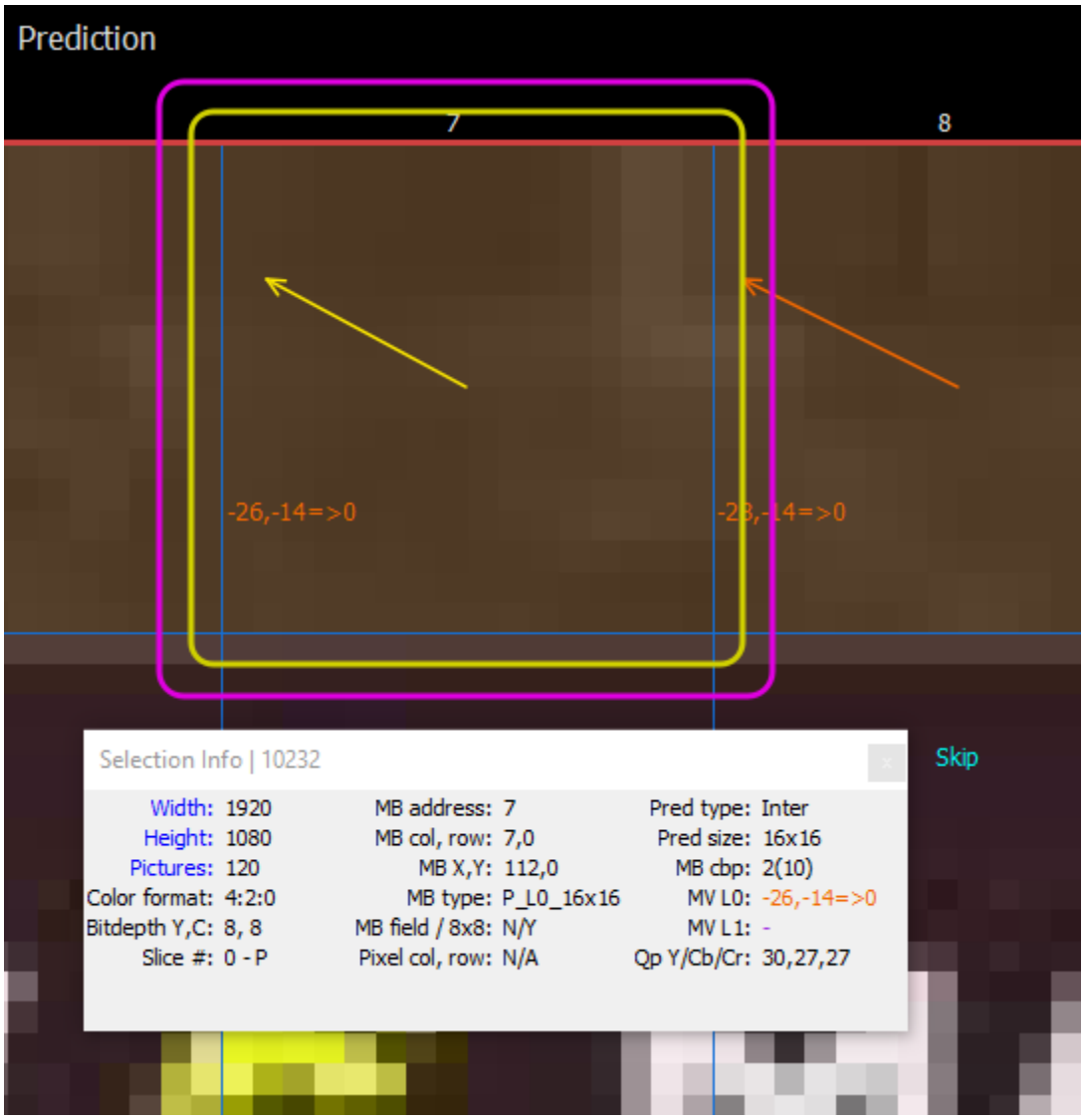
300. When encoding Twitch.tv video, on information and belief, Amazon performs a method of interpolating sub-pixel values for sub-pixels having co-ordinates with even values of both K and L, used in the interpolation of sub-pixel values for the sub-pixels having co-ordinates with odd values of both K and L, using a predetermined choice of either a weighted sum of the values of sub-pixels having co-ordinates with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the values of sub-pixels having co-ordinates with K equal

to zero and L equal to an even value and the values of sub-pixels having corresponding co-ordinates in immediately adjacent bounded rectangular regions, as demonstrated in the screenshots below using VQ Analyzer software, where the -26, -14 motion vector corresponds to $K/2^N$ unit horizontal and $L/2^N$ unit vertical locations.

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

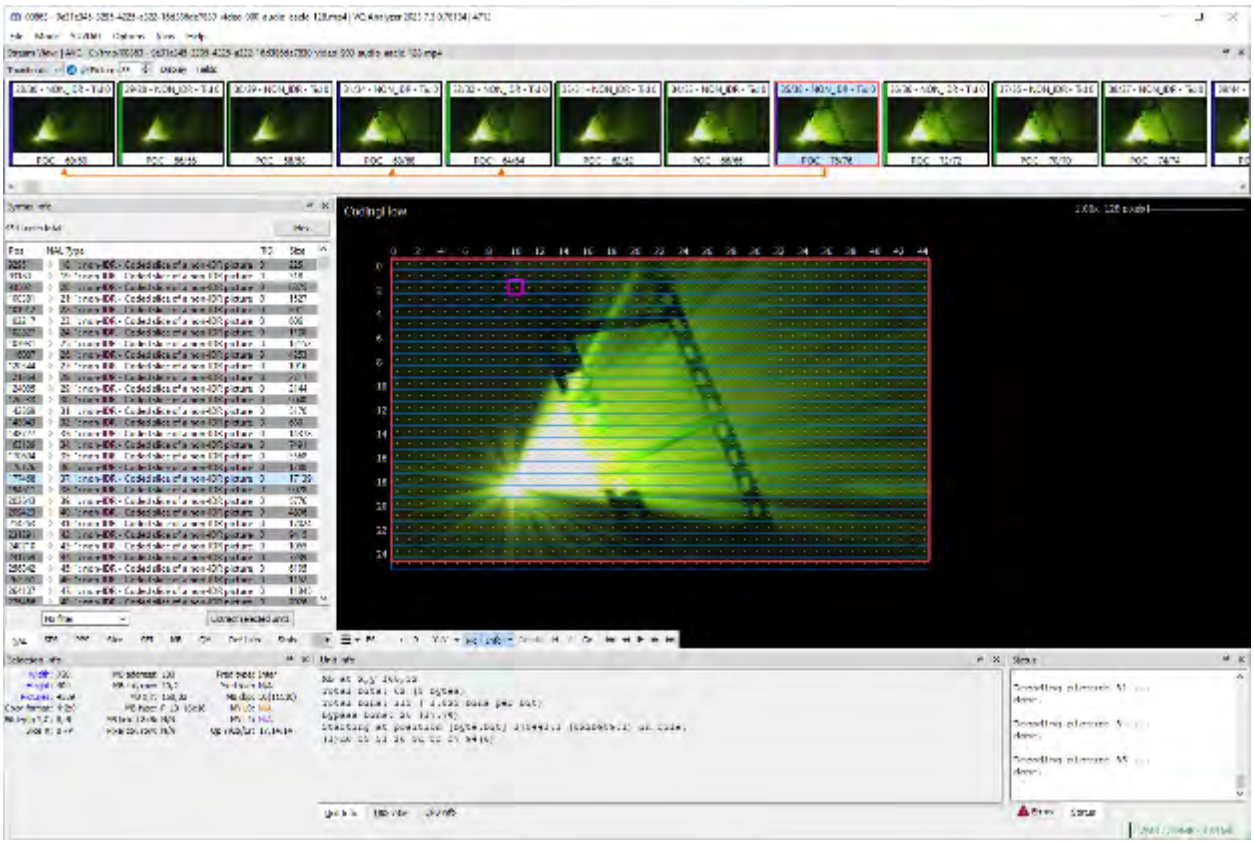
F. Amazon Makes, Imports, Uses, Sells, and/or Offers for Sale Products and Services that Infringe the ’273 Patent

301. The Accused Products infringe one or more claims of the ’273 Patent, including, for example, claim 1.

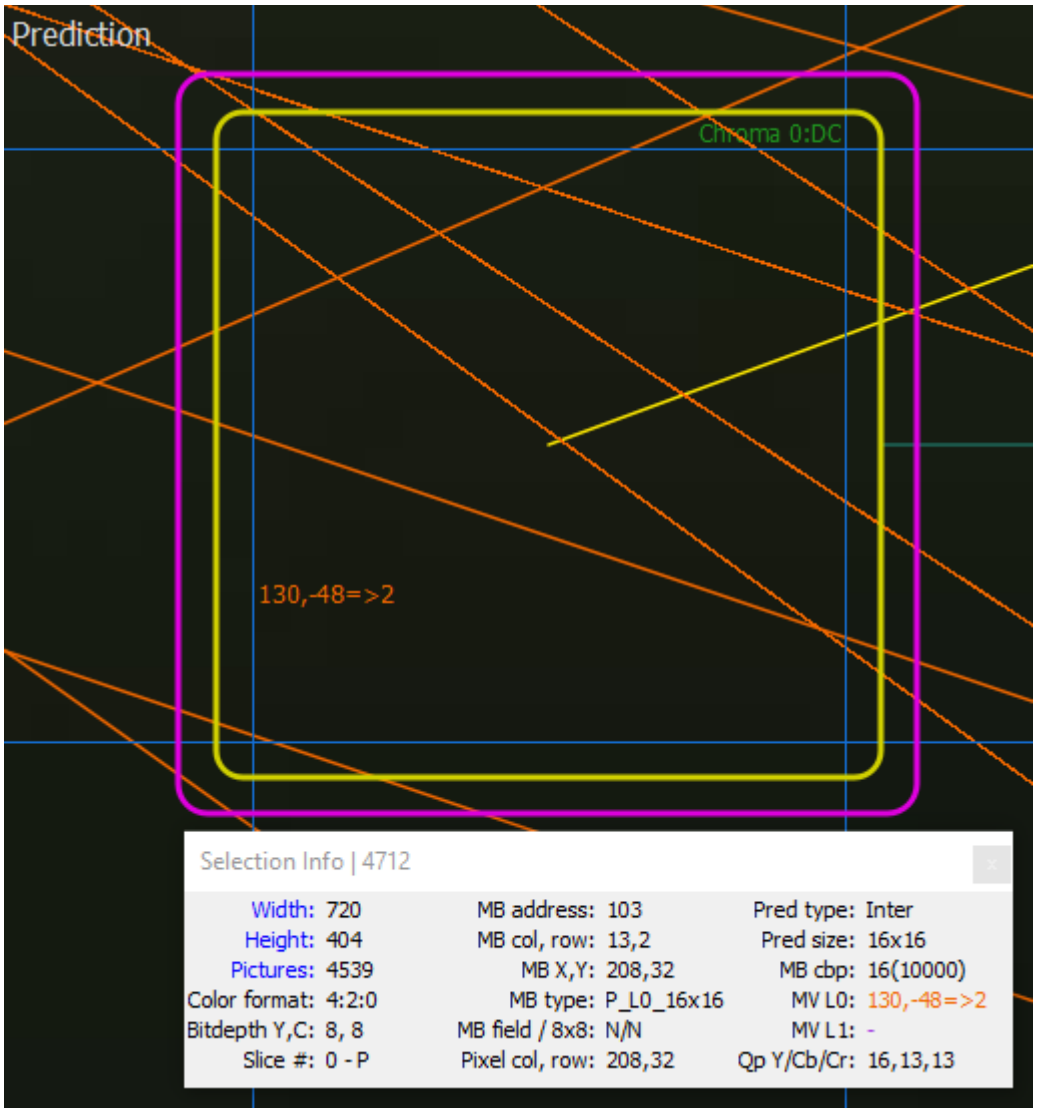
302. As just one example of infringement, on information and belief, Amazon performs a method of sub-pixel value interpolation in video coding in a manner that is covered by claim 1 of the ’273 Patent for Amazon Prime Video content (such as trailers), as demonstrated in the

screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 212.

303. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs sub-pixel value interpolation to determine values for sub-pixels situated within a rectangular bounded region defined by four corner pixels with no intermediate pixels between the corners, the pixels and sub-pixels being configured for display in rows and columns, pixel and sub-pixel locations in the rows and columns being representable mathematically within the rectangular bounded region using the co-ordinate notation $K/2^N$, $L/2^N$, K and L being positive integers having respective values between zero and 2^N, N being a positive integer greater than one and representing a particular degree of sub-pixel value interpolation, as demonstrated in the screenshots below using VQ Analyzer software.



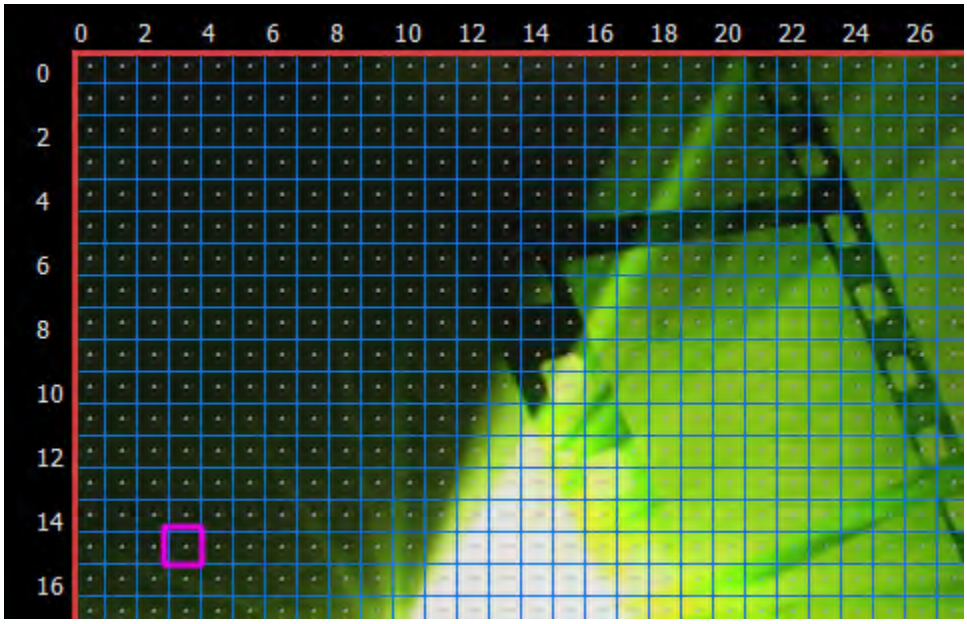
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



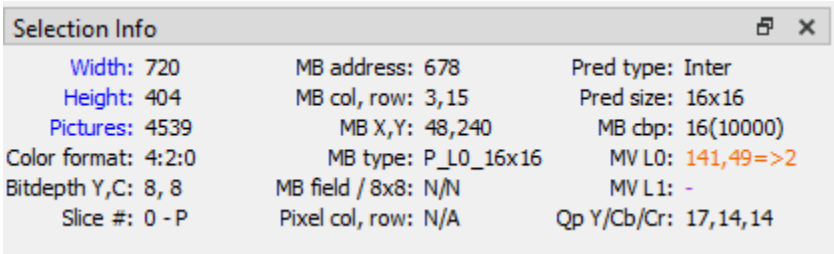
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

304. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs a method to interpolate a sub-pixel value for a sub-pixel having co-ordinates with odd values of K and L, according to a predetermined choice of either a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates ½, ½, or a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of K and L, including zero, situated within a quadrant of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates ½, ½ and the nearest

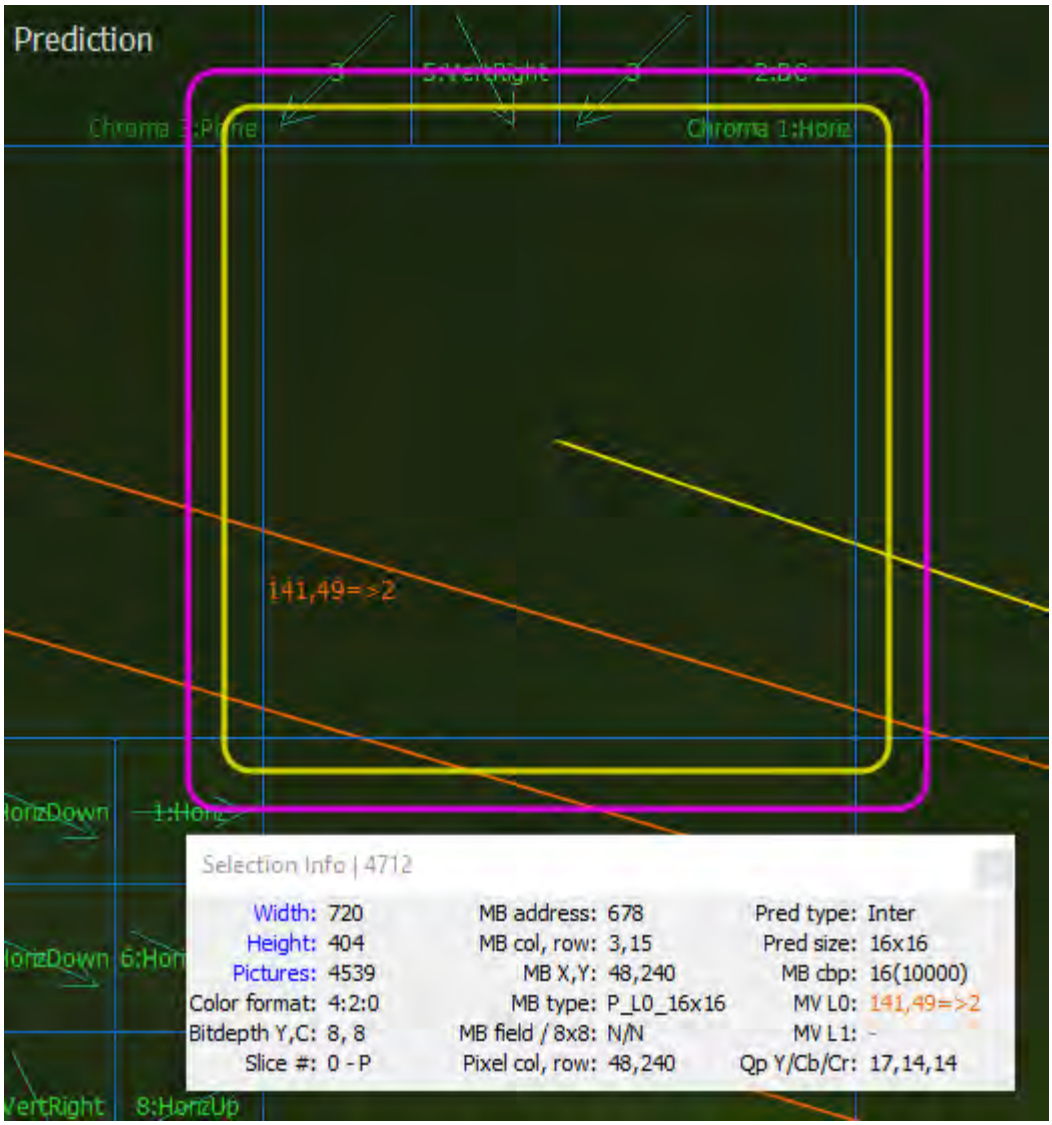
neighbouring pixel, as demonstrated in the screenshots below using VQ Analyzer software, where the 141, 49 motion vector corresponds to $K/2^N$ unit horizontal and $L/2^N$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

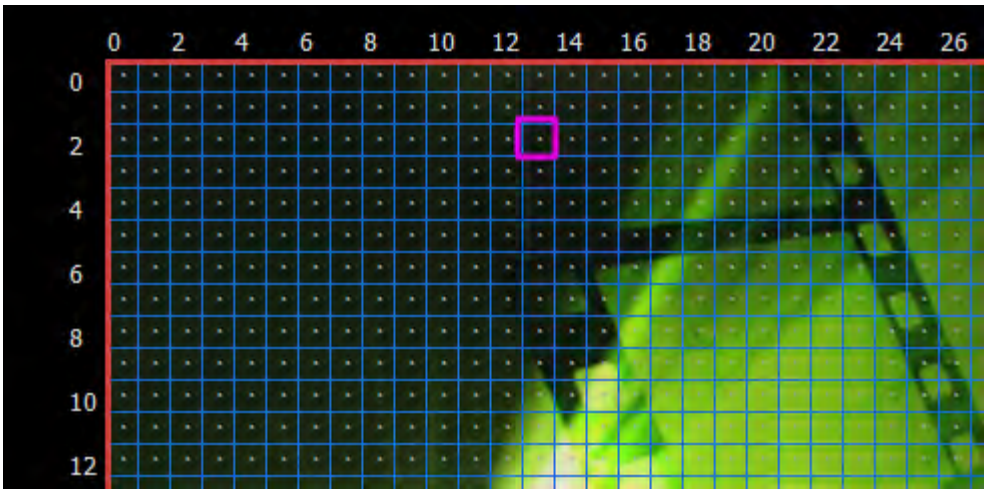


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

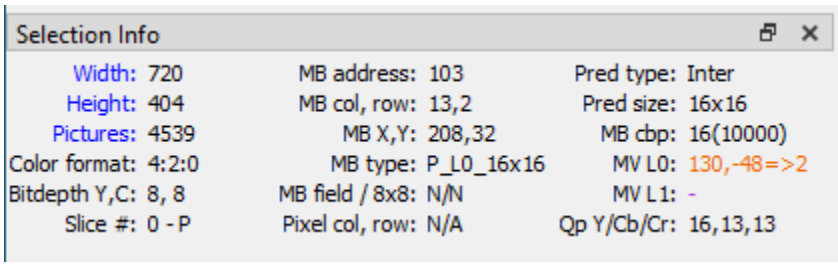


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

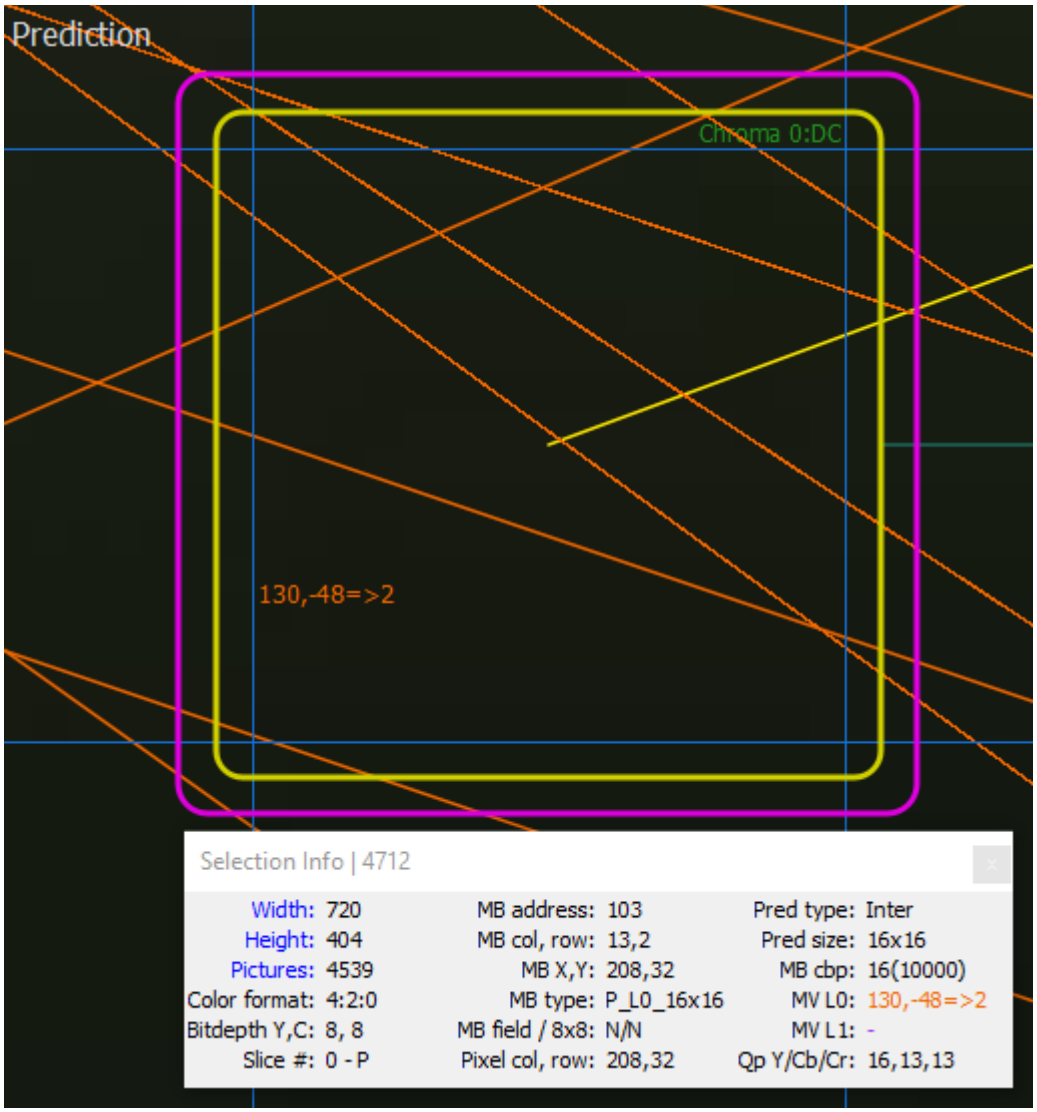
305. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs a method to interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal to zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of K and L, using weighted sums of the values of pixels located in rows and columns, as demonstrated in the screenshots below using VQ Analyzer software, where the 130, -48 motion vector corresponds to $K/2^N$ unit horizontal and unit vertical locations.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

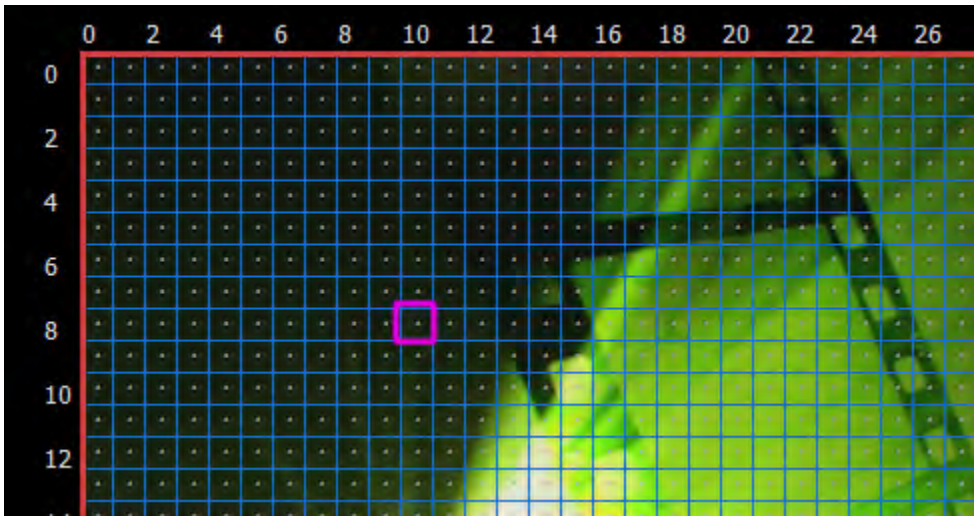


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

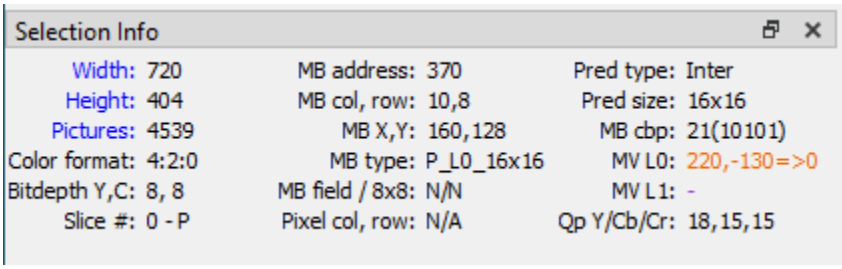


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

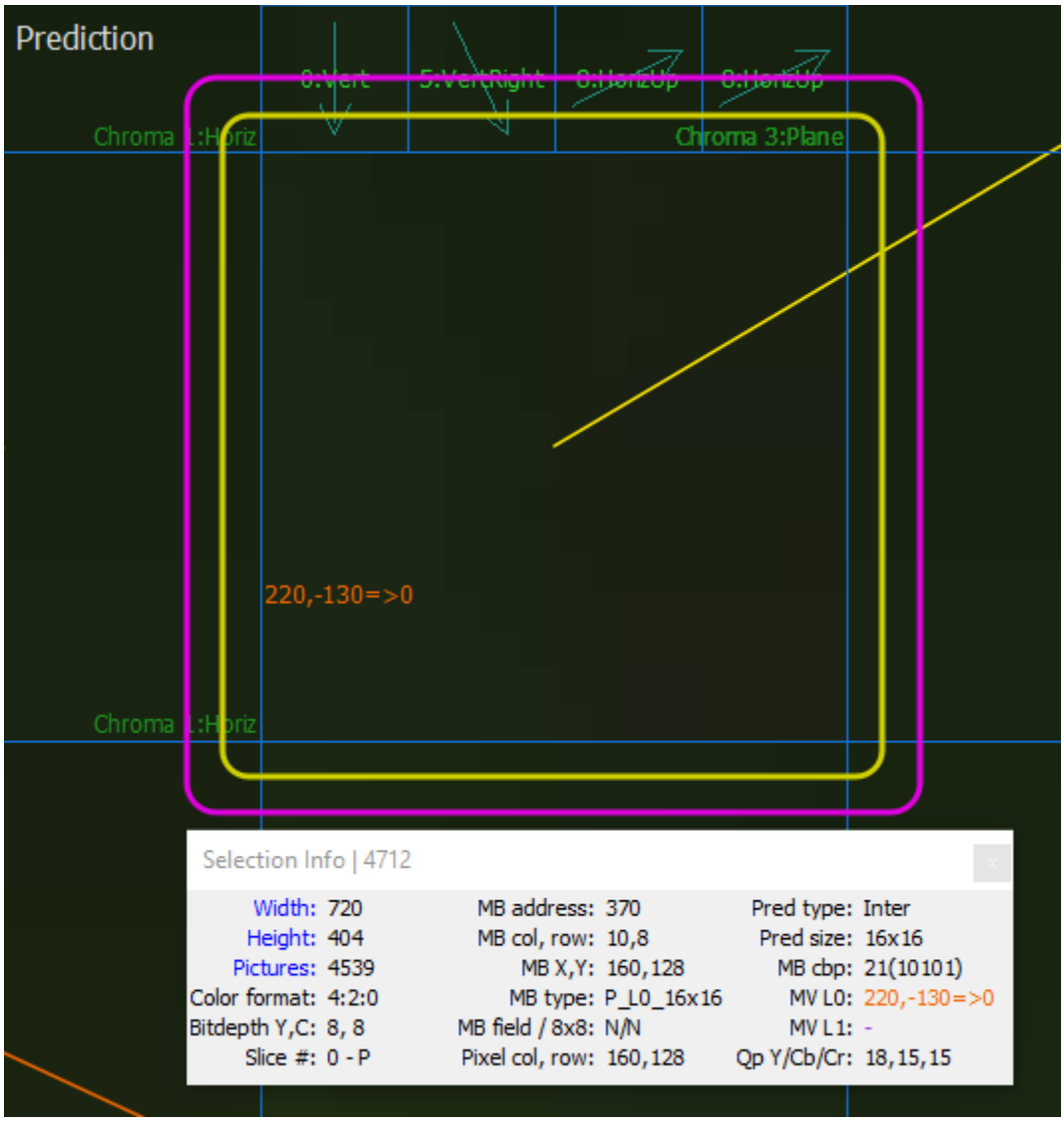
306. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs a method to interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal to zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of K and L, using weighted sums of the values of pixels located in rows and columns, as demonstrated in the screenshots below using VQ Analyzer software, where the 220, -130 motion vector corresponds to unit horizontal and $L/2^N$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



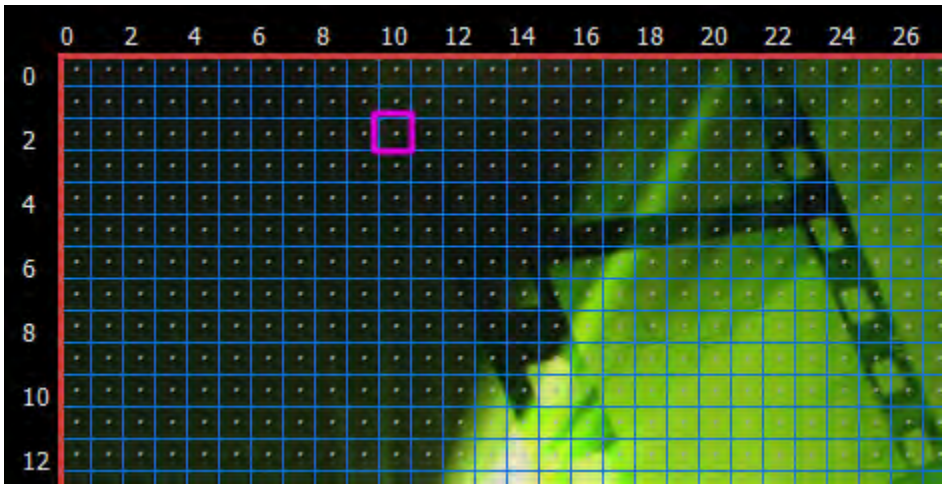
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



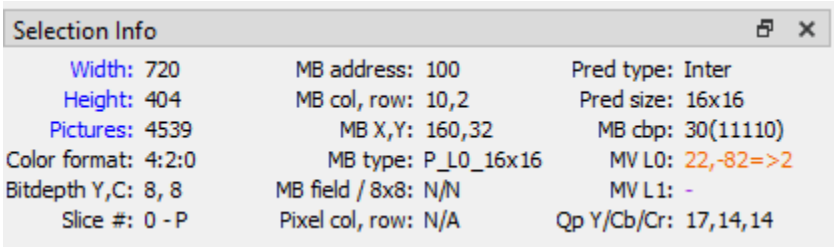
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

307. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs a method to interpolate sub-pixel values for sub-pixels having co-ordinates with even values of K and L, used in the interpolation of sub-pixel values for the sub-pixels having co-ordinates with odd values of K and L, using a predetermined choice of either a weighted sum of the values of sub-pixels having co-ordinates with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the values of sub-pixels having co-ordinates

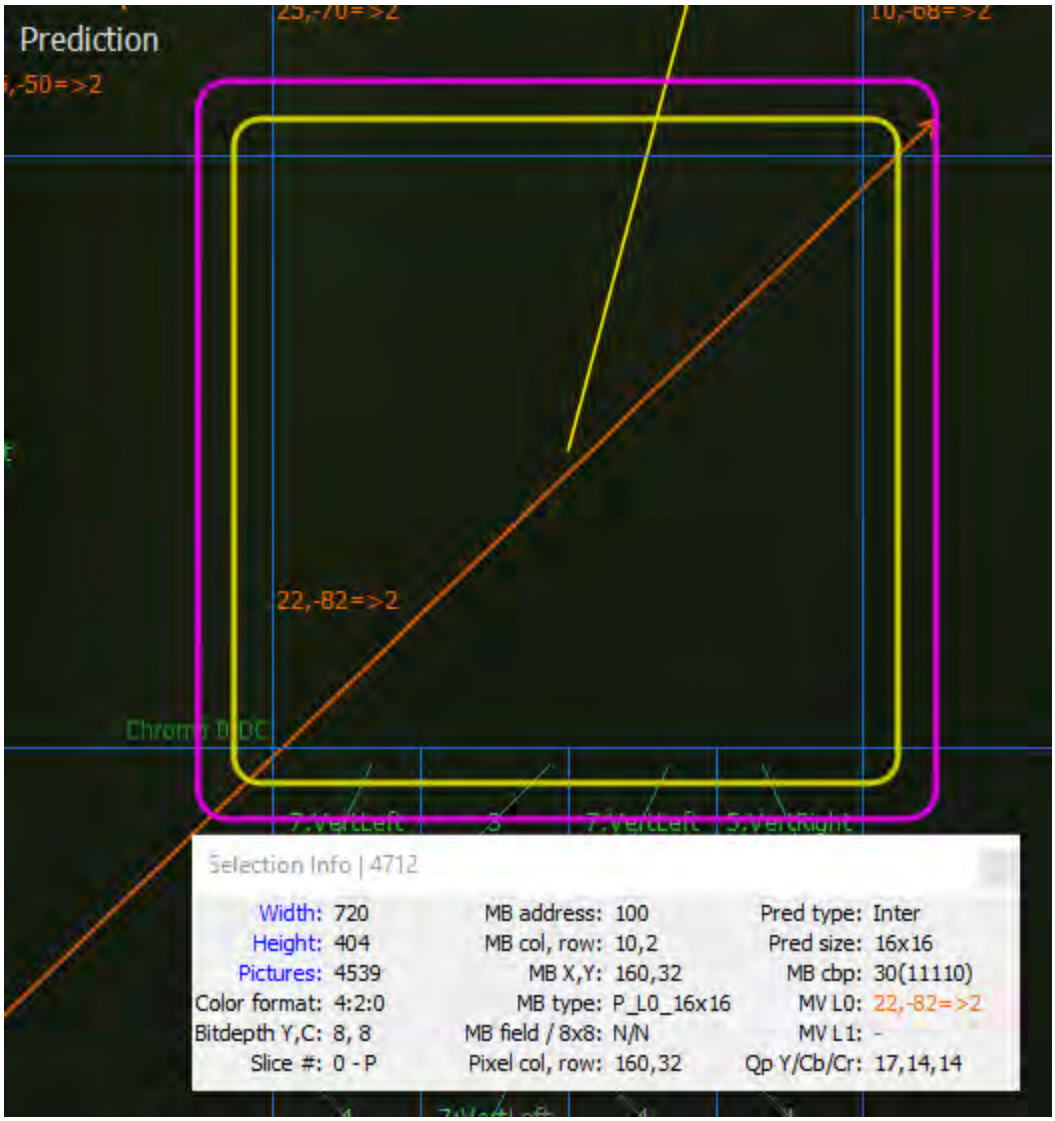
with K equal to zero and L equal to an even value and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, as demonstrated in the screenshots below using VQ Analyzer software, where the 22, -82 motion vector corresponds to $K/2^N$ unit horizontal and $L/2^N$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

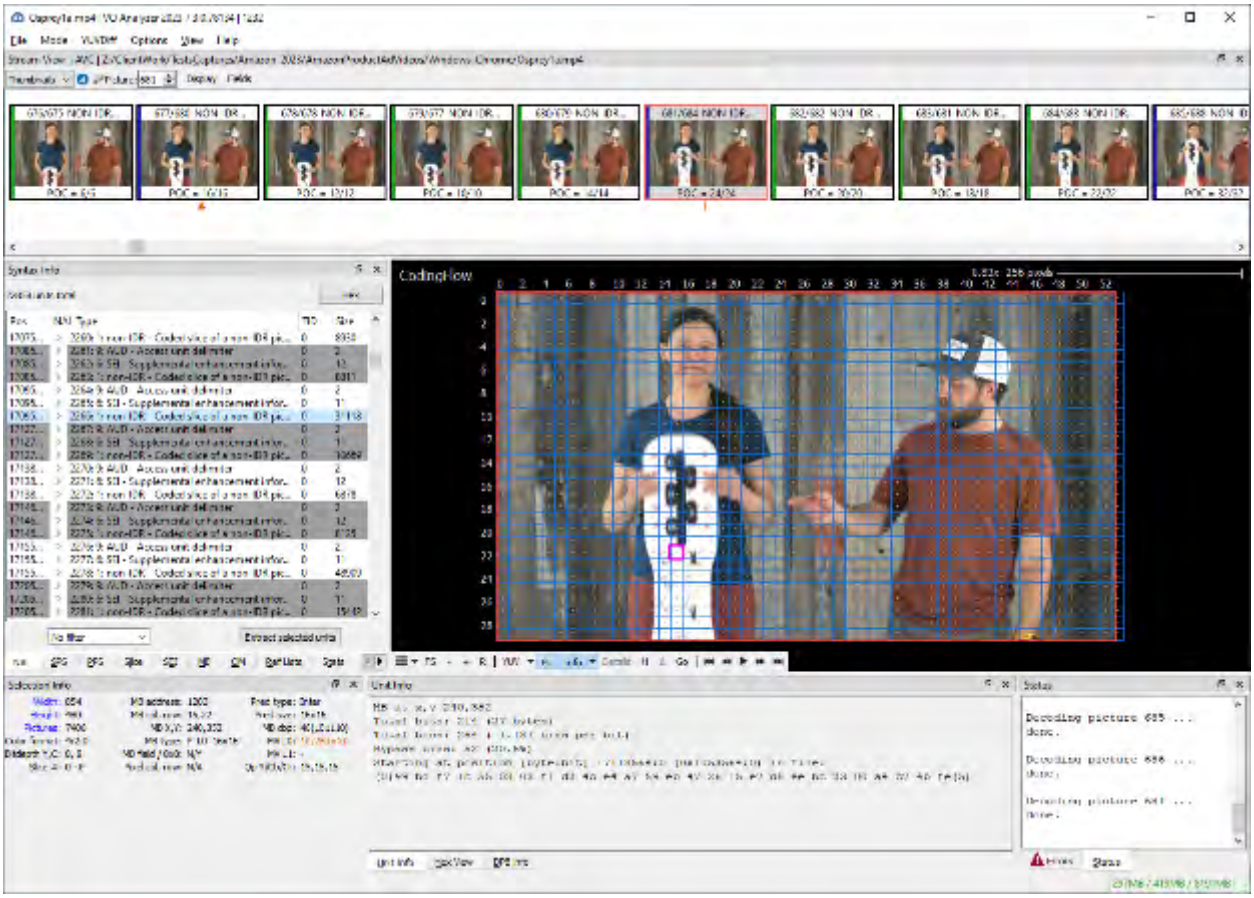


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

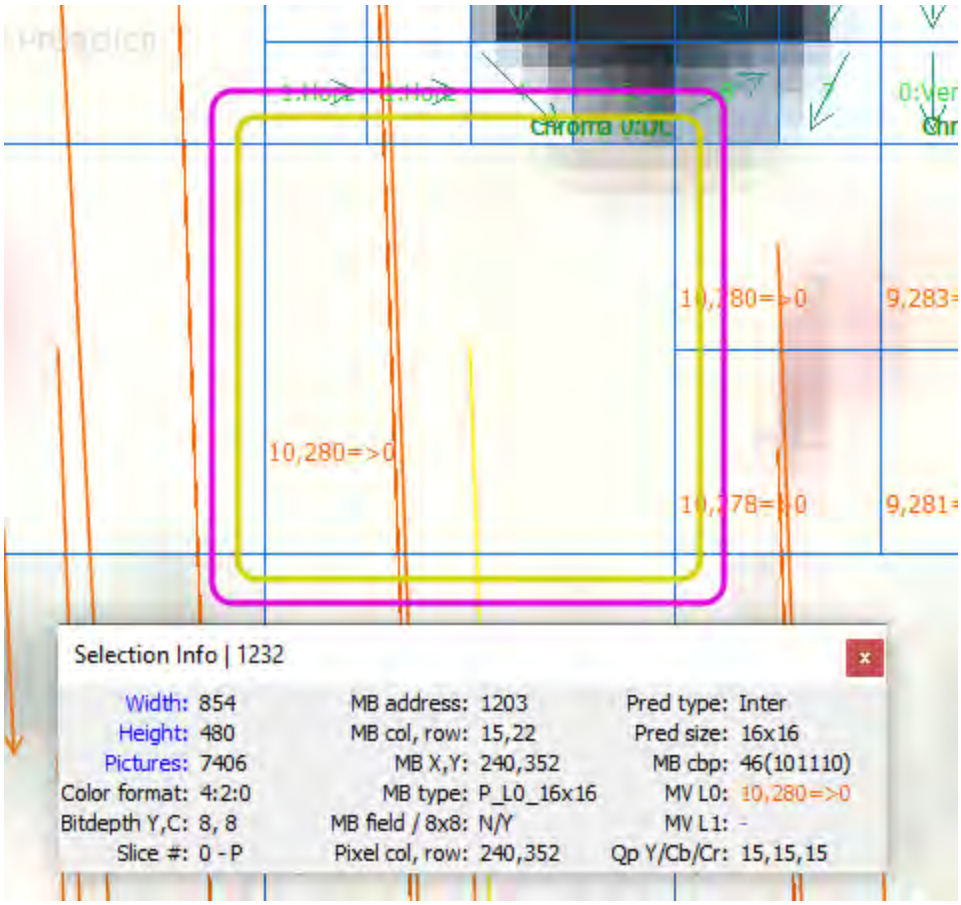
308. For another example, on information and belief, Amazon performs a method of sub-pixel value interpolation in video coding in a manner that is covered by claim 1 of the ’273 Patent for Amazon.com advertisements, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 217.

309. When encoding Amazon.com advertisements, on information and belief, Amazon performs sub-pixel value interpolation to determine values for sub-pixels situated within a

rectangular bounded region defined by four corner pixels with no intermediate pixels between the corners, the pixels and sub-pixels being configured for display in rows and columns, pixel and sub-pixel locations in the rows and columns being representable mathematically within the rectangular bounded region using the co-ordinate notation $K/2^N$, $L/2^N$, K and L being positive integers having respective values between zero and 2^N, N being a positive integer greater than one and representing a particular degree of sub-pixel value interpolation, as demonstrated in the screenshots below using VQ Analyzer software.

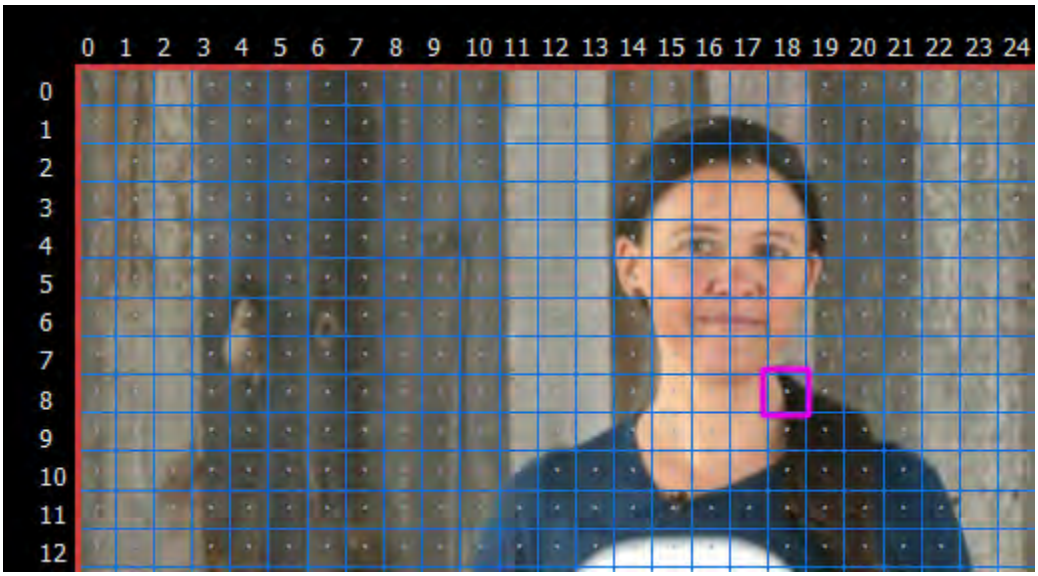


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

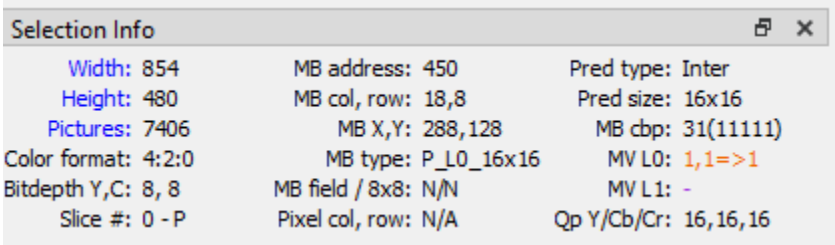


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

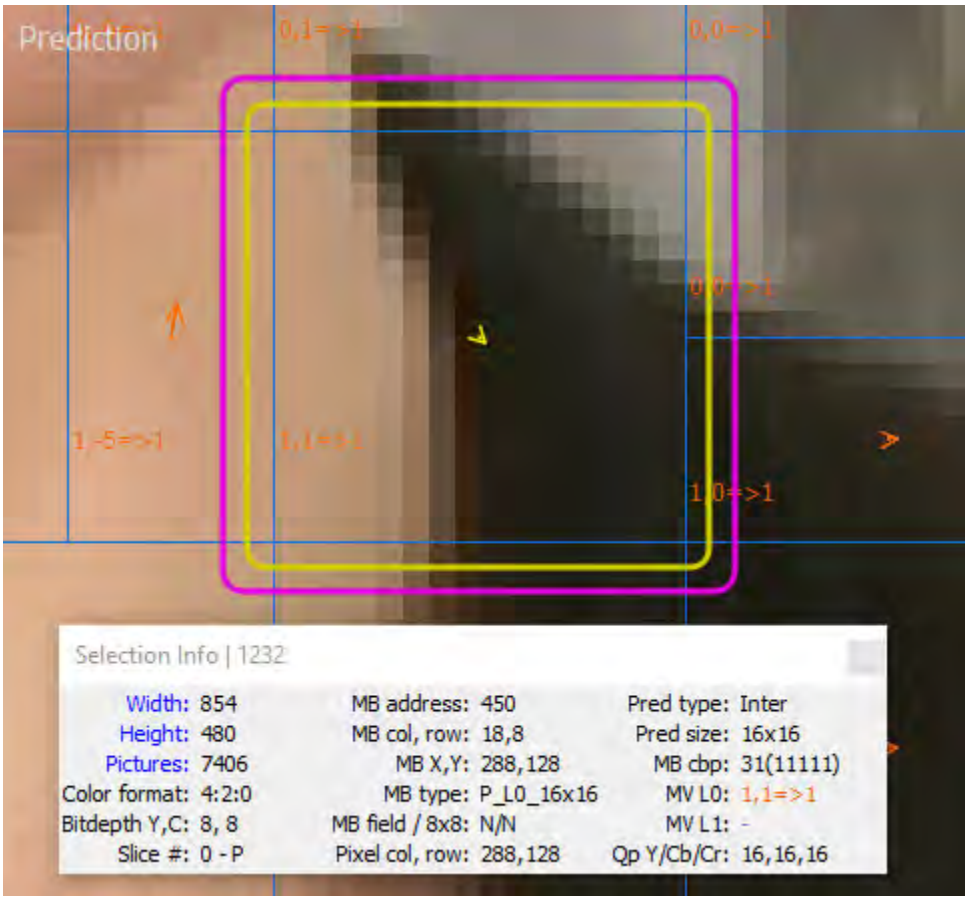
310. When encoding Amazon.com advertisements, on information and belief, Amazon performs a method to interpolate a sub-pixel value for a sub-pixel having co-ordinates with odd values of K and L, according to a predetermined choice of either a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates ½, ½, or a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of K and L, including zero, situated within a quadrant of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates ½, ½ and the nearest neighbouring pixel, as demonstrated in the screenshots below using VQ Analyzer software, where the 1, 1 motion vector corresponds to $K/2^N$ unit horizontal and $L/2^N$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

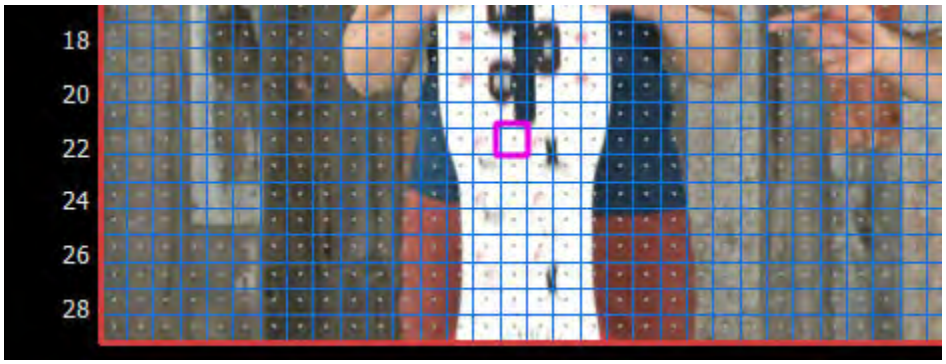


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

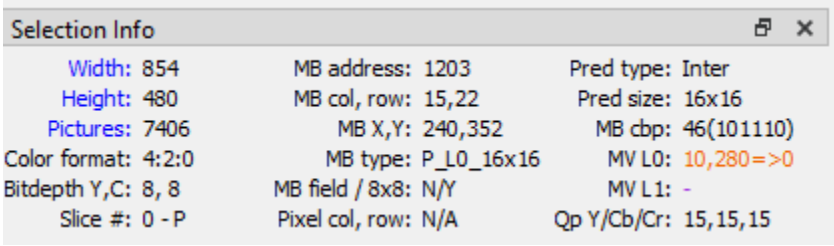


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

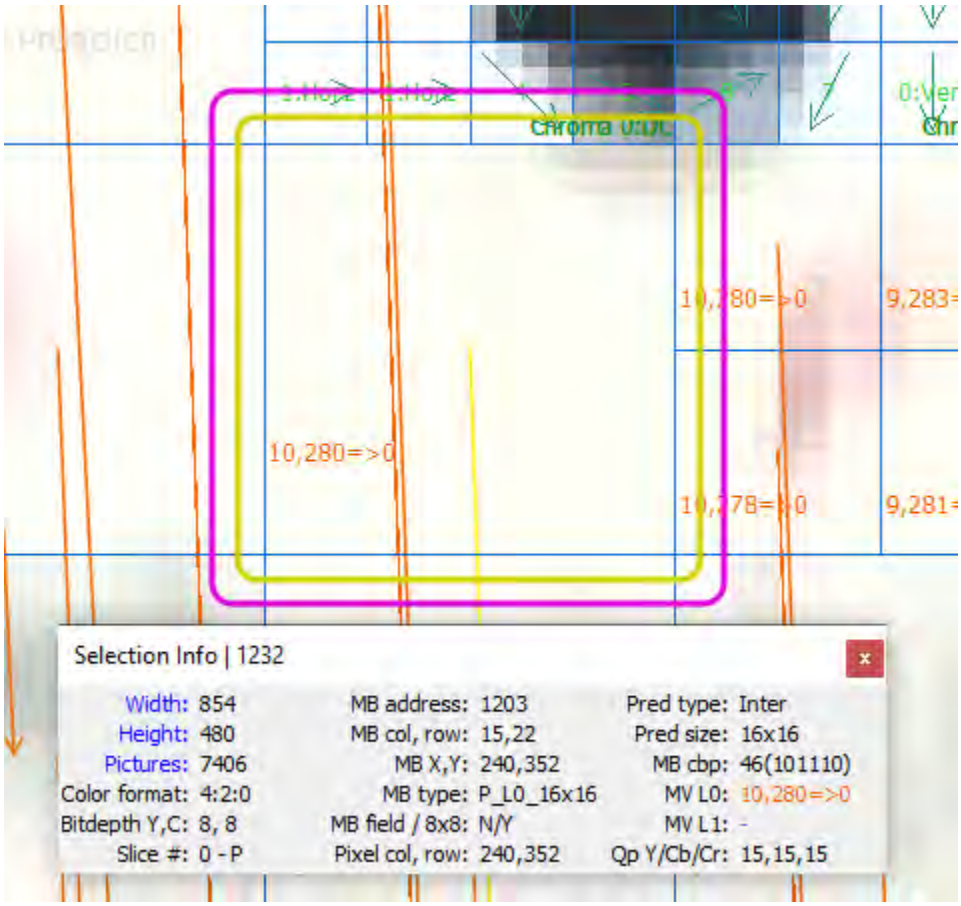
311. When encoding Amazon.com advertisements, on information and belief, Amazon performs a method to interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal to zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of K and L, using weighted sums of the values of pixels located in rows and columns, as demonstrated in the screenshots below using VQ Analyzer software, where the 10, 280 motion vector corresponds to $K/2^N$ unit horizontal and unit vertical locations.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

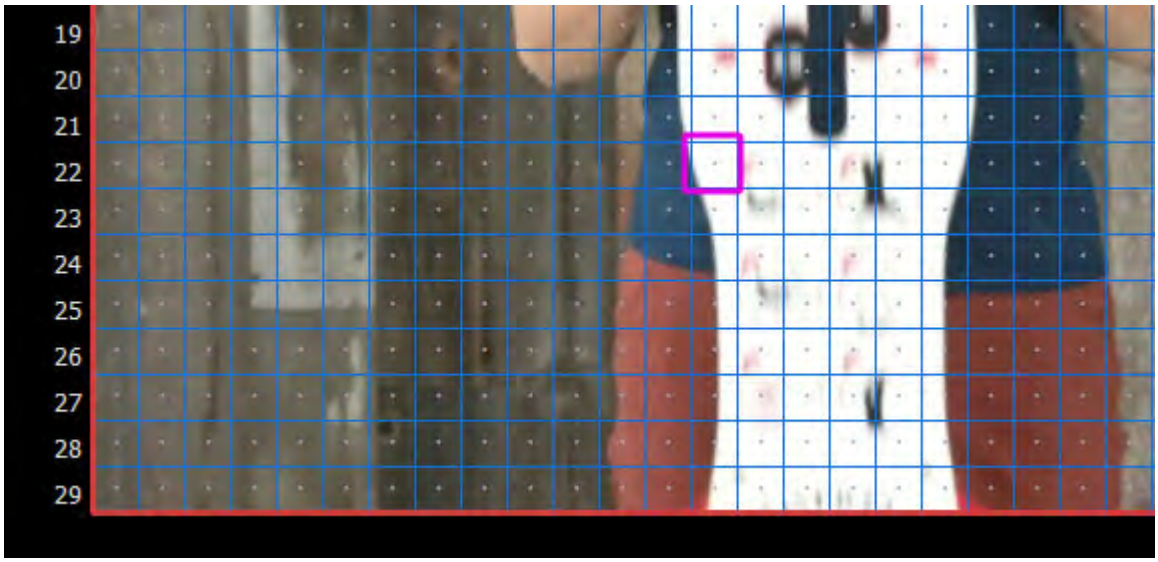


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

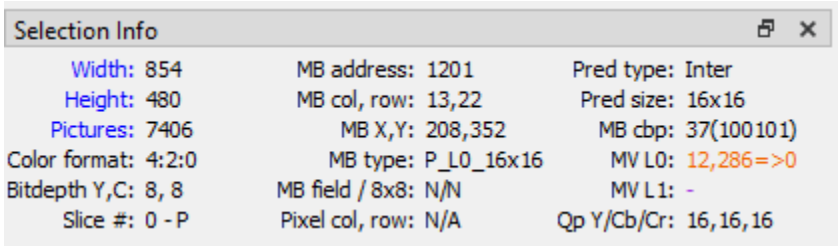


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

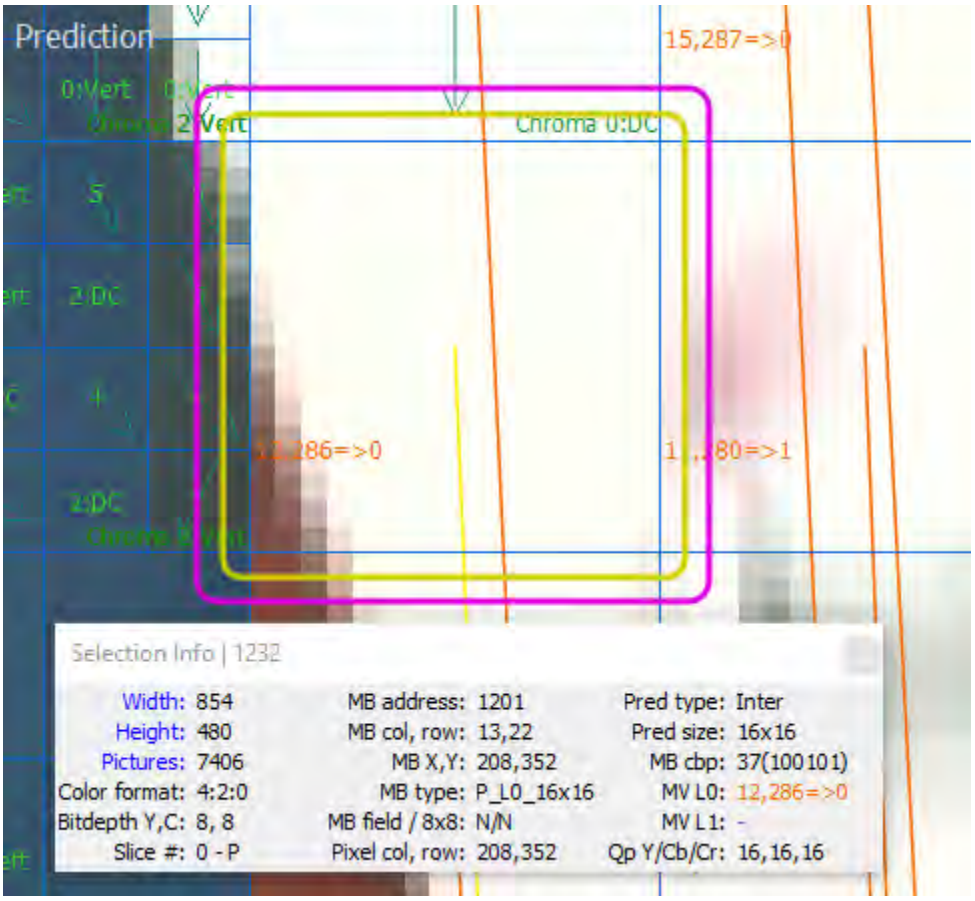
312. When encoding Amazon.com advertisements, on information and belief, Amazon performs a method to interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal to zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of K and L, using weighted sums of the values of pixels located in rows and columns, as demonstrated in the screenshots below using VQ Analyzer software, where the 12, 286 motion vector corresponds to unit horizontal and $L/2^N$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



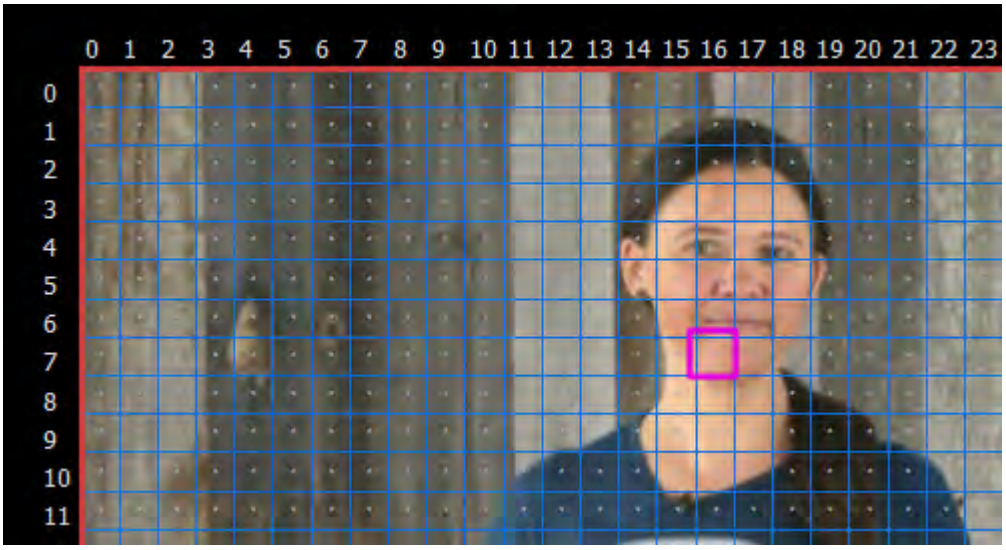
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



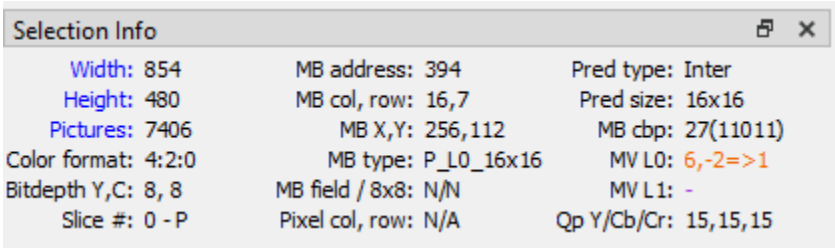
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

313. When encoding Amazon.com advertisements, on information and belief, Amazon performs a method to interpolate sub-pixel values for sub-pixels having co-ordinates with even values of K and L, used in the interpolation of sub-pixel values for the sub-pixels having co-ordinates with odd values of K and L, using a predetermined choice of either a weighted sum of the values of sub-pixels having co-ordinates with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the values of sub-pixels having co-ordinates with K equal to zero and L equal to an even value and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, as

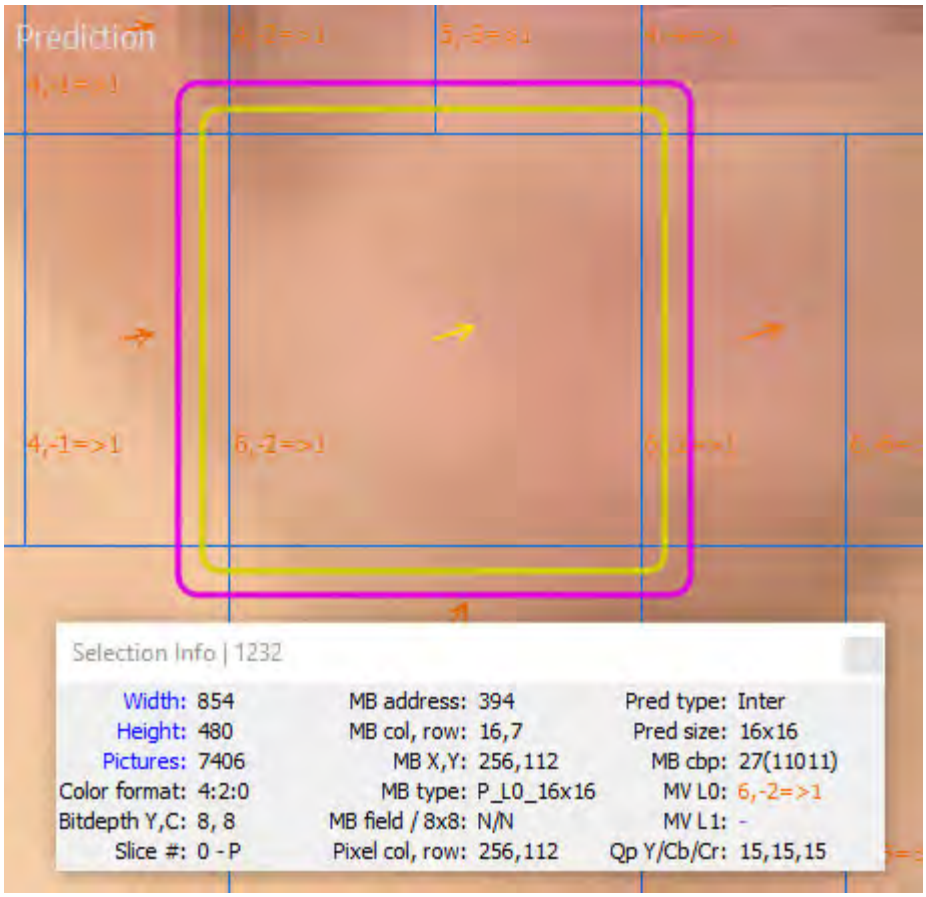
demonstrated in the screenshots below using VQ Analyzer software, where the 6, -2 motion vector corresponds to $K/2^N$ unit horizontal and $L/2^N$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

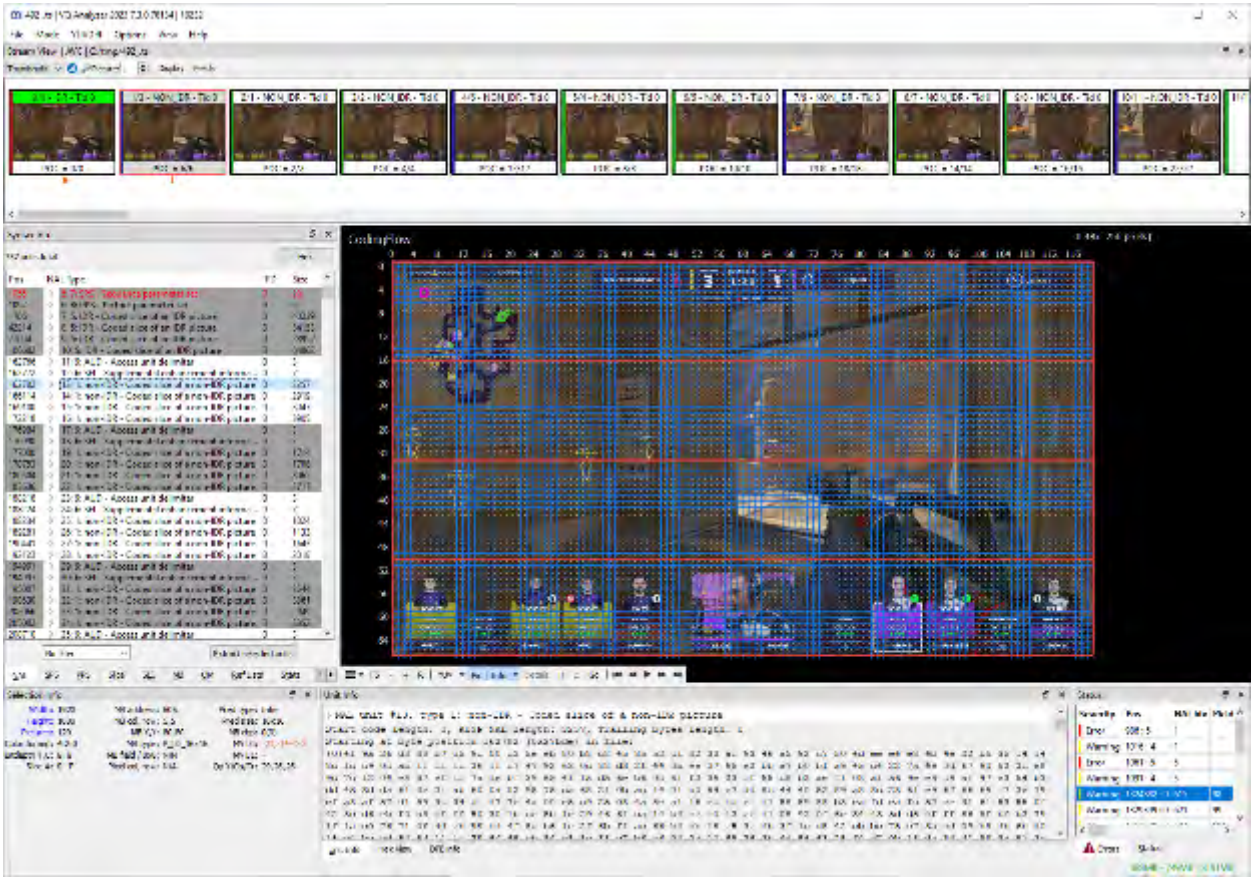


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

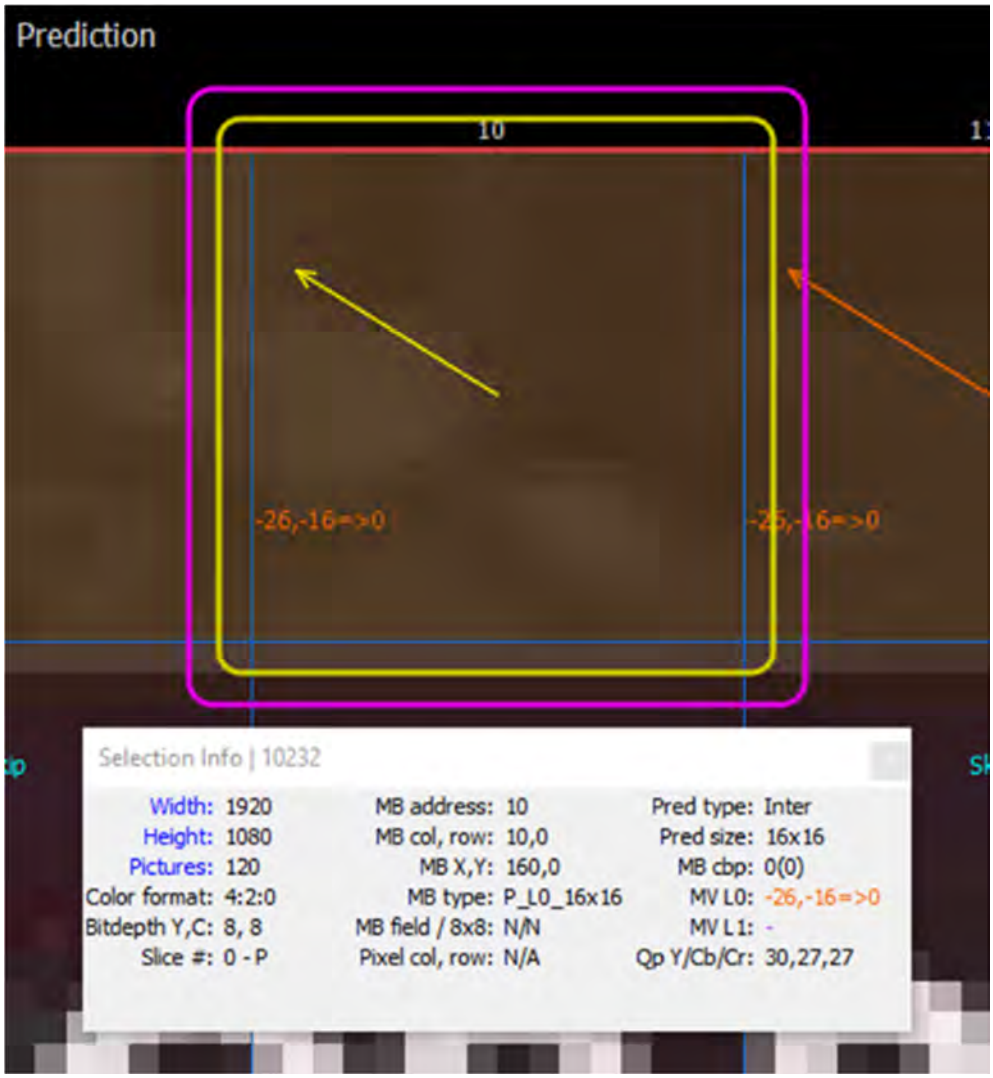
314. For another example, on information and belief, Amazon performs a method of sub-pixel value interpolation in video coding in a manner that is covered by claim 1 of the ’273 Patent for Twitch.tv video, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 222.

315. When encoding Twitch.tv video, on information and belief, Amazon performs sub-pixel value interpolation to determine values for sub-pixels situated within a rectangular bounded region defined by four corner pixels with no intermediate pixels between the corners, the pixels and sub-pixels being configured for display in rows and columns, pixel and sub-pixel locations in the rows and columns being representable mathematically within the rectangular bounded

region using the co-ordinate notation $K/2^N$, $L/2^N$, K and L being positive integers having respective values between zero and 2^N, N being a positive integer greater than one and representing a particular degree of sub-pixel value interpolation, as demonstrated in the screenshots below using VQ Analyzer software.



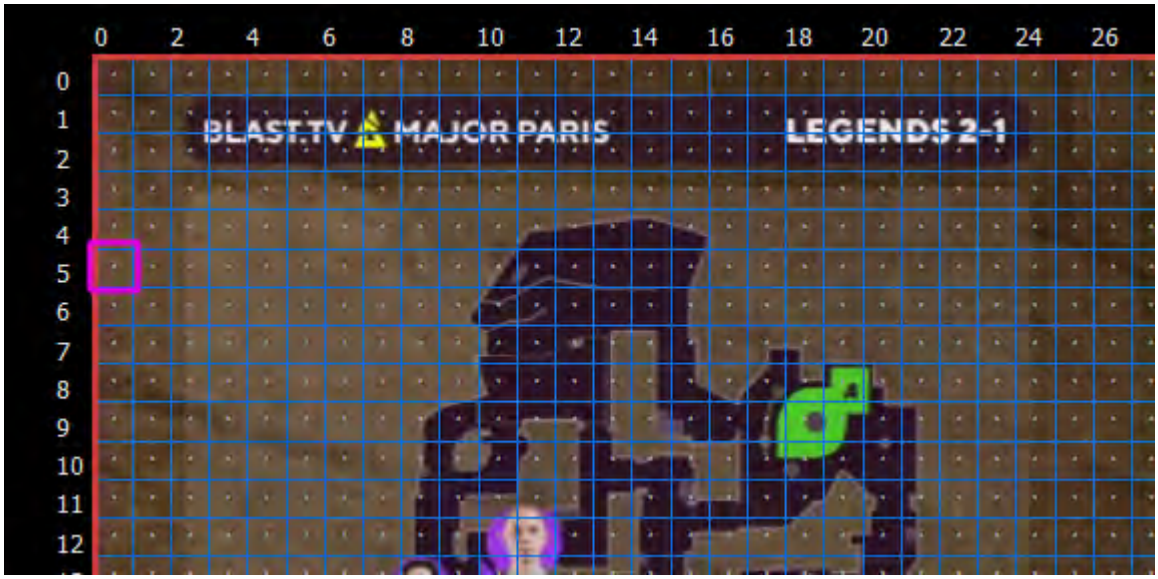
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



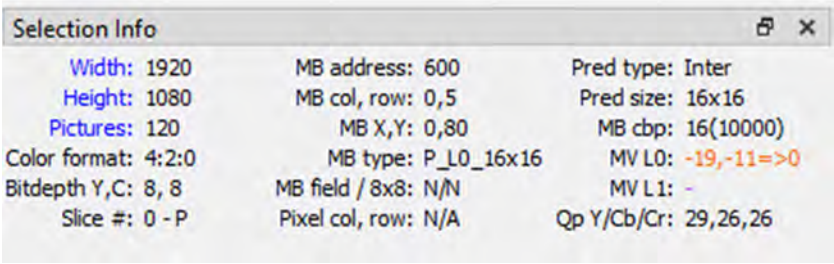
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

316. When encoding Twitch.tv video, on information and belief, Amazon performs a method to interpolate a sub-pixel value for a sub-pixel having co-ordinates with odd values of K and L, according to a predetermined choice of either a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates ½, ½, or a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of K and L, including zero, situated within a quadrant of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates ½, ½ and the nearest

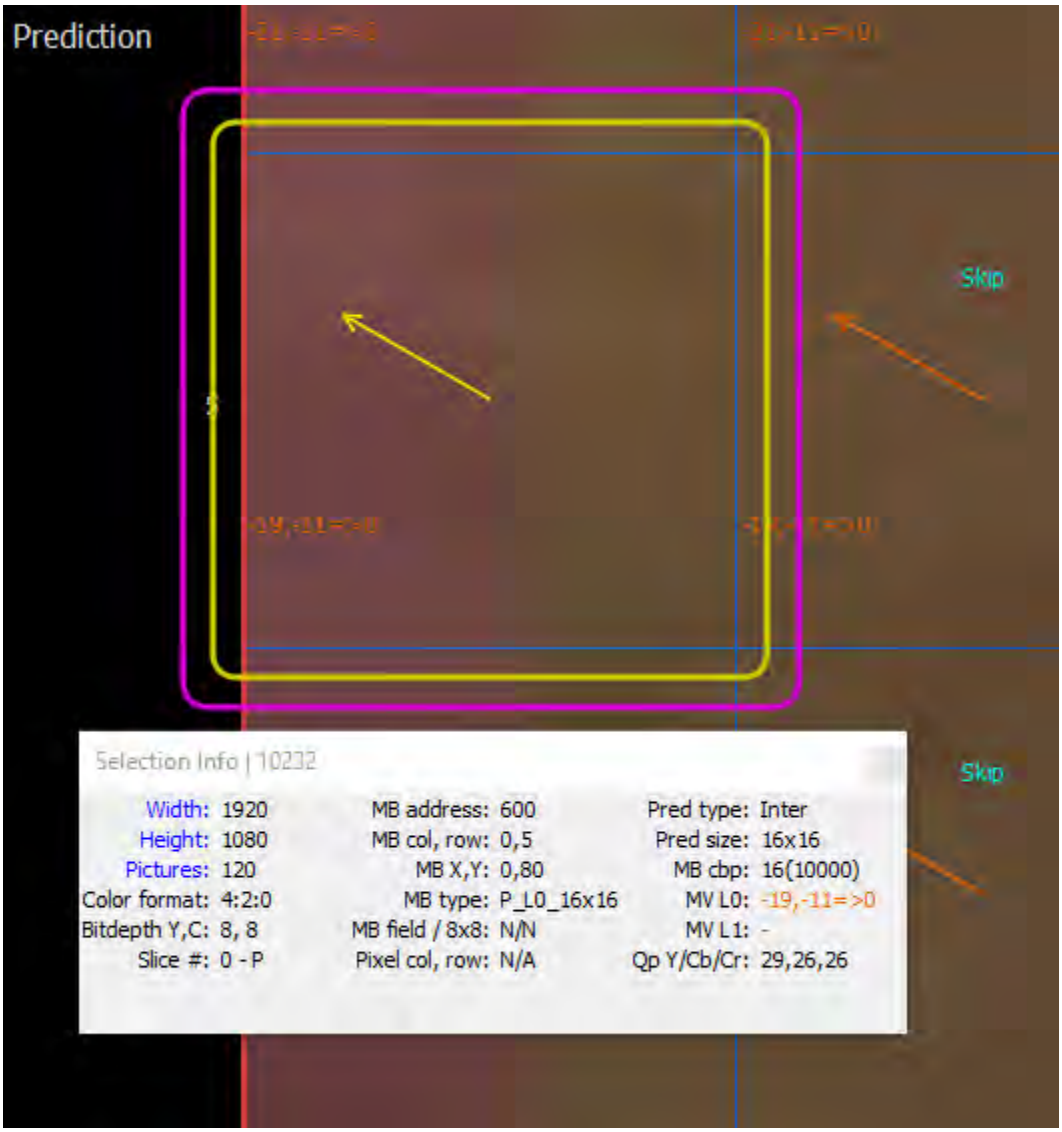
neighbouring pixel, as demonstrated in the screenshots below using VQ Analyzer software, where the -19, -11 motion vector corresponds to $K/2^N$ unit horizontal and $L/2^N$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

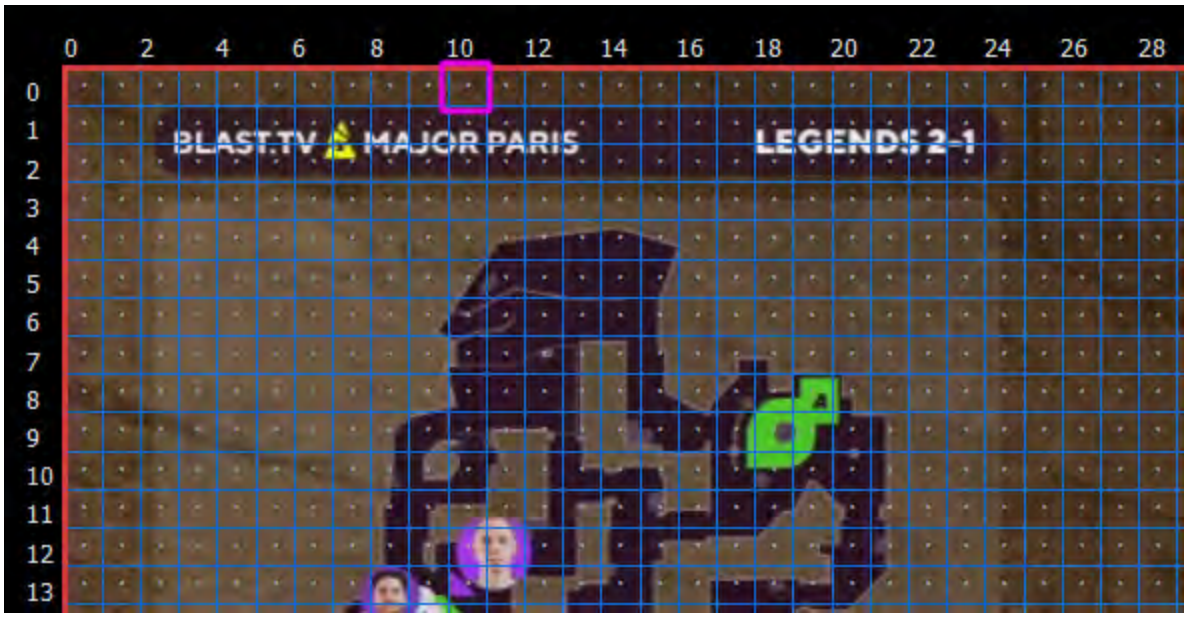


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

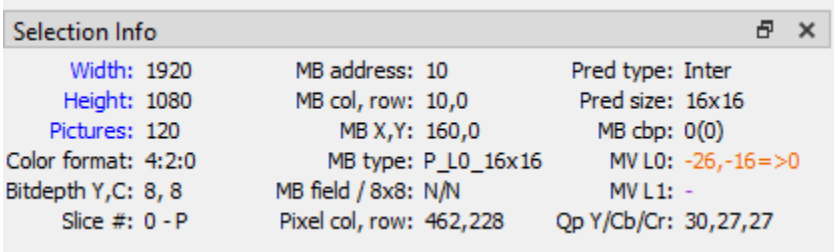


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

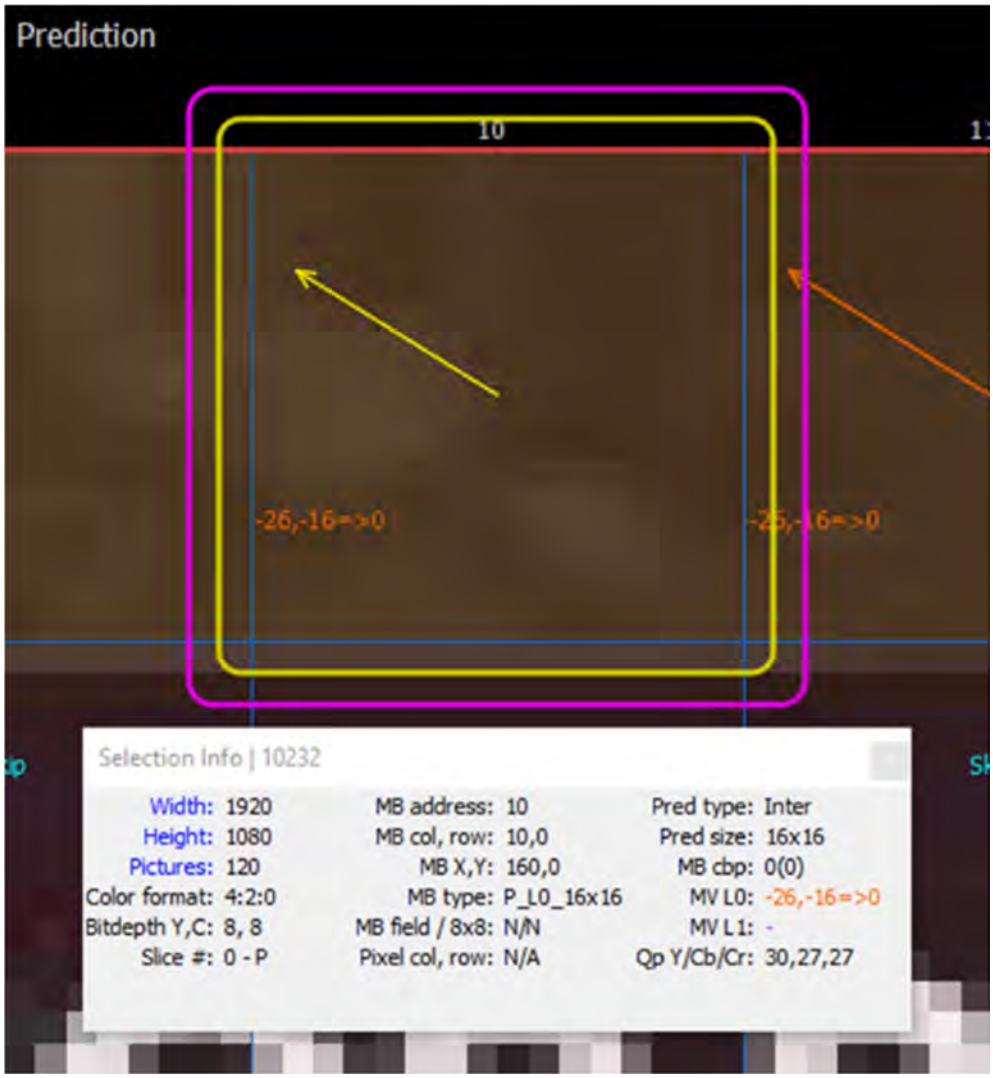
317. When encoding Twitch.tv video, on information and belief, Amazon performs a method to interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal to zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of K and L, using weighted sums of the values of pixels located in rows and columns, as demonstrated in the screenshots below using VQ Analyzer software, where the -26, -16 motion vector corresponds to $K/2^N$ unit horizontal and unit vertical locations.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

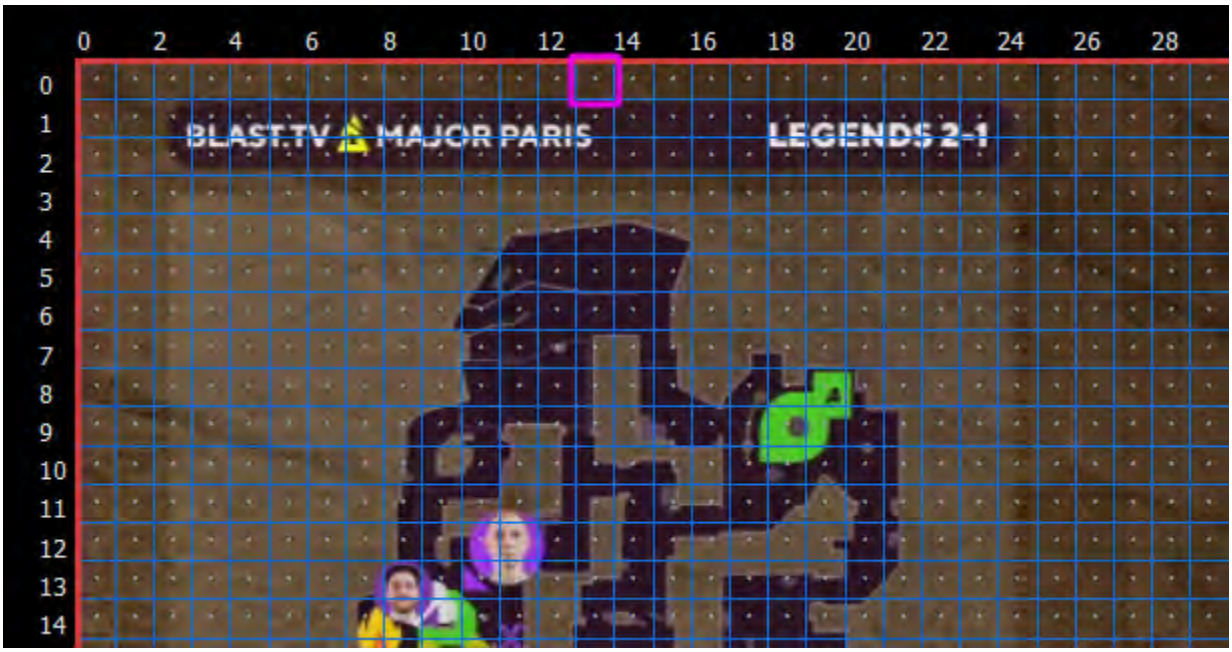


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

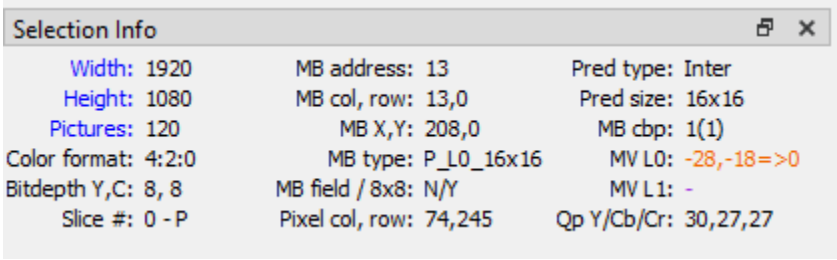


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

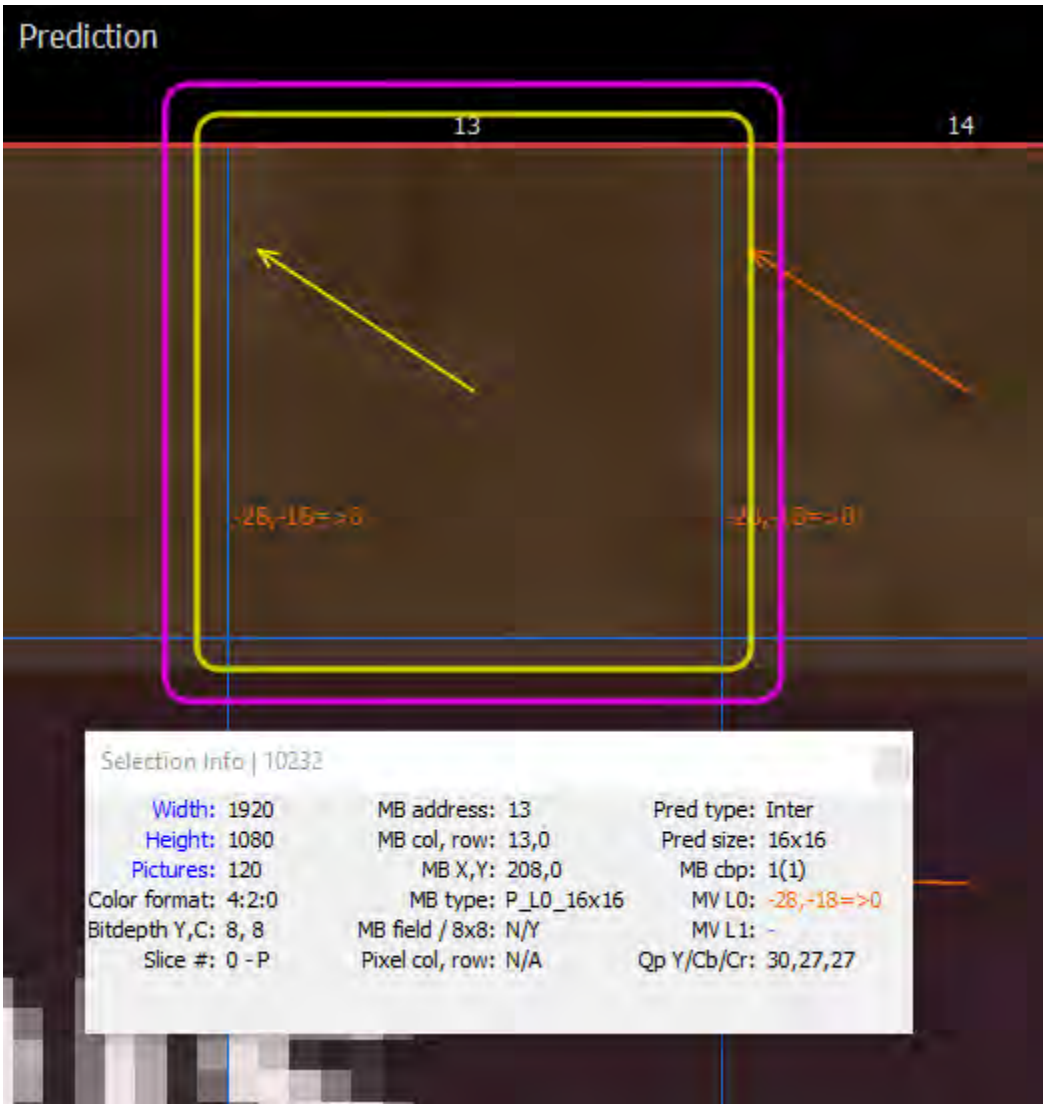
318. When encoding Twitch.tv video, on information and belief, Amazon performs a method to interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal to zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of K and L, using weighted sums of the values of pixels located in rows and columns, as demonstrated in the screenshots below using VQ Analyzer software, where the -28, -18 motion vector corresponds to unit horizontal and $L/2^N$ unit vertical locations.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

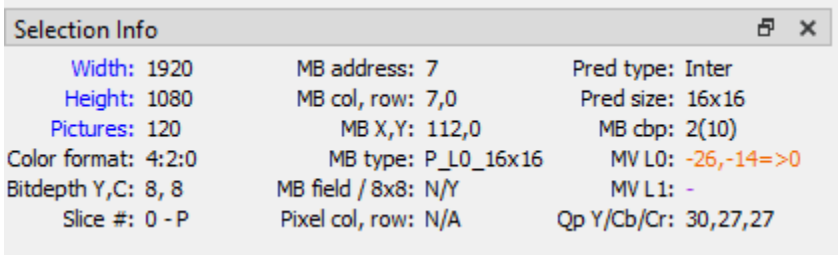


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

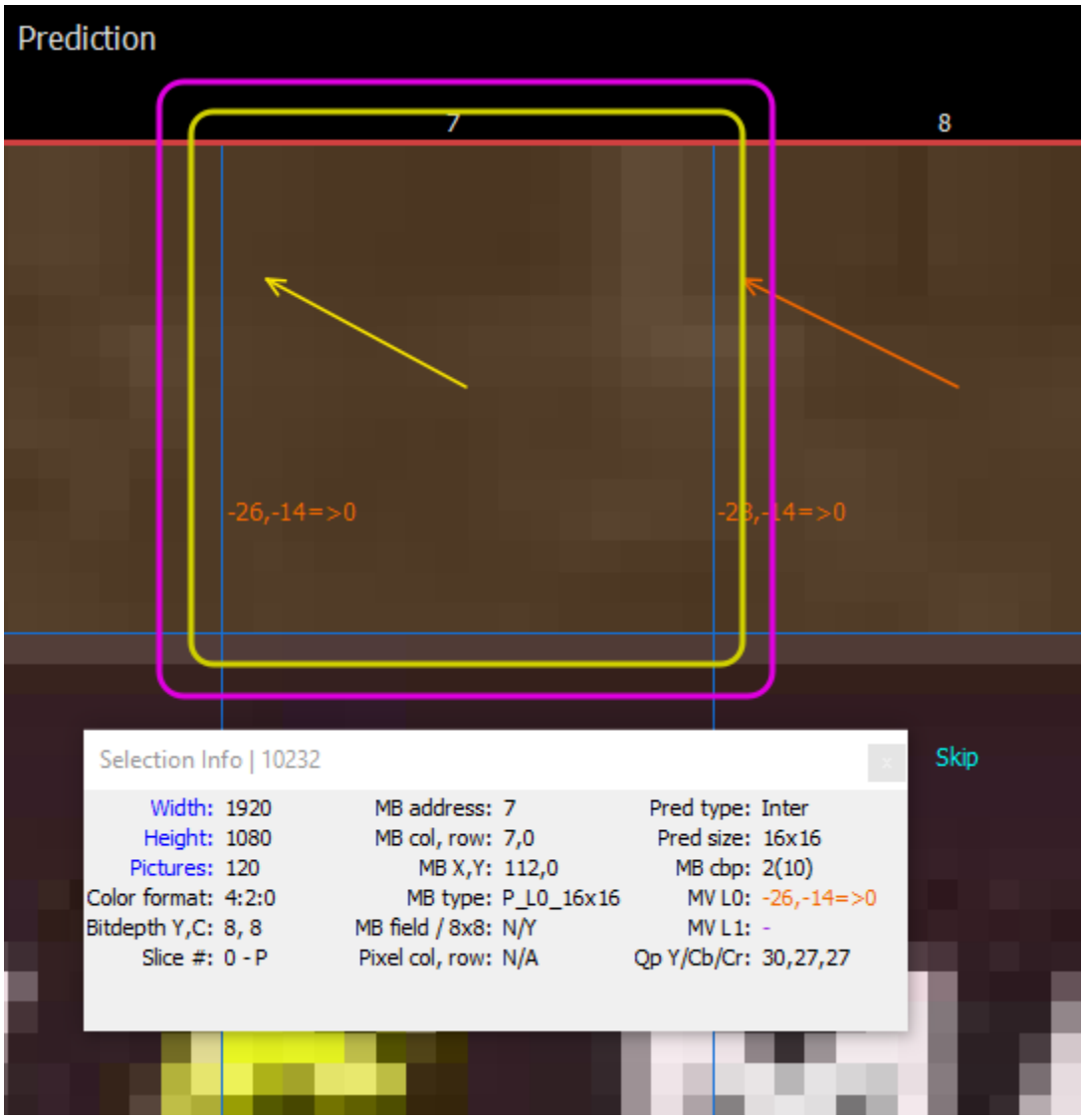
319. When encoding Twitch.tv video, on information and belief, Amazon performs a method to interpolate sub-pixel values for sub-pixels having co-ordinates with even values of K and L, used in the interpolation of sub-pixel values for the sub-pixels having co-ordinates with odd values of K and L, using a predetermined choice of either a weighted sum of the values of sub-pixels having co-ordinates with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the values of sub-pixels having co-ordinates with K equal to zero

and L equal to an even value and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, as demonstrated in the screenshots below using VQ Analyzer software, where the -26, -14 motion vector corresponds to $K/2^N$ unit horizontal and $L/2^N$ unit vertical locations.

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

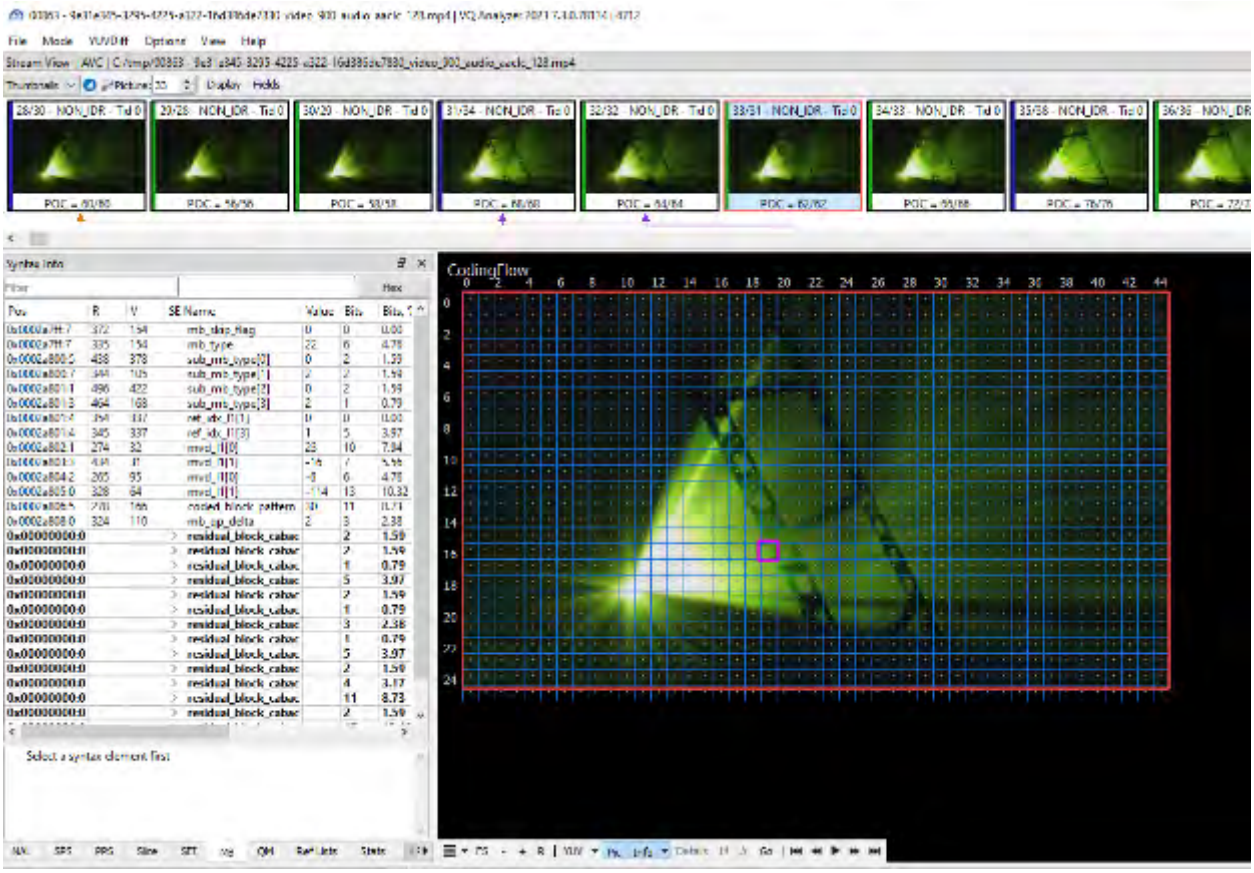
G. Amazon Makes, Imports, Uses, Sells, and/or Offers for Sale Products and Services that Infringe the '701 Patent

320. The Accused Products infringe one or more claims of the '701 Patent, including, for example, claim 1.

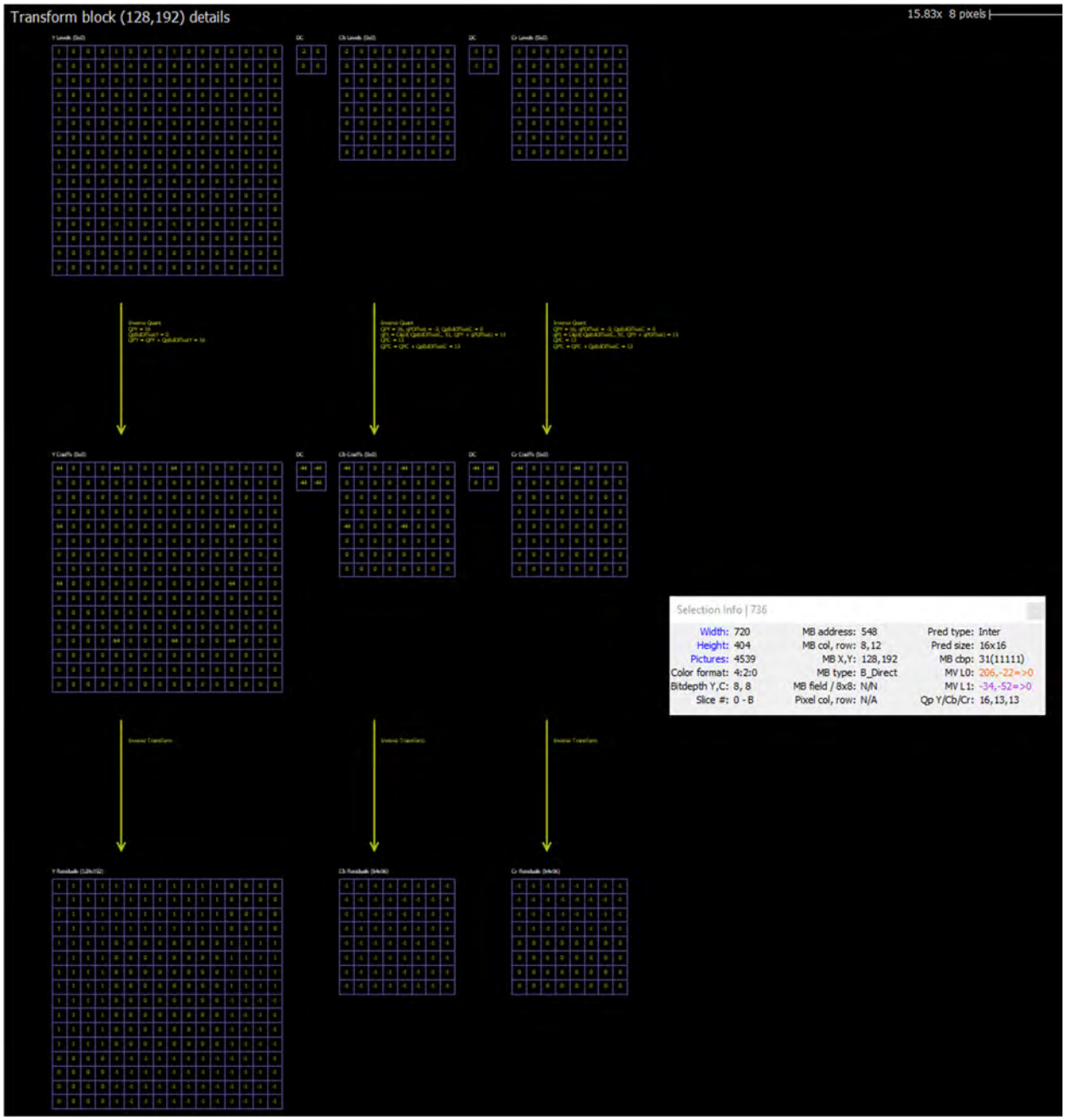
321. As just one example of infringement, on information and belief, Amazon performs a method of image coding in a manner that is covered by claim 1 of the '701 Patent for Amazon Prime Video content (such as trailers), as demonstrated in the screenshots using VQ Analyzer

software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 212.

322. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs image coding, wherein an image is divided in an encoder into a plurality of blocks having a plurality of pixels, each pixel having a pixel value, and a transform coding operation is performed on a block of pixels to produce a corresponding block of transform coefficient values, the block of transform coefficient values being scanned in a given scanning order to produce a scanned array of coefficient values arranged according to the scanning order, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

323. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs representing the coefficient values in the scanned array by a plurality of number pairs, each of said number pairs having a first number and a second number, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info

Filter | Hex

SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	0
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	3
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	-3
deblocking_filter_control_present_flag	1
constrained_intra_pred_flag	0
redundant_pic_cnt_present_flag	0
rbsp_stop_one_bit	1
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info | 4712

Filter | Hex

Pos	R	V	SE Name	Value
0x00000000:0			**residual_block_cabac**	
0x00000000:0			**residual_block_cabac**	
0x0002a80b:3	414	304	coded_block_flag	1
0x0002a80b:3	371	304	significant_coeff_flag[0]	1
0x0002a80b:3	364	304	last_significant_coeff_flag[0]	1
0x0002a80b:3	357	304	coeff_abs_level_minus1[0]	1
0x0002a80b:6	354	155	coeff_sign_flag[0]	0
0x00000000:0			**residual_block_cabac**	
0x0002a80b:7	354	310	coded_block_flag	1
0x0002a80b:7	319	310	significant_coeff_flag[0]	1
0x0002a80b:7	313	310	last_significant_coeff_flag[0]	0
0x0002a80c:5	384	253	significant_coeff_flag[1]	0
0x0002a80c:6	356	94	significant_coeff_flag[2]	1
0x0002a80c:6	287	94	last_significant_coeff_flag[2]	1
0x0002a80c:7	394	189	coeff_abs_level_minus1[2]	0
0x0002a80c:7	289	189	coeff_sign_flag[2]	1
0x0002a80d:0	289	90	coeff_abs_level_minus1[0]	0
0x0002a80d:1	388	180	coeff_sign_flag[0]	0
0x00000000:0			**residual_block_cabac**	
0x00000000:0			**residual_block_cabac**	
0x0002a80d:4	380	271	coded_block_flag	1
0x0002a80d:4	286	271	significant_coeff_flag[0]	0
0x0002a80e:0	320	80	significant_coeff_flag[1]	1
0x0002a80e:1	306	161	last_significant_coeff_flag[1]	0
0x0002a80e:2	450	323	significant_coeff_flag[2]	1
0x0002a80e:2	346	323	last_significant_coeff_flag[2]	0
0x0002a80e:3	256	211	coeff_abs_level_minus1[2]	0
0x0002a80e:4	430	423	coeff_sign_flag[2]	1
0x0002a80e:5	430	416	coeff_abs_level_minus1[1]	1
0x0002a80f:1	283	128	coeff_sign_flag[1]	0
0x0002a80f:2	283	257	coeff_abs_level_minus1[0]	0
0x0002a80f:4	308	205	coeff_sign_flag[0]	1
0x0002a80f:5	308	102	end_of_slice_flag	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Sta

Selection Info | 4712

Width:	720	MB address:	739	Pred type:	Inter
Height:	404	MB col, row:	19,16	Pred size:	N/A
Pictures:	4539	MB X,Y:	304,256	MB cbp:	30(11110)
Color format:	4:2:0	MB type:	B8x8	MV L0:	N/A
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/N	MV L1:	N/A
Slice #:	0 - B	Pixel col, row:	N/A	Qp Y/Cb/Cr:	20,17,17

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

324. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs assigning the first numbers to one of a plurality of contexts representative of the first numbers such that the first number of a first number pair is assigned to a context at least partly in dependence on a first number of a second number pair, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info

Filter | Hex

SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	0
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	3
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	-3
deblocking_filter_control_present_flag	1
constrained_intra_pred_flag	0
redundant_pic_cnt_present_flag	0
rbsp_stop_one_bit	1
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info | 4712

Filter | Hex

Pos	R	V	SE Name	Value
0x00000000:0			**residual_block_cabac**	
0x00000000:0			**residual_block_cabac**	
0x0002a80b:3	414	304	coded_block_flag	1
0x0002a80b:3	371	304	significant_coeff_flag[0]	1
0x0002a80b:3	364	304	last_significant_coeff_flag[0]	1
0x0002a80b:3	357	304	coeff_abs_level_minus1[0]	1
0x0002a80b:6	354	155	coeff_sign_flag[0]	0
0x00000000:0			**residual_block_cabac**	
0x0002a80b:7	354	310	coded_block_flag	1
0x0002a80b:7	319	310	significant_coeff_flag[0]	1
0x0002a80b:7	313	310	last_significant_coeff_flag[0]	0
0x0002a80c:5	384	253	significant_coeff_flag[1]	0
0x0002a80c:6	356	94	significant_coeff_flag[2]	1
0x0002a80c:6	287	94	last_significant_coeff_flag[2]	1
0x0002a80c:7	394	189	coeff_abs_level_minus1[2]	0
0x0002a80c:7	289	189	coeff_sign_flag[2]	1
0x0002a80d:0	289	90	coeff_abs_level_minus1[0]	0
0x0002a80d:1	388	180	coeff_sign_flag[0]	0
0x00000000:0			**residual_block_cabac**	
0x00000000:0			**residual_block_cabac**	
0x0002a80d:4	380	271	coded_block_flag	1
0x0002a80d:4	286	271	significant_coeff_flag[0]	0
0x0002a80e:0	320	80	significant_coeff_flag[1]	1
0x0002a80e:1	306	161	last_significant_coeff_flag[1]	0
0x0002a80e:2	450	323	significant_coeff_flag[2]	1
0x0002a80e:2	346	323	last_significant_coeff_flag[2]	0
0x0002a80e:3	256	211	coeff_abs_level_minus1[2]	0
0x0002a80e:4	430	423	coeff_sign_flag[2]	1
0x0002a80e:5	430	416	coeff_abs_level_minus1[1]	1
0x0002a80f:1	283	128	coeff_sign_flag[1]	0
0x0002a80f:2	283	257	coeff_abs_level_minus1[0]	0
0x0002a80f:4	308	205	coeff_sign_flag[0]	1
0x0002a80f:5	308	102	end_of_slice_flag	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Sta

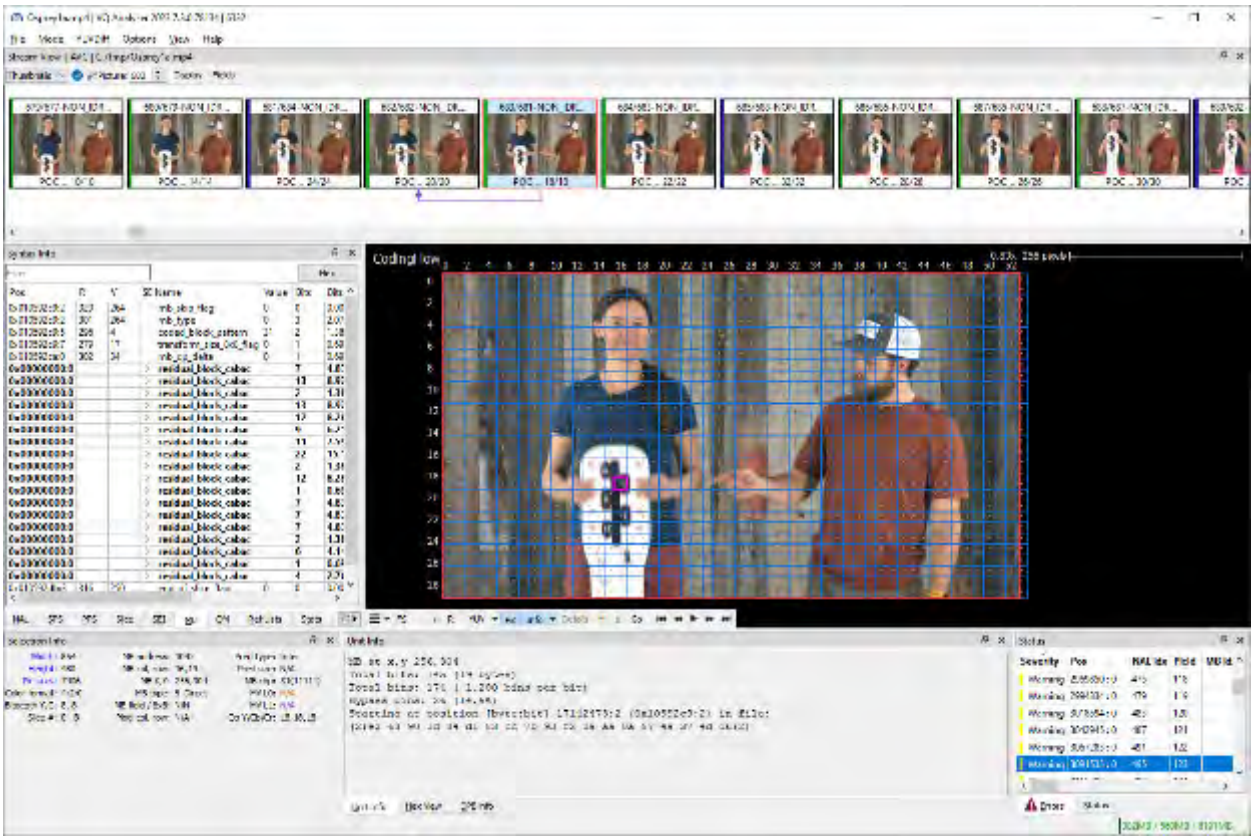
Selection Info | 4712

Width:	720	MB address:	739	Pred type:	Inter
Height:	404	MB col, row:	19,16	Pred size:	N/A
Pictures:	4539	MB X,Y:	304,256	MB cbp:	30(11110)
Color format:	4:2:0	MB type:	B8x8	MV L0:	N/A
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/N	MV L1:	N/A
Slice #:	0 - B	Pixel col, row:	N/A	Qp Y/Cb/Cr:	20,17,17

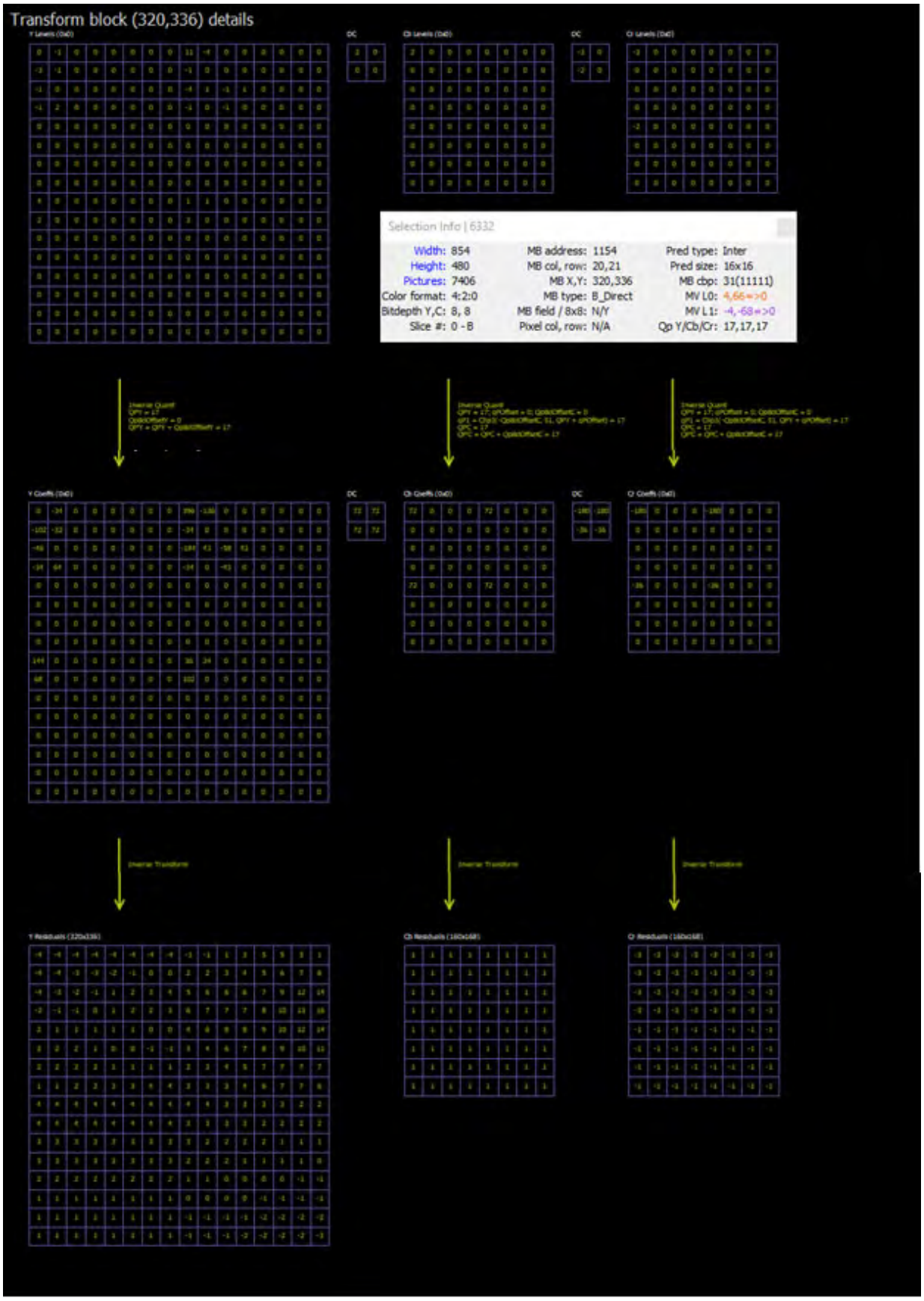
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

325. For another example, on information and belief, Amazon performs a method of image coding in a manner that is covered by claim 1 of the ’701 Patent for Amazon.com advertisements, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 217.

326. When encoding Amazon.com advertisements, on information and belief, Amazon performs image coding, wherein an image is divided in an encoder into a plurality of blocks having a plurality of pixels, each pixel having a pixel value, and a transform coding operation is performed on a block of pixels to produce a corresponding block of transform coefficient values, the block of transform coefficient values being scanned in a given scanning order to produce a scanned array of coefficient values arranged according to the scanning order, as demonstrated in the screenshots below using VQ Analyzer software.

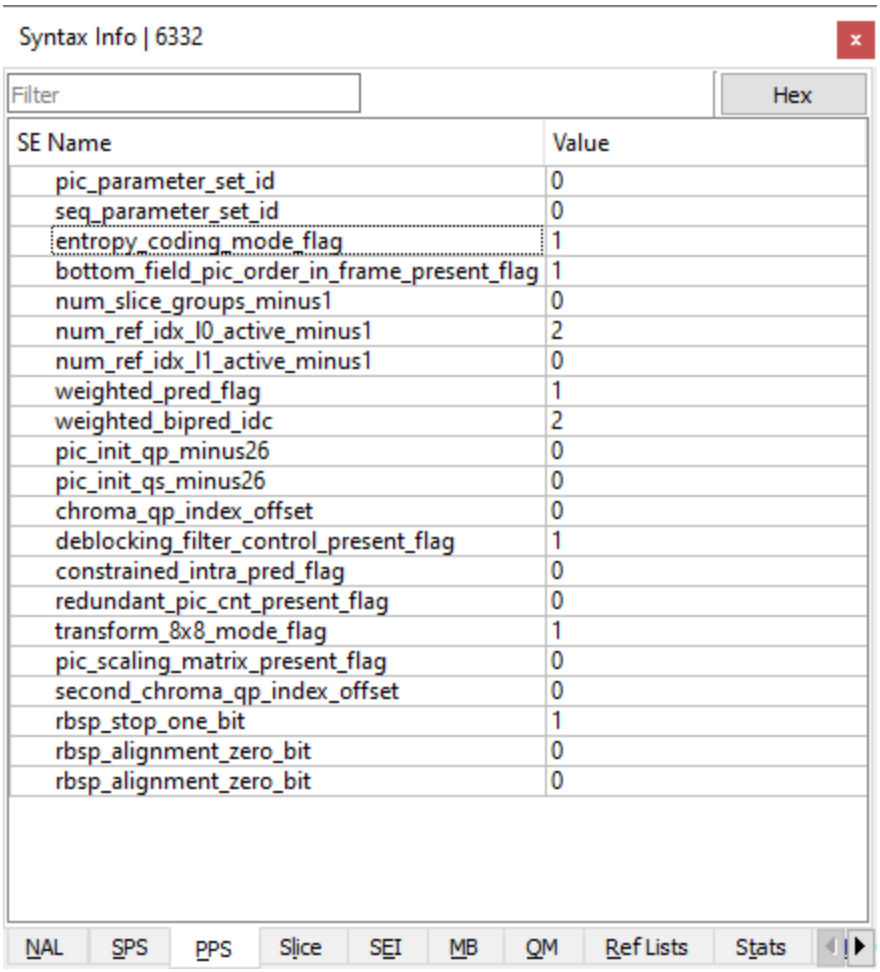


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.


Transform block (320,336) details
Selection Info | 6332
Width: 854
Height: 480
Pictures: 7406
Color format: 4:2:0
Bitdepth Y,C: 8, 8
Slice #: 0 - B
MB address: 1154
MB col, row: 20,21
MB X,Y: 320,336
MB type: B_Direct
MB field / 8x8: N/Y
Pixel col, row: N/A
Pred type: Inter
Pred size: 16x16
MB cbp: 31(11111)
MV L0: 4,66=>0
MV L1: -4,-68=>0
Qp Y/Cb/Cr: 17,17,17

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

327. When encoding Amazon.com advertisements, on information and belief, Amazon performs representing the coefficient values in the scanned array by a plurality of number pairs, each of said number pairs having a first number and a second number, as demonstrated in the screenshots below using VQ Analyzer software.



SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	1
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	2
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	0
deblocking_filter_control_present_flag	1
constrained_intra_pred_flag	0
redundant_pic_cnt_present_flag	0
transform_8x8_mode_flag	1
pic_scaling_matrix_present_flag	0
second_chroma_qp_index_offset	0
rbsp_stop_one_bit	1
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info | 6332

Filter | Hex

Pos	R	V	SE Name	Value	Bits
0x00000000:0			**residual_block_cabac**		**9**
0x00000000:0			**residual_block_cabac**		**11**
0x010592d1:1	427	96	coded_block_flag	1	0
0x010592d1:1	316	96	significant_coeff_flag[0]	1	1
0x010592d1:2	452	193	last_significant_coeff_flag[0]	0	0
0x010592d1:2	257	193	significant_coeff_flag[1]	1	2
0x010592d1:4	492	239	last_significant_coeff_flag[1]	0	0
0x010592d1:4	416	239	significant_coeff_flag[2]	0	1
0x010592d1:5	416	63	significant_coeff_flag[3]	0	0
0x010592d1:5	343	63	significant_coeff_flag[4]	0	1
0x010592d1:6	402	126	significant_coeff_flag[5]	1	1
0x010592d1:7	410	253	last_significant_coeff_flag[5]	1	1
0x010592d2:0	394	81	coeff_abs_level_minus1[5]	0	0
0x010592d2:0	361	81	coeff_sign_flag[5]	0	1
0x010592d2:1	361	163	coeff_abs_level_minus1[1]	0	0
0x010592d2:1	299	163	coeff_sign_flag[1]	1	1
0x010592d2:2	299	27	coeff_abs_level_minus1[0]	0	1
0x010592d2:3	486	54	coeff_sign_flag[0]	0	1
0x00000000:0			**residual_block_cabac**		**22**
0x010592d2:4	486	109	coded_block_flag	1	0
0x010592d2:4	364	109	significant_coeff_flag[0]	1	0
0x010592d2:4	260	109	last_significant_coeff_flag[0]	0	1
0x010592d2:5	298	218	significant_coeff_flag[1]	1	1
0x010592d2:6	256	96	last_significant_coeff_flag[1]	0	1
0x010592d2:7	426	193	significant_coeff_flag[2]	0	1
0x010592d3:0	436	387	significant_coeff_flag[3]	1	2
0x010592d3:2	276	83	last_significant_coeff_flag[3]	0	1
0x010592d3:3	406	167	significant_coeff_flag[4]	0	1
0x010592d3:4	492	335	significant_coeff_flag[5]	0	1
0x010592d3:5	432	118	significant_coeff_flag[6]	0	0
0x010592d3:5	280	118	significant_coeff_flag[7]	0	1
0x010592d3:6	304	236	significant_coeff_flag[8]	0	1
0x010592d3:7	256	121	significant_coeff_flag[9]	0	1
0x010592d4:0	350	242	significant_coeff_flag[10]	0	0
0x010592d4:0	265	242	significant_coeff_flag[11]	0	1
0x010592d4:1	256	210	significant_coeff_flag[12]	0	3
0x010592d4:4	424	56	significant_coeff_flag[13]	0	1
0x010592d4:5	432	113	significant_coeff_flag[14]	1	0
0x010592d4:5	366	113	last_significant_coeff_flag[14]	1	0
0x010592d4:5	344	113	coeff_abs_level_minus1[14]	0	0
0x010592d4:5	317	113	coeff_sign_flag[14]	0	1
0x010592d4:6	317	226	coeff_abs_level_minus1[3]	0	0
0x010592d4:6	269	226	coeff_sign_flag[3]	1	1
0x010592d4:7	269	184	coeff_abs_level_minus1[1]	0	1
0x010592d5:0	432	368	coeff_sign_flag[1]	1	1
0x010592d5:1	432	305	coeff_abs_level_minus1[0]	0	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats | HRD

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

328. When encoding Amazon.com advertisements, on information and belief, Amazon performs assigning the first numbers to one of a plurality of contexts representative of the first numbers such that the first number of a first number pair is assigned to a context at least partly in dependence on a first number of a second number pair, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info | 6332

Filter | Hex

SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	1
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	2
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	0
deblocking_filter_control_present_flag	1
constrained_intra_pred_flag	0
redundant_pic_cnt_present_flag	0
transform_8x8_mode_flag	1
pic_scaling_matrix_present_flag	0
second_chroma_qp_index_offset	0
rbsp_stop_one_bit	1
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info | 6332

Filter | Hex

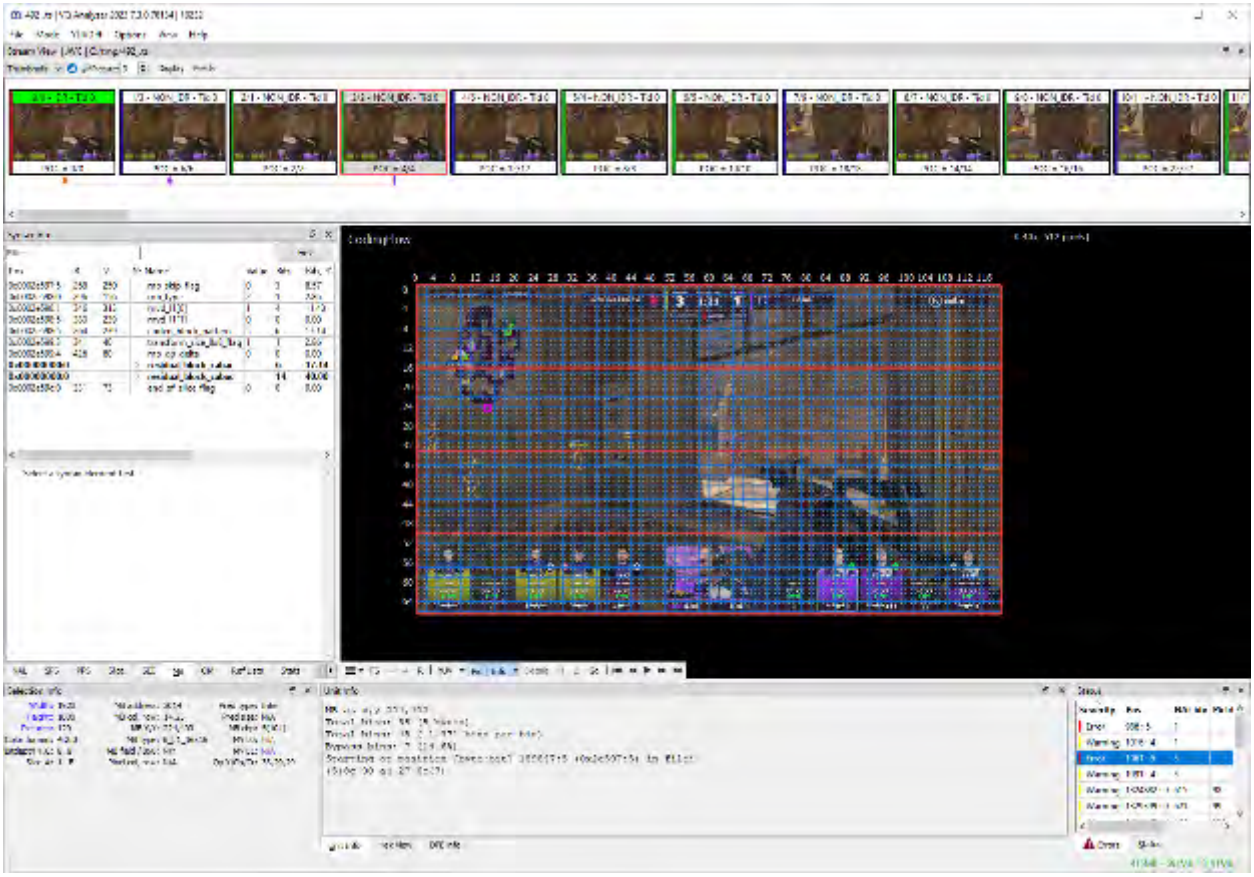
Pos	R	V	SE Name	Value	Bits
0x00000000:0			**residual_block_cabac**		**9**
0x00000000:0			**residual_block_cabac**		**11**
0x010592d1:1	427	96	coded_block_flag	1	0
0x010592d1:1	316	96	significant_coeff_flag[0]	1	1
0x010592d1:2	452	193	last_significant_coeff_flag[0]	0	0
0x010592d1:2	257	193	significant_coeff_flag[1]	1	2
0x010592d1:4	492	239	last_significant_coeff_flag[1]	0	0
0x010592d1:4	416	239	significant_coeff_flag[2]	0	1
0x010592d1:5	416	63	significant_coeff_flag[3]	0	0
0x010592d1:5	343	63	significant_coeff_flag[4]	0	1
0x010592d1:6	402	126	significant_coeff_flag[5]	1	1
0x010592d1:7	410	253	last_significant_coeff_flag[5]	1	1
0x010592d2:0	394	81	coeff_abs_level_minus1[5]	0	0
0x010592d2:0	361	81	coeff_sign_flag[5]	0	1
0x010592d2:1	361	163	coeff_abs_level_minus1[1]	0	0
0x010592d2:1	299	163	coeff_sign_flag[1]	1	1
0x010592d2:2	299	27	coeff_abs_level_minus1[0]	0	1
0x010592d2:3	486	54	coeff_sign_flag[0]	0	1
0x00000000:0			**residual_block_cabac**		**22**
0x010592d2:4	486	109	coded_block_flag	1	0
0x010592d2:4	364	109	significant_coeff_flag[0]	1	0
0x010592d2:4	260	109	last_significant_coeff_flag[0]	0	1
0x010592d2:5	298	218	significant_coeff_flag[1]	1	1
0x010592d2:6	256	96	last_significant_coeff_flag[1]	0	1
0x010592d2:7	426	193	significant_coeff_flag[2]	0	1
0x010592d3:0	436	387	significant_coeff_flag[3]	1	2
0x010592d3:2	276	83	last_significant_coeff_flag[3]	0	1
0x010592d3:3	406	167	significant_coeff_flag[4]	0	1
0x010592d3:4	492	335	significant_coeff_flag[5]	0	1
0x010592d3:5	432	118	significant_coeff_flag[6]	0	0
0x010592d3:5	280	118	significant_coeff_flag[7]	0	1
0x010592d3:6	304	236	significant_coeff_flag[8]	0	1
0x010592d3:7	256	121	significant_coeff_flag[9]	0	1
0x010592d4:0	350	242	significant_coeff_flag[10]	0	0
0x010592d4:0	265	242	significant_coeff_flag[11]	0	1
0x010592d4:1	256	210	significant_coeff_flag[12]	0	3
0x010592d4:4	424	56	significant_coeff_flag[13]	0	1
0x010592d4:5	432	113	significant_coeff_flag[14]	1	0
0x010592d4:5	366	113	last_significant_coeff_flag[14]	1	0
0x010592d4:5	344	113	coeff_abs_level_minus1[14]	0	0
0x010592d4:5	317	113	coeff_sign_flag[14]	0	1
0x010592d4:6	317	226	coeff_abs_level_minus1[3]	0	0
0x010592d4:6	269	226	coeff_sign_flag[3]	1	1
0x010592d4:7	269	184	coeff_abs_level_minus1[1]	0	1
0x010592d5:0	432	368	coeff_sign_flag[1]	1	1
0x010592d5:1	432	305	coeff_abs_level_minus1[0]	0	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats | HRD

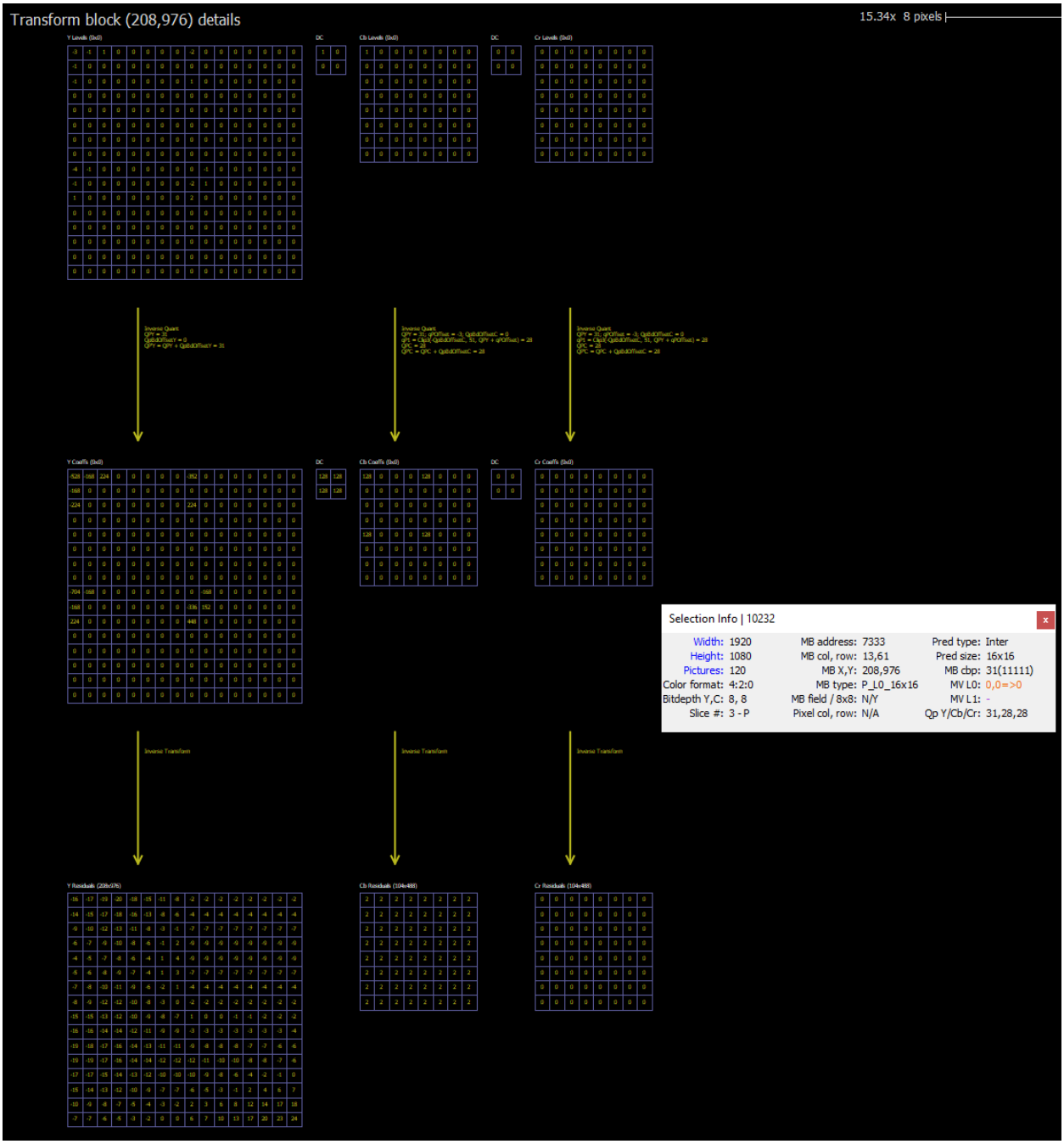
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

329. For another example, on information and belief, Amazon performs a method of image coding in a manner that is covered by claim 1 of the ’701 Patent for Twitch.tv video, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 222.

330. When encoding Twitch.tv video, on information and belief, Amazon performs image coding, wherein an image is divided in an encoder into a plurality of blocks having a plurality of pixels, each pixel having a pixel value, and a transform coding operation is performed on a block of pixels to produce a corresponding block of transform coefficient values, the block of transform coefficient values being scanned in a given scanning order to produce a scanned array of coefficient values arranged according to the scanning order, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

331. When encoding Twitch.tv video, on information and belief, Amazon performs representing the coefficient values in the scanned array by a plurality of number pairs, each of

said number pairs having a first number and a second number, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info | 10232

Filter | Hex

SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	0
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	4
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	-3
deblocking_filter_control_present_flag	1
constrained_intra_pred_flag	0
redundant_pic_cnt_present_flag	0
transform_8x8_mode_flag	1
pic_scaling_matrix_present_flag	0
second_chroma_qp_index_offset	-3
rbsp_stop_one_bit	1

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info | 10232

Filter | Hex

Pos	R	V	SE Name	Value	Bits
0x0002e597:5	268	250	mb_skip_flag	0	3
0x0002e598:0	296	156	mb_type	2	1
0x0002e598:1	346	313	mvd_l1[0]	1	4
0x0002e598:5	380	239	mvd_l1[1]	0	0
0x0002e598:5	264	239	coded_block_pattern	5	6
0x0002e599:3	341	40	transform_size_8x8_flag	1	1
0x0002e599:4	426	80	mb_qp_delta	0	0
0x00000000:0			**residual_block_cabac**		**6**
0x0002e599:4	266	80	significant_coeff_flag[0]	1	1
0x0002e599:5	276	161	last_significant_coeff_flag[0]	0	0
0x0002e599:5	256	161	significant_coeff_flag[1]	0	1
0x0002e599:6	394	323	significant_coeff_flag[2]	1	1
0x0002e599:7	356	215	last_significant_coeff_flag[2]	1	1
0x0002e59a:0	300	19	coeff_abs_level_minus1[1]	0	0
0x0002e59a:0	286	19	coeff_sign_flag[1]	0	1
0x0002e59a:1	286	38	coeff_abs_level_minus1[0]	0	0
0x0002e59a:1	273	38	coeff_sign_flag[0]	0	1
0x00000000:0			**residual_block_cabac**		**14**
0x0002e59a:2	273	77	significant_coeff_flag[0]	1	1
0x0002e59a:3	290	155	last_significant_coeff_flag[0]	0	0
0x0002e59a:3	271	155	significant_coeff_flag[1]	0	1
0x0002e59a:4	430	311	significant_coeff_flag[2]	1	1
0x0002e59a:5	374	137	last_significant_coeff_flag[2]	0	1
0x0002e59a:6	432	275	significant_coeff_flag[3]	0	1
0x0002e59a:7	338	25	significant_coeff_flag[4]	0	0
0x0002e59a:7	316	25	significant_coeff_flag[5]	0	0
0x0002e59a:7	268	25	significant_coeff_flag[6]	0	1
0x0002e59b:0	444	51	significant_coeff_flag[7]	0	0
0x0002e59b:0	419	51	significant_coeff_flag[8]	0	0
0x0002e59b:0	396	51	significant_coeff_flag[9]	1	1
0x0002e59b:1	418	102	last_significant_coeff_flag[9]	0	1
0x0002e59b:2	480	204	significant_coeff_flag[10]	1	0
0x0002e59b:2	275	204	last_significant_coeff_flag[10]	1	2
0x0002e59b:4	464	181	coeff_abs_level_minus1[3]	0	0
0x0002e59b:4	441	181	coeff_sign_flag[3]	0	1
0x0002e59b:5	441	362	coeff_abs_level_minus1[2]	0	0
0x0002e59b:5	423	362	coeff_sign_flag[2]	1	1
0x0002e59b:6	423	301	coeff_abs_level_minus1[1]	0	0
0x0002e59b:6	400	301	coeff_sign_flag[1]	1	1
0x0002e59b:7	400	202	coeff_abs_level_minus1[0]	0	0
0x0002e59b:7	331	202	coeff_sign_flag[0]	1	1
0x0002e59c:0	331	73	end_of_slice_flag	0	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

332. When encoding Twitch.tv video, on information and belief, Amazon performs assigning the first numbers to one of a plurality of contexts representative of the first numbers such that the first number of a first number pair is assigned to a context at least partly in dependence on a first number of a second number pair, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info | 10232

Filter | Hex

SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	0
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	4
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	-3
deblocking_filter_control_present_flag	1
constrained_intra_pred_flag	0
redundant_pic_cnt_present_flag	0
transform_8x8_mode_flag	1
pic_scaling_matrix_present_flag	0
second_chroma_qp_index_offset	-3
rbsp_stop_one_bit	1

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info | 10232

Filter | Hex

Pos	R	V	SE Name	Value	Bits
0x0002e597:5	268	250	mb_skip_flag	0	3
0x0002e598:0	296	156	mb_type	2	1
0x0002e598:1	346	313	mvd_l1[0]	1	4
0x0002e598:5	380	239	mvd_l1[1]	0	0
0x0002e598:5	264	239	coded_block_pattern	5	6
0x0002e599:3	341	40	transform_size_8x8_flag	1	1
0x0002e599:4	426	80	mb_qp_delta	0	0
0x00000000:0			**residual_block_cabac**		**6**
0x0002e599:4	266	80	significant_coeff_flag[0]	1	1
0x0002e599:5	276	161	last_significant_coeff_flag[0]	0	0
0x0002e599:5	256	161	significant_coeff_flag[1]	0	1
0x0002e599:6	394	323	significant_coeff_flag[2]	1	1
0x0002e599:7	356	215	last_significant_coeff_flag[2]	1	1
0x0002e59a:0	300	19	coeff_abs_level_minus1[1]	0	0
0x0002e59a:0	286	19	coeff_sign_flag[1]	0	1
0x0002e59a:1	286	38	coeff_abs_level_minus1[0]	0	0
0x0002e59a:1	273	38	coeff_sign_flag[0]	0	1
0x00000000:0			**residual_block_cabac**		**14**
0x0002e59a:2	273	77	significant_coeff_flag[0]	1	1
0x0002e59a:3	290	155	last_significant_coeff_flag[0]	0	0
0x0002e59a:3	271	155	significant_coeff_flag[1]	0	1
0x0002e59a:4	430	311	significant_coeff_flag[2]	1	1
0x0002e59a:5	374	137	last_significant_coeff_flag[2]	0	1
0x0002e59a:6	432	275	significant_coeff_flag[3]	0	1
0x0002e59a:7	338	25	significant_coeff_flag[4]	0	0
0x0002e59a:7	316	25	significant_coeff_flag[5]	0	0
0x0002e59a:7	268	25	significant_coeff_flag[6]	0	1
0x0002e59b:0	444	51	significant_coeff_flag[7]	0	0
0x0002e59b:0	419	51	significant_coeff_flag[8]	0	0
0x0002e59b:0	396	51	significant_coeff_flag[9]	1	1
0x0002e59b:1	418	102	last_significant_coeff_flag[9]	0	1
0x0002e59b:2	480	204	significant_coeff_flag[10]	1	0
0x0002e59b:2	275	204	last_significant_coeff_flag[10]	1	2
0x0002e59b:4	464	181	coeff_abs_level_minus1[3]	0	0
0x0002e59b:4	441	181	coeff_sign_flag[3]	0	1
0x0002e59b:5	441	362	coeff_abs_level_minus1[2]	0	0
0x0002e59b:5	423	362	coeff_sign_flag[2]	1	1
0x0002e59b:6	423	301	coeff_abs_level_minus1[1]	0	0
0x0002e59b:6	400	301	coeff_sign_flag[1]	1	1
0x0002e59b:7	400	202	coeff_abs_level_minus1[0]	0	0
0x0002e59b:7	331	202	coeff_sign_flag[0]	1	1
0x0002e59c:0	331	73	end_of_slice_flag	0	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

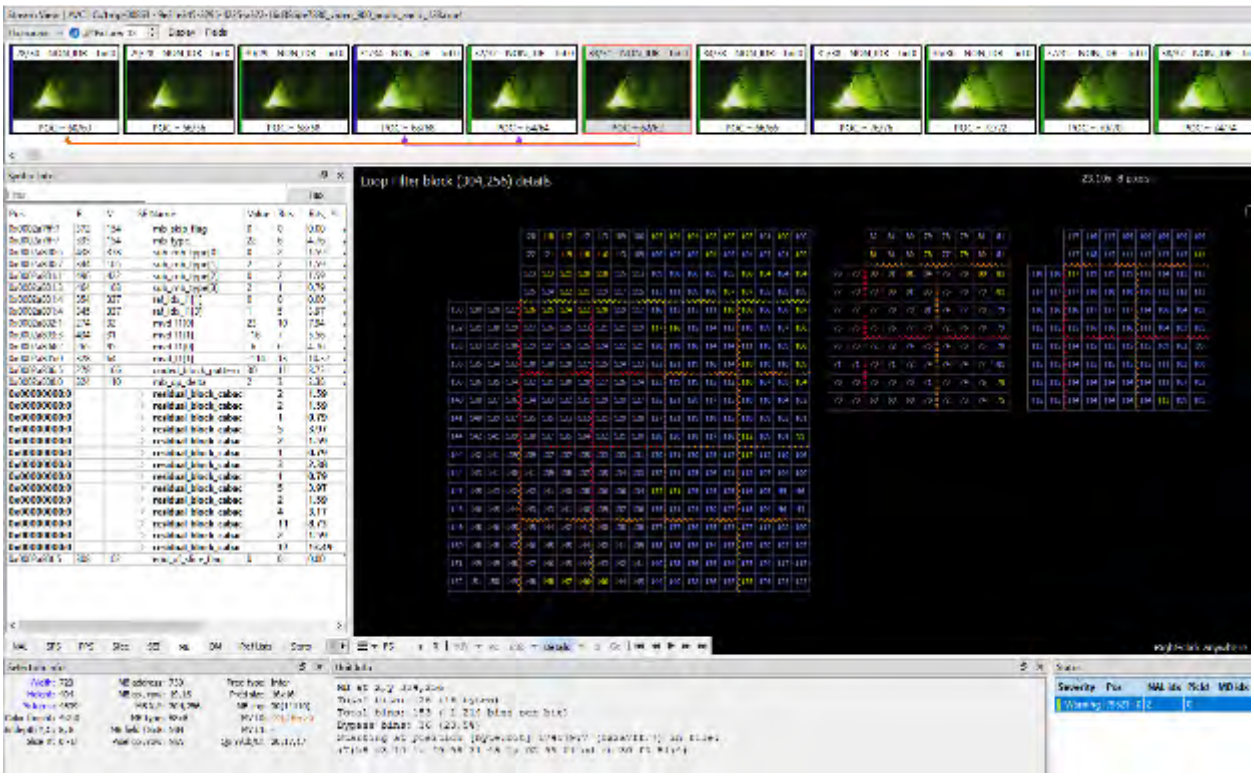
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

H. Amazon Makes, Imports, Uses, Sells, and/or Offers for Sale Products and Services that Infringe the ’891 Patent

333. The Accused Products infringe one or more claims of the ’891 Patent, including, for example, claim 23.

334. As just one example of infringement, on information and belief, Amazon performs a method of video encoding in a manner that is covered by claim 23 of the ’891 Patent for Amazon Prime Video content (such as trailers), as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 212.

335. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs performing an adaptive block boundary filtering operation on a block boundary formed between a first decoded image block on a first side of the block boundary and a second decoded image block on a second side of the block boundary using an adaptive block boundary filter, the first decoded image block having been encoded using a first type of prediction encoding method and the second decoded image block having been encoded using a second type of prediction encoding method, the first type of prediction encoding method and the second type of prediction encoding method being selected from a group of prediction encoding methods comprising at least: intra coding, copy coding, motion-compensated prediction coding, and not-coded coding, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info | 4712

Filter | Hex

SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	0
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	3
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	-3
deblocking_filter_control_present_flag	1
constrained_intra_pred_flag	0
redundant_pic_cnt_present_flag	0
rbsp_stop_one_bit	1
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stat

Selection Info | 4712

Width:	720	MB address:	739	Pred type:	Inter
Height:	404	MB col, row:	19,16	Pred size:	16x16
Pictures:	4539	MB X,Y:	304,256	MB cbp:	30(11110)
Color format:	4:2:0	MB type:	B8x8	MV L0:	-39,16=>0
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/N	MV L1:	-
Slice #:	0 - B	Pixel col, row:	N/A	Qp Y/Cb/Cr:	20,17,17

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info | 4712

Slice #0 size in bits: 44544 SLICE_NONIDR

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	6
pic_parameter_set_id	0
frame_num	15
pic_order_cnt_lsb	62
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	0
num_ref_idx_l1_active_minus1	1
ref_pic_list_reordering()	
cabac_init_idc	0
slice_qp_delta	-10
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	-1
slice_beta_offset_div2	-1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stat

Selection Info | 4712

Width:	720	MB address:	739	Pred type:	Inter
Height:	404	MB col, row:	19,16	Pred size:	16x16
Pictures:	4539	MB X,Y:	304,256	MB cbp:	30(11110)
Color format:	4:2:0	MB type:	B8x8	MV L0:	-39,16=>0
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/N	MV L1:	-
Slice #:	0 - B	Pixel col, row:	N/A	Qp Y/Cb/Cr:	20,17,17

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info | 4712

Filter | Hex

Pos	R	V	SE Name	Value	Bits	E
0x0002a7ff:7	372	154	mb_skip_flag	0	0	0
0x0002a7ff:7	335	154	mb_type	22	6	4
0x0002a800:5	438	378	sub_mb_type[0]	0	2	1
0x0002a800:7	344	105	sub_mb_type[1]	2	2	1
0x0002a801:1	496	422	sub_mb_type[2]	0	2	1
0x0002a801:3	464	168	sub_mb_type[3]	2	1	0
0x0002a801:4	354	337	ref_idx_l1[1]	0	0	0
0x0002a801:4	345	337	ref_idx_l1[3]	1	5	3
0x0002a802:1	274	32	mvd_l1[0]	23	10	7
0x0002a803:3	434	31	mvd_l1[1]	-16	7	5
0x0002a804:2	265	95	mvd_l1[0]	-6	6	4
0x0002a805:0	328	64	mvd_l1[1]	-114	13	1
0x0002a806:5	278	166	coded_block_pattern	30	11	8
0x0002a808:0	324	110	mb_qp_delta	2	3	2
0x00000000:0			**residual_block_cabac**		**2**	**1**
0x00000000:0			**residual_block_cabac**		**2**	**1**
0x00000000:0			**residual_block_cabac**		**1**	**0**
0x00000000:0			**residual_block_cabac**		**5**	**3**
0x00000000:0			**residual_block_cabac**		**2**	**1**
0x00000000:0			**residual_block_cabac**		**1**	**0**
0x00000000:0			**residual_block_cabac**		**3**	**2**
0x00000000:0			**residual_block_cabac**		**1**	**0**
0x00000000:0			**residual_block_cabac**		**5**	**3**
0x00000000:0			**residual_block_cabac**		**2**	**1**
0x00000000:0			**residual_block_cabac**		**4**	**3**
0x00000000:0			**residual_block_cabac**		**11**	**8**
0x00000000:0			**residual_block_cabac**		**2**	**1**
0x00000000:0			**residual_block_cabac**		**17**	**1**
0x0002a80f:5	308	102	end_of_slice_flag	0	0	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stat

Selection Info | 4712

Width:	720	MB address:	739	Pred type:	Inter
Height:	404	MB col, row:	19,16	Pred size:	16x16
Pictures:	4539	MB X,Y:	304,256	MB cbp:	30(11110)
Color format:	4:2:0	MB type:	B8x8	MV L0:	-39,16=>0
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/N	MV L1:	-
Slice #:	0 - B	Pixel col, row:	N/A	Qp Y/Cb/Cr:	20,17,17

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info | 4712

Filter | Hex

Pos	R	V	SE Name	Value	Bits	Bits, %	De
0x00029a74:1	510	334	mb_skip_flag	1	0	0.00	Ari
0x00029a74:1	416	334	end_of_slice_flag	0	0	0.00	Ter

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stat

Selection Info | 4712

Width:	720	MB address:	0	Pred type:	Inter
Height:	404	MB col, row:	0,0	Pred size:	16x16
Pictures:	4539	MB X,Y:	0,0	MB cbp:	0(0)
Color format:	4:2:0	MB type:	Skip	MV L0:	0,0=>0
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/N	MV L1:	0,0=>0
Slice #:	0 - B	Pixel col, row:	N/A	Qp Y/Cb/Cr:	16,13,13

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info | 4712

Filter | Hex

Pos	R	V	SE Name	Value	Bits	Bits, %	De
0x0002a7fb:2	314	309	mb_skip_flag	1	4	80.00	Ari
0x0002a7fb:6	256	187	end_of_slice_flag	0	1	20.00	Ter

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stat

Selection Info | 4712

Width:	720	MB address:	736	Pred type:	Inter
Height:	404	MB col, row:	16,16	Pred size:	16x16
Pictures:	4539	MB X,Y:	256,256	MB cbp:	0(0)
Color format:	4:2:0	MB type:	Skip	MV L0:	-27,15=>0
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/N	MV L1:	17,-16=>0
Slice #:	0 - B	Pixel col, row:	N/A	Qp Y/Cb/Cr:	18,15,15

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info | 4712

Filter | Hex

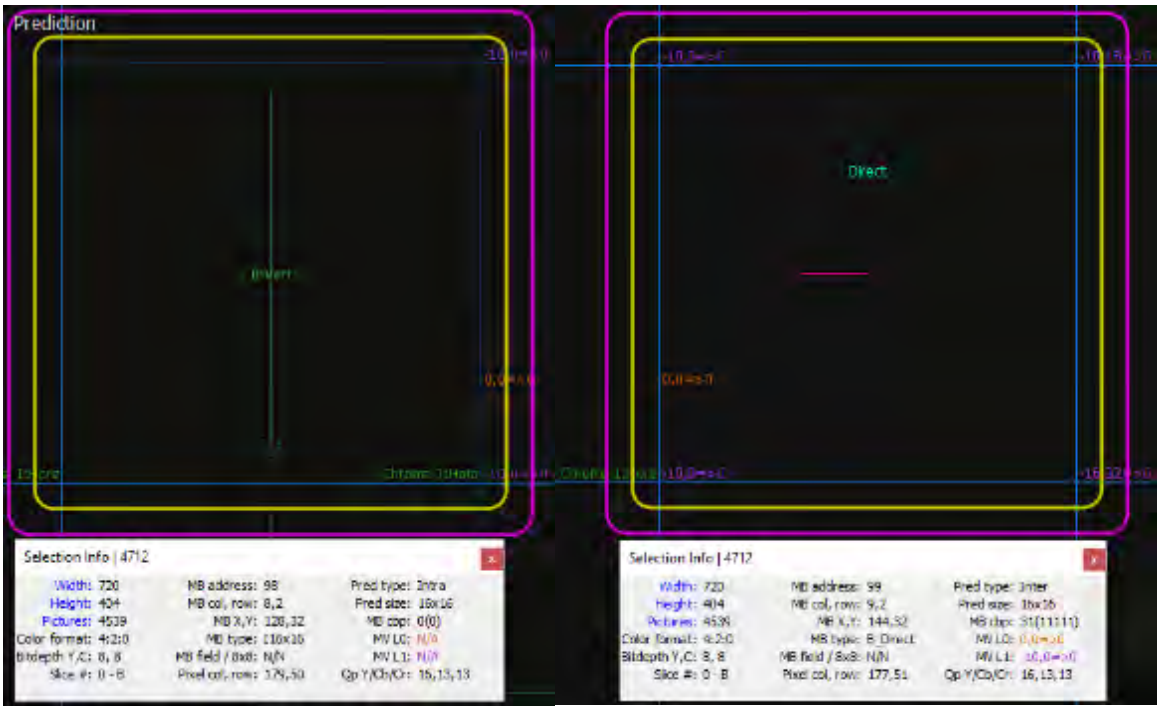
Pos	R	V	SE Name	Value	Bits
0x00029b75:4	470	268	mb_skip_flag	0	0
0x00029b75:4	422	268	mb_type	23	9
0x00029b76:5	360	297	intra_chroma_pred_mode	0	2
0x00029b76:7	464	214	mb_qp_delta	0	1
0x00000000:0			**residual_block_cabac**		**2**
0x00029b77:2	416	239	end_of_slice_flag	0	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stat

Selection Info | 4712

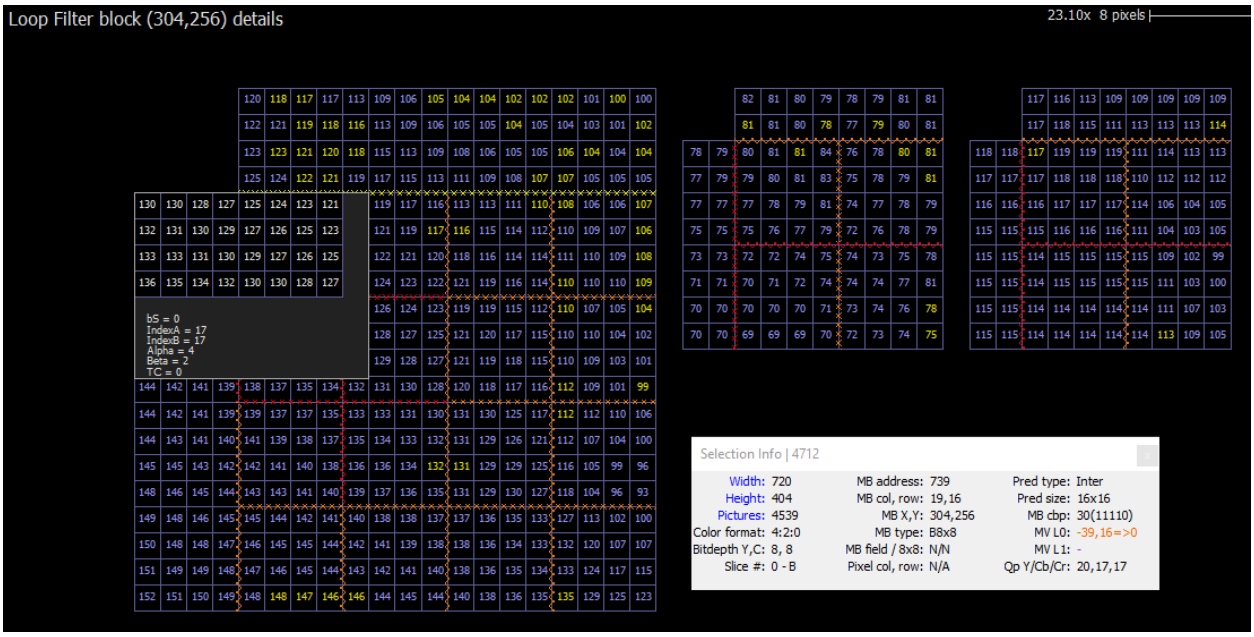
Width: 720	MB address: 93	Pred type: Intra
Height: 404	MB col, row: 3,2	Pred size: 16x16
Pictures: 4539	MB X,Y: 48,32	MB cbp: 0(0)
Color format: 4:2:0	MB type: I16x16	MV L0: N/A
Bitdepth Y,C: 8, 8	MB field / 8x8: N/N	MV L1: N/A
Slice #: 0 - B	Pixel col, row: N/A	Qp Y/Cb/Cr: 16,13,13

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

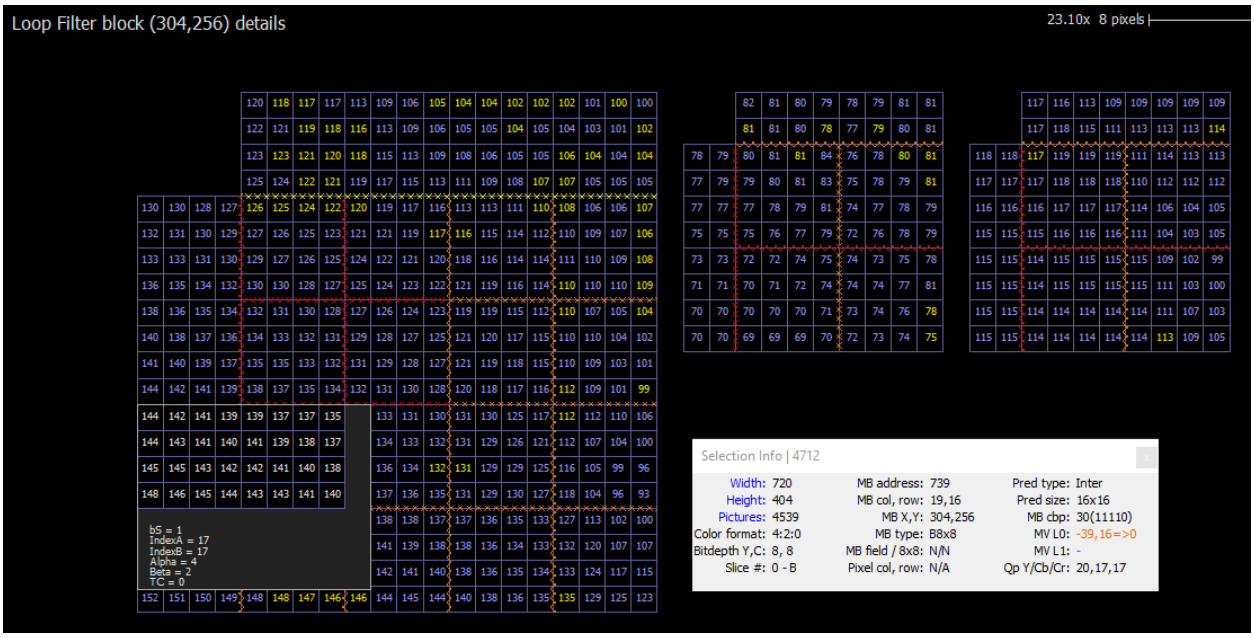


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

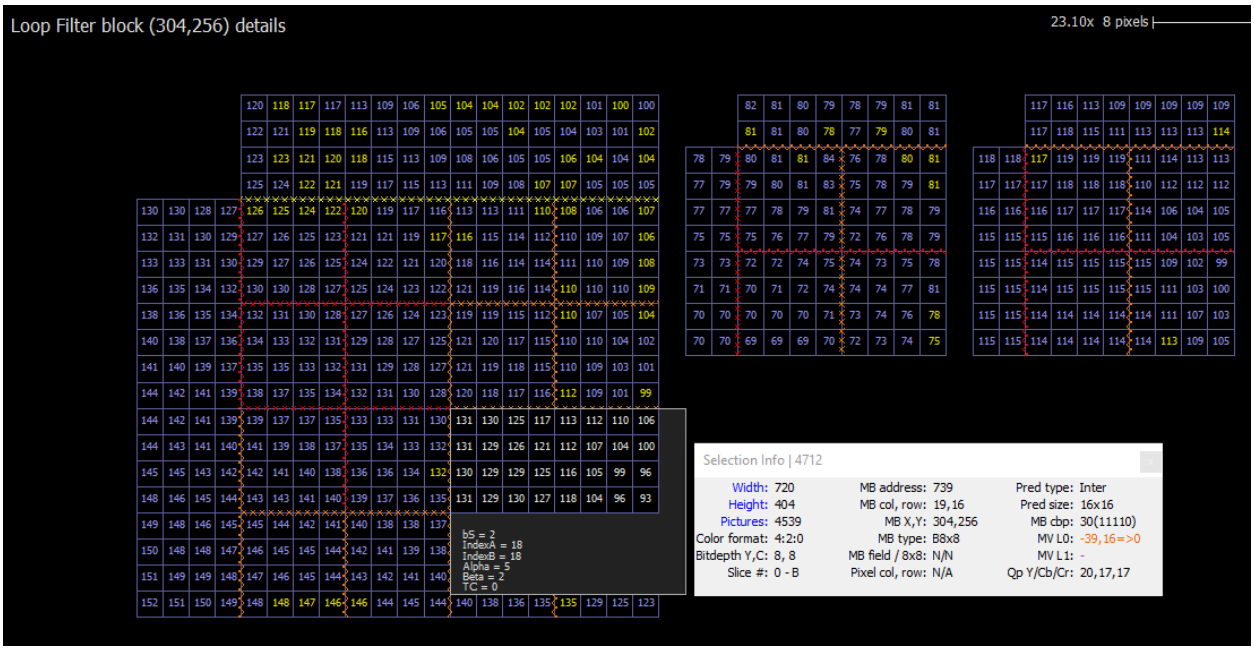
336. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs examining, by the adaptive block boundary filter, the type of the first prediction encoding method and the type of the second prediction encoding method, as demonstrated in the screenshots below using VQ Analyzer software.



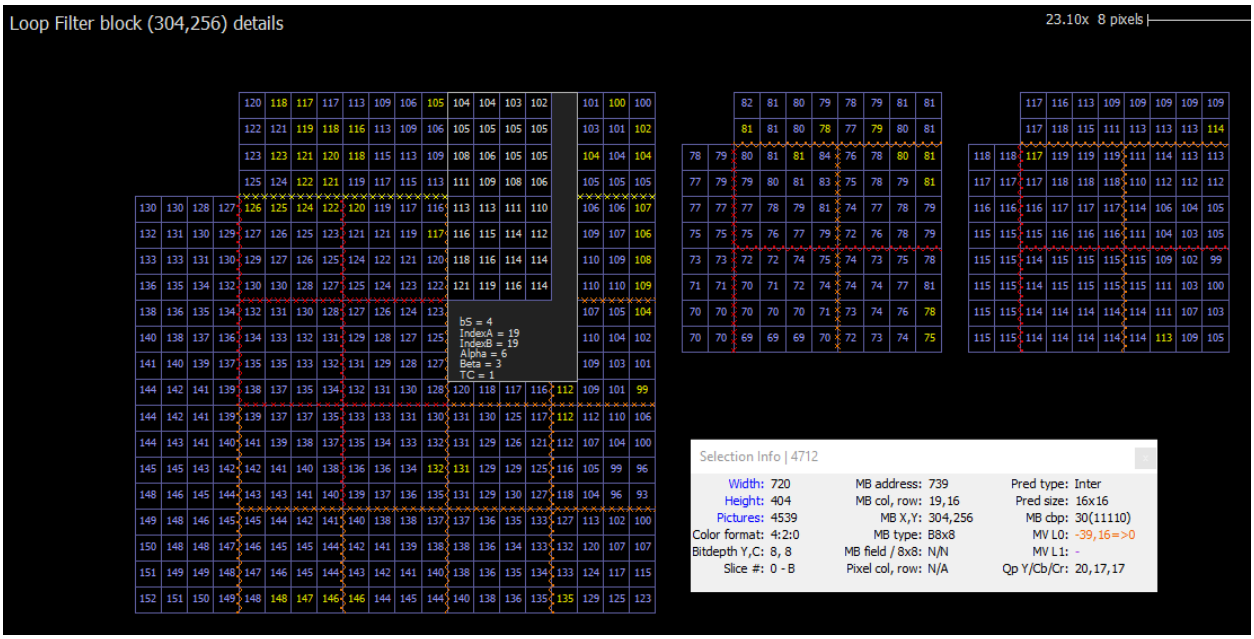
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



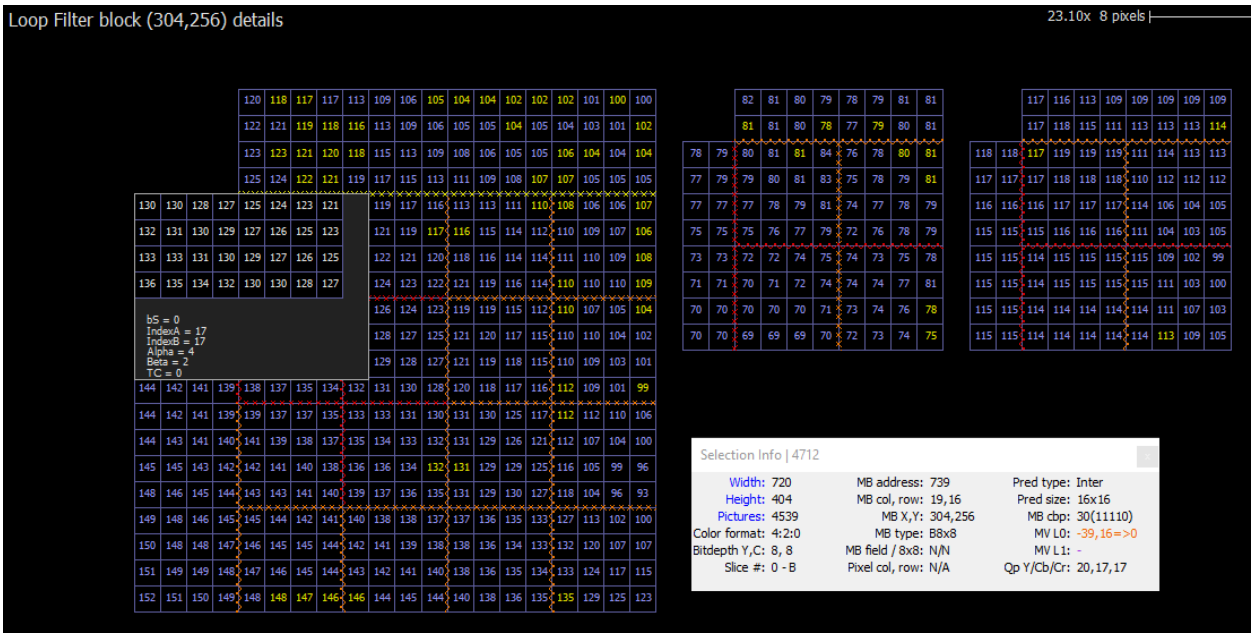
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



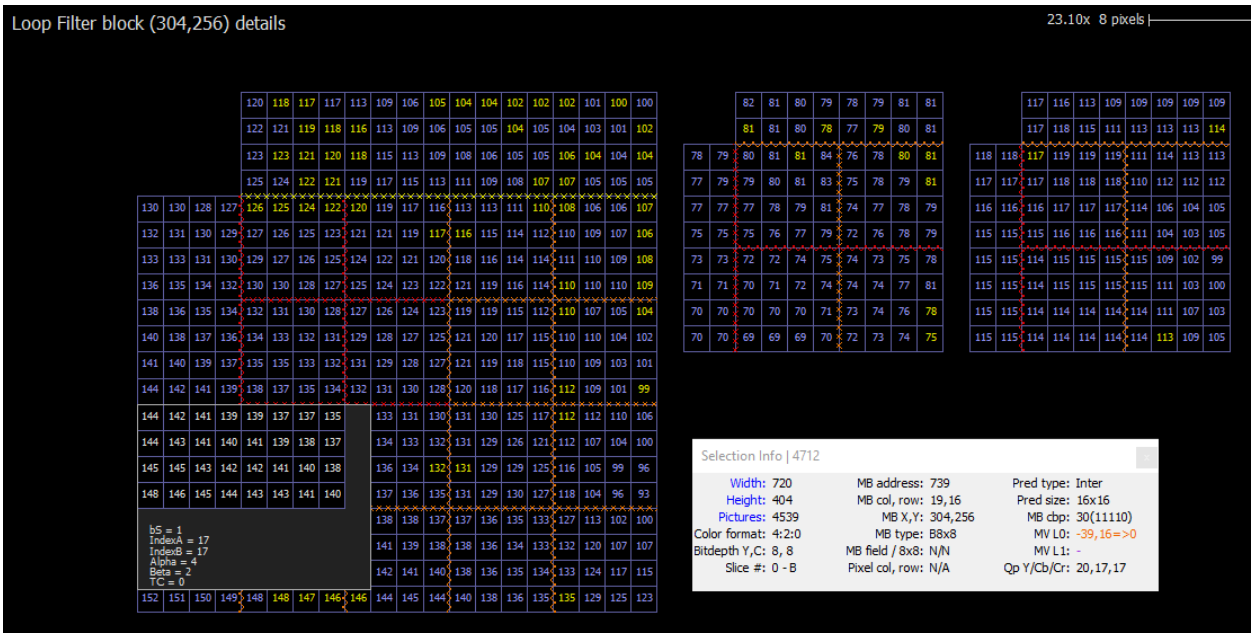
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

337. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs determining, by the adaptive block boundary filter, a first number of pixels to be examined on the first side of the block boundary and a second number of pixels to be examined

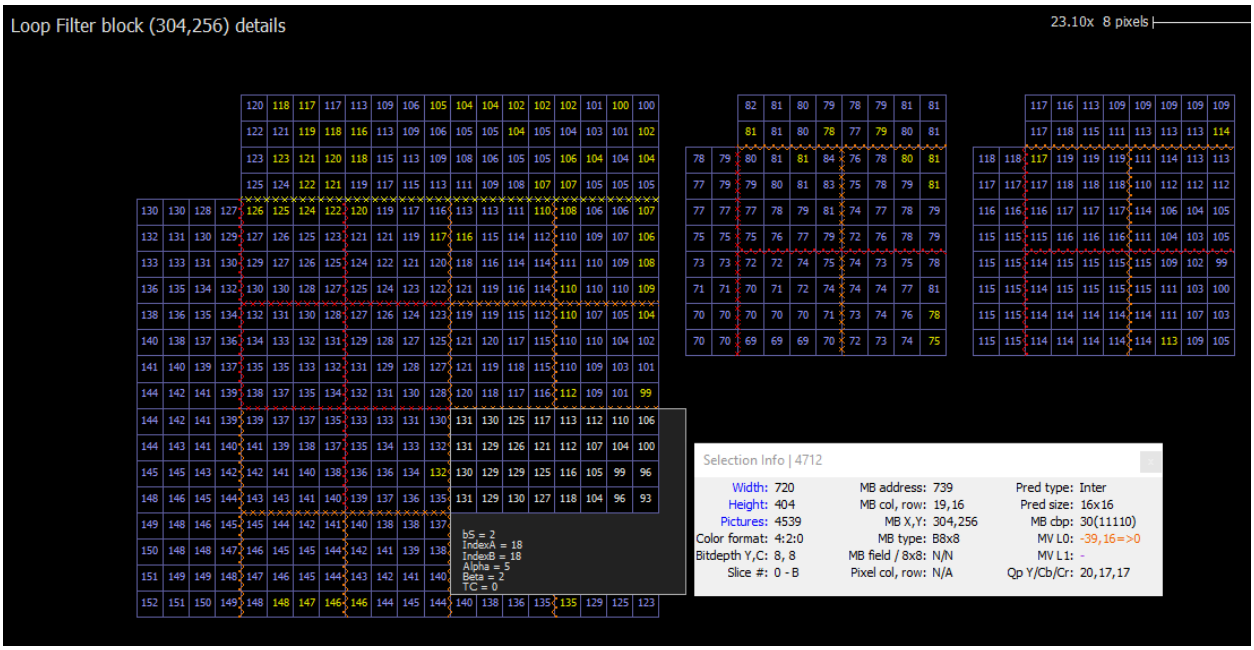
on the second side of the block boundary, as a parameter of the filtering operation, based on the types of the first and second prediction encoding methods, as demonstrated in the screenshots below using VQ Analyzer software.



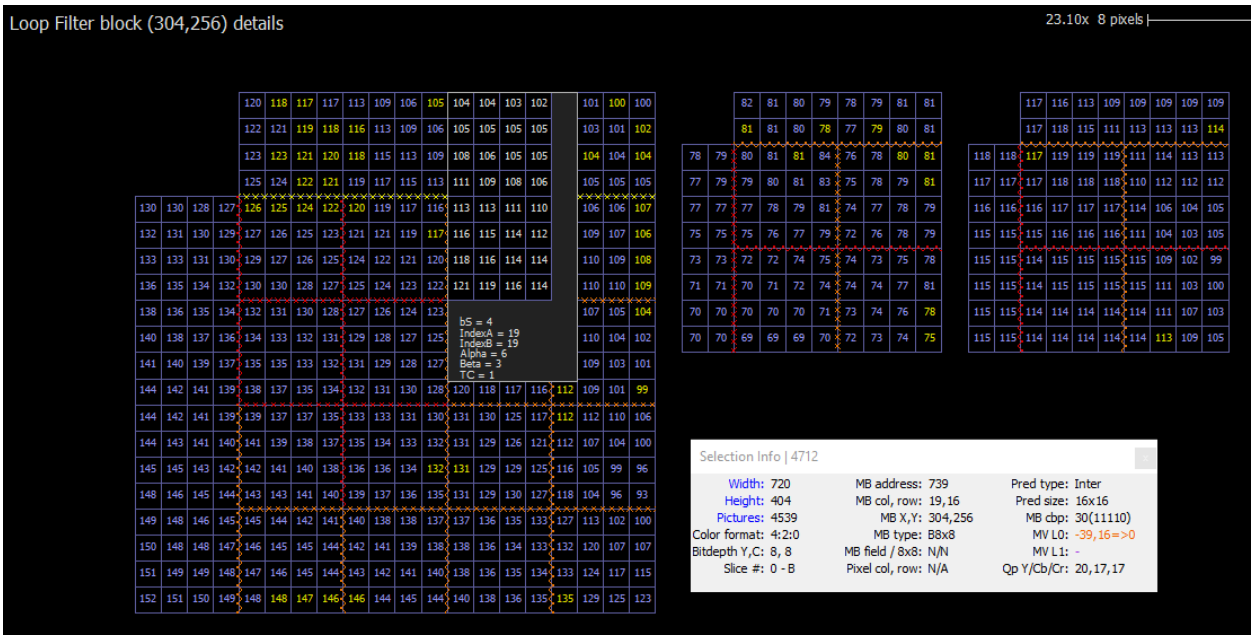
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

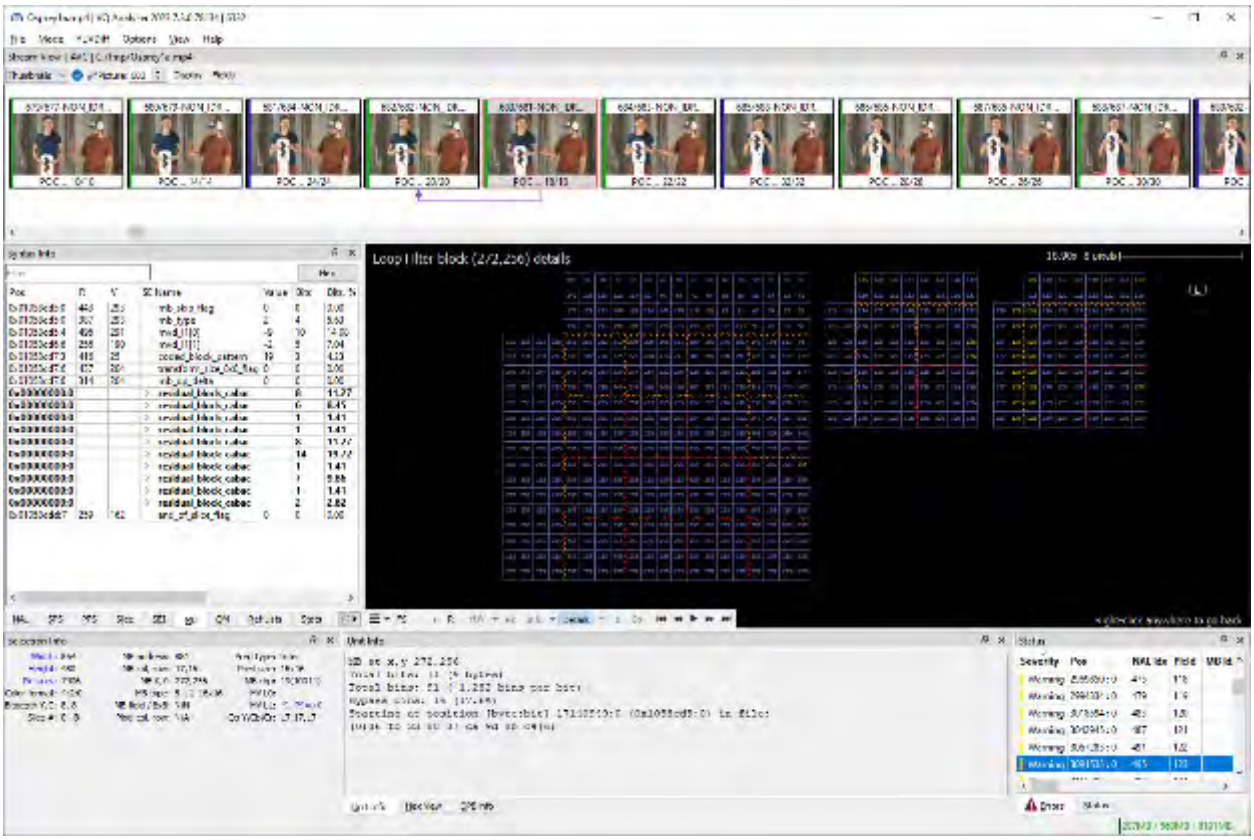


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

338. For another example, on information and belief, Amazon performs a method of video coding in a manner that is covered by claim 23 of the ’891 Patent for Amazon.com

advertisements, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 217.

339. When encoding Amazon.com advertisements, on information and belief, Amazon performs performing an adaptive block boundary filtering operation on a block boundary formed between a first decoded image block on a first side of the block boundary and a second decoded image block on a second side of the block boundary using an adaptive block boundary filter, the first decoded image block having been encoded using a first type of prediction encoding method and the second decoded image block having been encoded using a second type of prediction encoding method, the first type of prediction encoding method and the second type of prediction encoding method being selected from a group of prediction encoding methods comprising at least: intra coding, copy coding, motion-compensated prediction coding, and not-coded coding, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info

Filter | Hex

SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	1
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	2
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	0
deblocking_filter_control_present_flag	1
constrained_intra_pred_flag	0
redundant_pic_cnt_present_flag	0
transform_8x8_mode_flag	1
pic_scaling_matrix_present_flag	0
second_chroma_qp_index_offset	0
rbsp_stop_one_bit	1
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info

Slice #0 size in bits: 55024 SLICE_NONIDR

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	6
pic_parameter_set_id	0
frame_num	7
pic_order_cnt_lsb	18
delta_pic_order_cnt_bottom	0
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	1
num_ref_idx_l1_active_minus1	0
ref_pic_list_reordering()	
cabac_init_idc	0
slice_qp_delta	-9
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	0
slice_beta_offset_div2	0
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info | 6332

Filter | Hex

Pos	R	V	SE Name	Value	Bits	Bi
0x01058cd5:0	443	293	mb_skip_flag	0	0	0.0
0x01058cd5:0	387	293	mb_type	2	4	5.6
0x01058cd5:4	496	291	mvd_l1[0]	-9	10	14
0x01058cd6:6	256	190	mvd_l1[1]	-2	5	7.0
0x01058cd7:3	416	25	coded_block_pattern	19	3	4.2
0x01058cd7:6	437	204	transform_size_8x8_flag	0	0	0.0
0x01058cd7:6	314	204	mb_qp_delta	0	0	0.0
0x00000000:0			**residual_block_cabac**		**8**	**11**
0x00000000:0			**residual_block_cabac**		**6**	**8.4**
0x00000000:0			**residual_block_cabac**		**1**	**1.4**
0x00000000:0			**residual_block_cabac**		**1**	**1.4**
0x00000000:0			**residual_block_cabac**		**8**	**11**
0x00000000:0			**residual_block_cabac**		**14**	**19**
0x00000000:0			**residual_block_cabac**		**1**	**1.4**
0x00000000:0			**residual_block_cabac**		**7**	**9.8**
0x00000000:0			**residual_block_cabac**		**1**	**1.4**
0x00000000:0			**residual_block_cabac**		**2**	**2.8**
0x01058cdd:7	259	162	end_of_slice_flag	0	0	0.0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stat

Selection Info | 6332

Width:	854	MB address:	881	Pred type:	Inter
Height:	480	MB col, row:	17,16	Pred size:	16x16
Pictures:	7406	MB X,Y:	272,256	MB cbp:	19(10011)
Color format:	4:2:0	MB type:	B_L1_16x16	MV L0:	-
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/N	MV L1:	-9,-79=>0
Slice #:	0 - B	Pixel col, row:	N/A	Qp Y/Cb/Cr:	17,17,17

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info | 6332

Filter | Hex

Pos	R	V	SE Name	Value	Bits	Bits, %	De
0x01058d17:1	350	317	mb_skip_flag	1	1	100.00	Ar
0x01058d17:2	316	251	end_of_slice_flag	0	0	0.00	Te

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stat

Selection Info | 6332

Width:	854	MB address:	889	Pred type:	Inter
Height:	480	MB col, row:	25,16	Pred size:	16x16
Pictures:	7406	MB X,Y:	400,256	MB cbp:	0(0)
Color format:	4:2:0	MB type:	Skip	MV L0:	0,0=>0
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/N	MV L1:	0,0=>0
Slice #:	0 - B	Pixel col, row:	N/A	Qp Y/Cb/Cr:	17,17,17

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info

Filter | Hex

Pos	R	V	SE Name	Value	Bits	Bits, %	De
0x0105962b:1	478	464	mb_skip_flag	1	3	100.00	Ar
0x0105962b:4	264	153	end_of_slice_flag	0	0	0.00	Ter

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stat

Selection Info

Width:	854	MB address:	1111	Pred type:	Inter
Height:	480	MB col, row:	31,20	Pred size:	16x16
Pictures:	7406	MB X,Y:	496,320	MB cbp:	0(0)
Color format:	4:2:0	MB type:	Skip	MV L0:	9,9=>0
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/N	MV L1:	-4,-10=>0
Slice #:	0 - B	Pixel col, row:	496,320	Qp Y/Cb/Cr:	17,17,17

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info | 6332

Filter | Hex

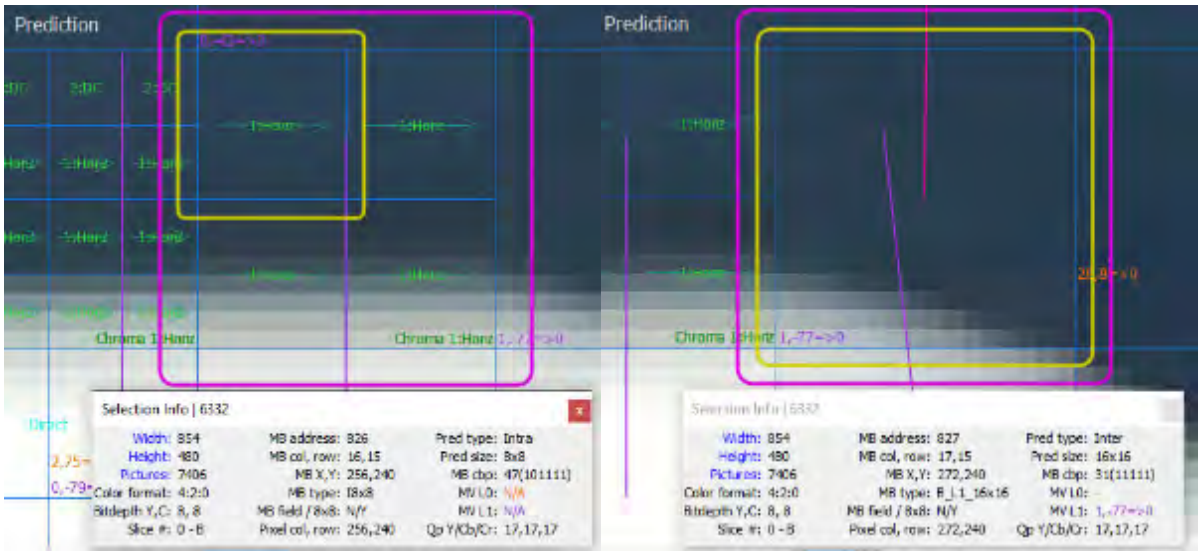
Pos	R	V	SE Name	Value
0x01058bca:7	492	182	mb_skip_flag	0
0x01058bca:7	382	182	mb_type	23
0x01058bcc:1	390	382	transform_size_8x8_flag	1
0x01058bcc:3	344	312	prev_intra8x8_pred_mode_flag[0]	0
0x01058bcc:4	270	206	rem_intra8x8_pred_mode[0]	1
0x01058bcc:7	256	0	prev_intra8x8_pred_mode_flag[1]	1
0x01058bcd:0	280	0	prev_intra8x8_pred_mode_flag[2]	1
0x01058bcd:1	338	0	prev_intra8x8_pred_mode_flag[3]	1
0x01058bcd:2	420	1	intra_chroma_pred_mode	1
0x01058bcd:3	380	3	coded_block_pattern	47
0x01058bcd:7	356	61	mb_qp_delta	0
0x00000000:0			**residual_block_cabac**	
0x00000000:0			**residual_block_cabac**	
0x00000000:0			**residual_block_cabac**	
0x00000000:0			**residual_block_cabac**	
0x00000000:0			**residual_block_cabac**	
0x00000000:0			**residual_block_cabac**	
0x00000000:0			**residual_block_cabac**	
0x01058be3:2	275	166	end_of_slice_flag	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stat

Selection Info | 6332

Width:	854	MB address:	826	Pred type:	Intra
Height:	480	MB col, row:	16,15	Pred size:	8x8
Pictures:	7406	MB X,Y:	256,240	MB cbp:	47(101111)
Color format:	4:2:0	MB type:	I8x8	MV L0:	N/A
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/Y	MV L1:	N/A
Slice #:	0 - B	Pixel col, row:	N/A	Qp Y/Cb/Cr:	17,17,17

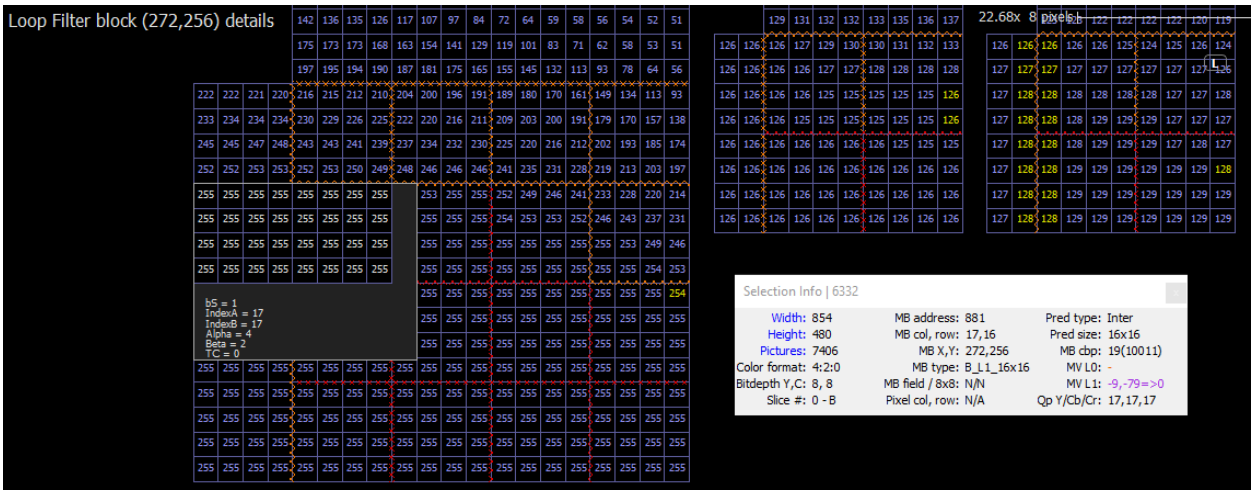
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



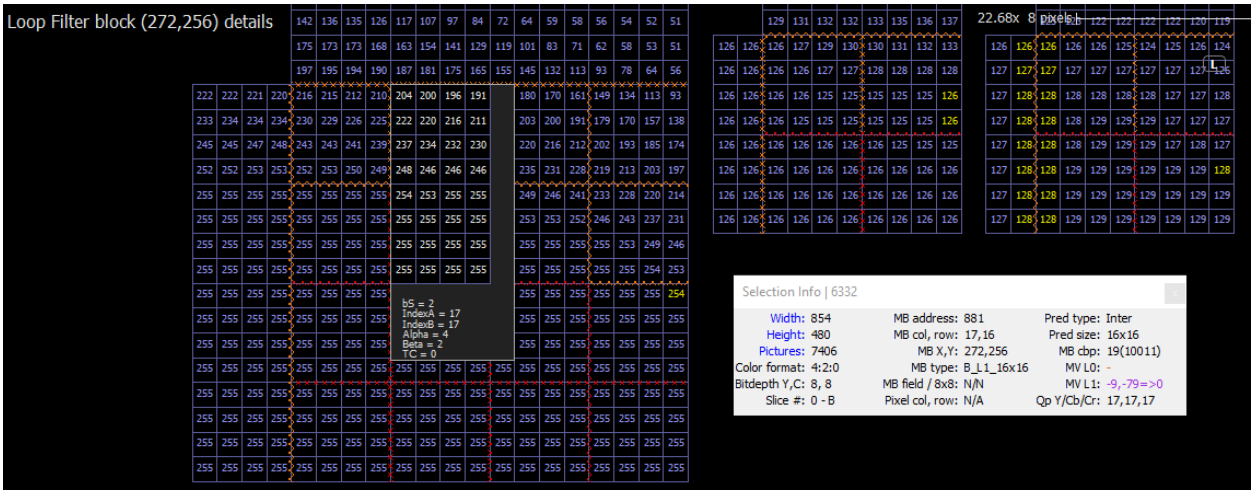
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

340. When encoding Amazon.com advertisements, on information and belief, Amazon performs examining, by the adaptive block boundary filter, the type of the first prediction encoding method and the type of the second prediction encoding method, as demonstrated in the screenshots below using VQ Analyzer software.

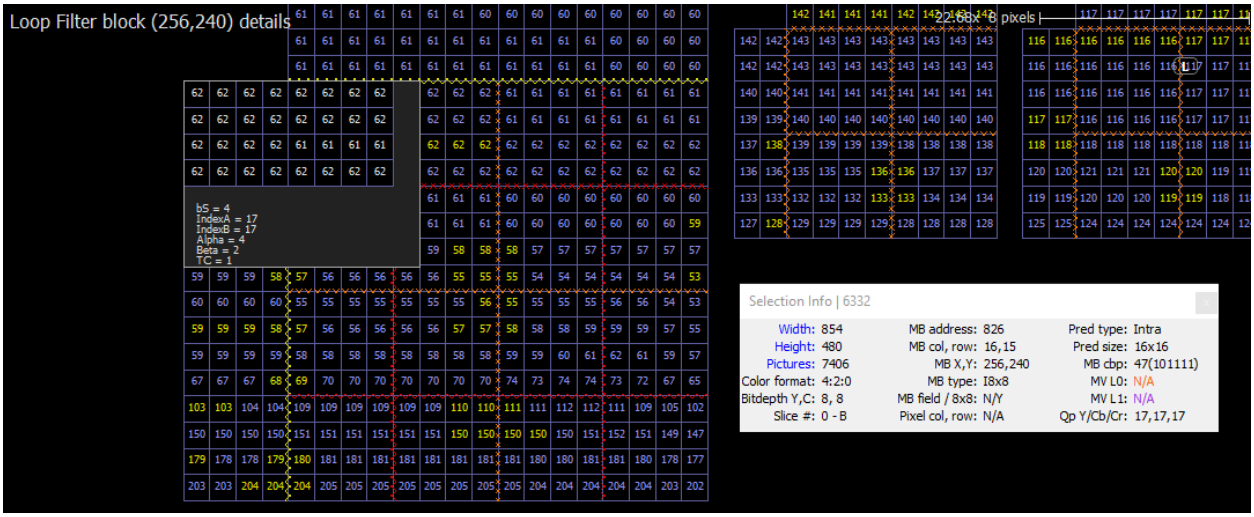
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



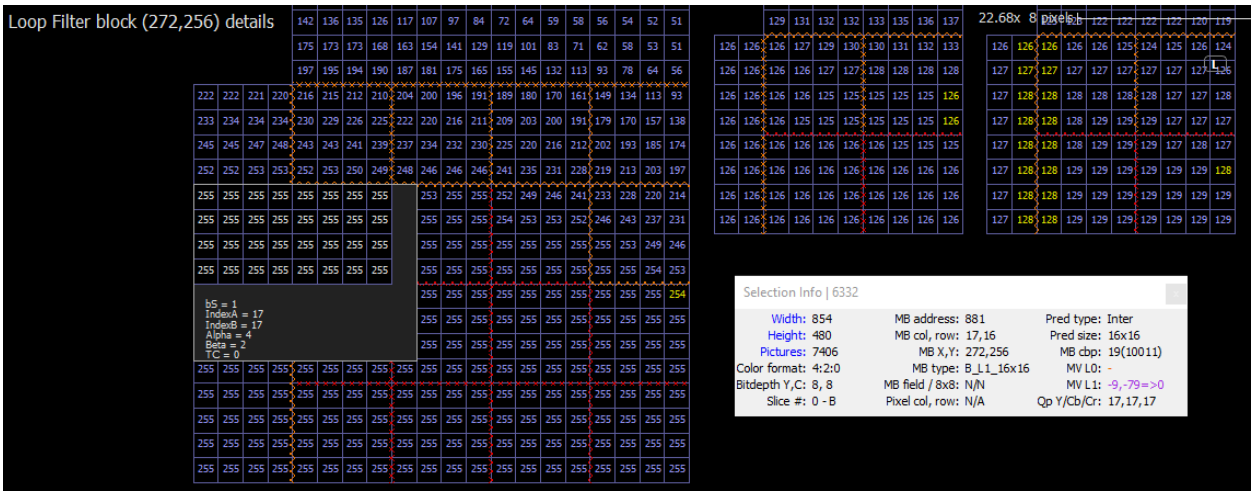
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



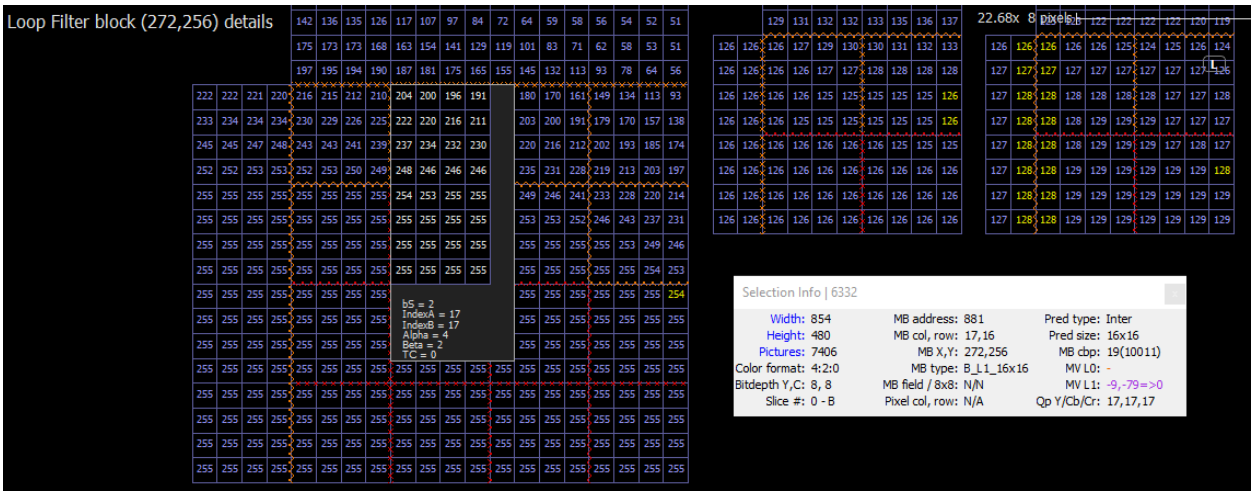
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

341. When encoding Amazon.com advertisements, on information and belief, Amazon performs determining, by the adaptive block boundary filter, a first number of pixels to be examined on the first side of the block boundary and a second number of pixels to be examined on the second side of the block boundary, as a parameter of the filtering operation, based on the types of the first and second prediction encoding methods, as demonstrated in the screenshots below using VQ Analyzer software.

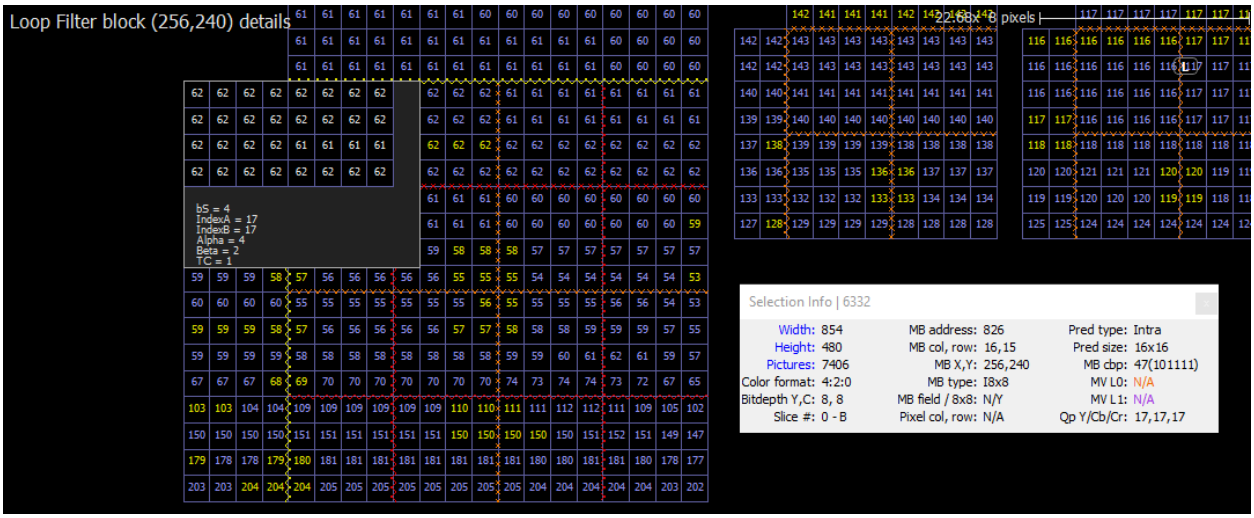
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

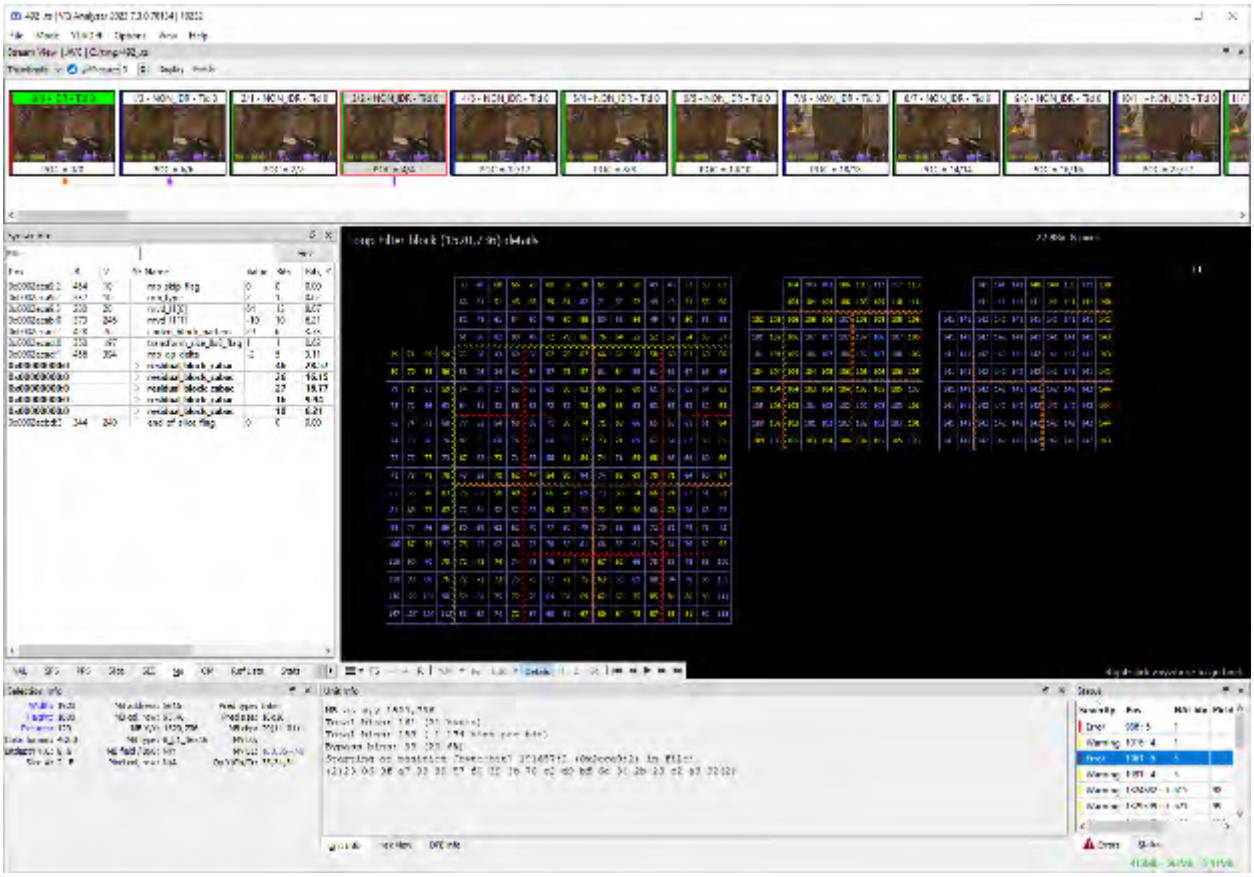


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

342. For another example, on information and belief, Amazon performs a method of video coding in a manner that is covered by claim 23 of the ’891 Patent for Twitch.tv video, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 222.

343. When encoding Twitch.tv video, on information and belief, Amazon performs performing an adaptive block boundary filtering operation on a block boundary formed between a first decoded image block on a first side of the block boundary and a second decoded image

block on a second side of the block boundary using an adaptive block boundary filter, the first decoded image block having been encoded using a first type of prediction encoding method and the second decoded image block having been encoded using a second type of prediction encoding method, the first type of prediction encoding method and the second type of prediction encoding method being selected from a group of prediction encoding methods comprising at least: intra coding, copy coding, motion-compensated prediction coding, and not-coded coding, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info

Filter | Hex

SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	0
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	4
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	-3
deblocking_filter_control_present_flag	1
constrained_intra_pred_flag	0
redundant_pic_cnt_present_flag	0
transform_8x8_mode_flag	1
pic_scaling_matrix_present_flag	0
second_chroma_qp_index_offset	-3
rbsp_stop_one_bit	1

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info

Slice #2 size in bits: 13144 **SLICE_NONIDR**

Filter Hex

SE Name	Value
first_mb_in_slice	4080
slice_type	6
pic_parameter_set_id	0
frame_num	2
pic_order_cnt_lsb	4
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	0
num_ref_idx_l1_active_minus1	0
ref_pic_list_reordering()	
cabac_init_idc	0
slice_qp_delta	5
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	-1
slice_beta_offset_div2	-1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info | 10232

Filter | Hex

Pos	R	V	SE Name	Value	Bits	Bi
0x0002eca9:2	454	10	mb_skip_flag	0	0	0.
0x0002eca9:2	332	10	mb_type	2	1	0.
0x0002eca9:3	280	20	mvd_l1[0]	64	13	8.
0x0002ecab:0	373	246	mvd_l1[1]	-10	10	6.
0x0002ecac:2	428	76	coded_block_pattern	29	6	3.
0x0002ecad:0	350	197	transform_size_8x8_flag	1	1	0.
0x0002ecad:1	456	394	mb_qp_delta	-2	5	3.
0x00000000:0			**residual_block_cabac**		**46**	**28**
0x00000000:0			**residual_block_cabac**		**26**	**16**
0x00000000:0			**residual_block_cabac**		**27**	**16**
0x00000000:0			**residual_block_cabac**		**16**	**9.**
0x00000000:0			**residual_block_cabac**		**10**	**6.**
0x0002ecbd:3	344	240	end_of_slice_flag	0	0	0.

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Sta

Selection Info | 10232

Width:	1920	MB address:	5615	Pred type:	Inter
Height:	1080	MB col, row:	95,46	Pred size:	16x16
Pictures:	120	MB X,Y:	1520,736	MB cbp:	29(11101)
Color format:	4:2:0	MB type:	B_L1_16x16	MV L0:	-
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/Y	MV L1:	163,35=>0
Slice #:	2 - B	Pixel col, row:	1467,752	Qp Y/Cb/Cr:	35,31,31

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info | 10232

Filter | Hex

Pos	R	V	SE Name	Value	Bits	Bits, %	De
0x0002f342:1	258	169	mb_skip_flag	1	1	100.00	Ari
0x0002f342:2	492	339	end_of_slice_flag	0	0	0.00	Ter

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Sta

Selection Info | 10232

Width:	1920	MB address:	7356	Pred type:	Inter
Height:	1080	MB col, row:	36,61	Pred size:	16x16
Pictures:	120	MB X,Y:	576,976	MB cbp:	0(0)
Color format:	4:2:0	MB type:	Skip	MV L0:	0,0=>0
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/N	MV L1:	0,0=>0
Slice #:	3 - B	Pixel col, row:	506,990	Qp Y/Cb/Cr:	32,29,29

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info | 10232

Filter | Hex

Pos	R	V	SE Name	Value	Bits	Bits, %	De
0x0002e507:2	263	162	mb_skip_flag	1	1	100.00	Ar
0x0002e507:3	456	325	end_of_slice_flag	0	0	0.00	Te

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Sta

Selection Info | 10232

Width:	1920	MB address:	2785	Pred type:	Inter
Height:	1080	MB col, row:	25,23	Pred size:	16x16
Pictures:	120	MB X,Y:	400,368	MB cbp:	0(0)
Color format:	4:2:0	MB type:	Skip	MV L0:	-2,-29=>0
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/N	MV L1:	-
Slice #:	1 - B	Pixel col, row:	322,404	Qp Y/Cb/Cr:	35,31,31

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info | 10232

Filter | Hex

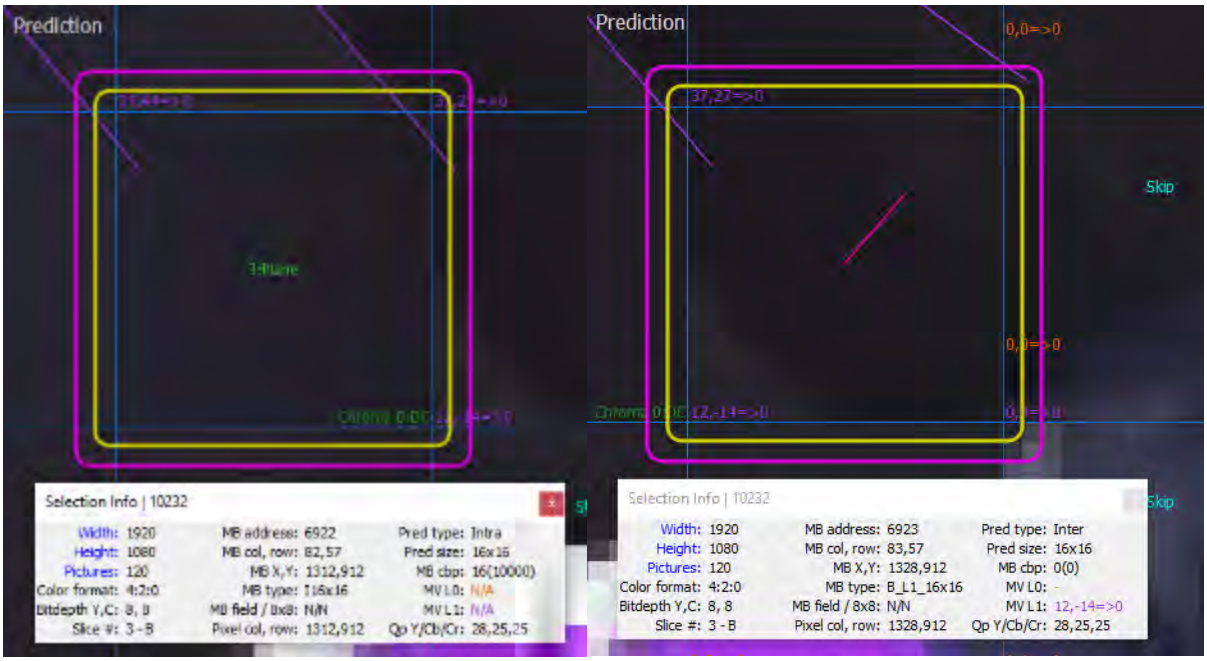
Pos	R	V	SE Name	Value	Bits
0x0002f133:2	335	323	mb_skip_flag	0	2
0x0002f133:4	488	443	mb_type	23	13
0x0002f135:1	480	313	intra_chroma_pred_mode	0	1
0x0002f135:2	390	56	mb_qp_delta	-4	6
0x00000000:0			**residual_block_cabac**		**3**
0x00000000:0			**residual_block_cabac**		**4**
0x00000000:0			**residual_block_cabac**		**1**
0x0002f137:0	370	285	end_of_slice_flag	0	0

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Sta

Selection Info | 10232

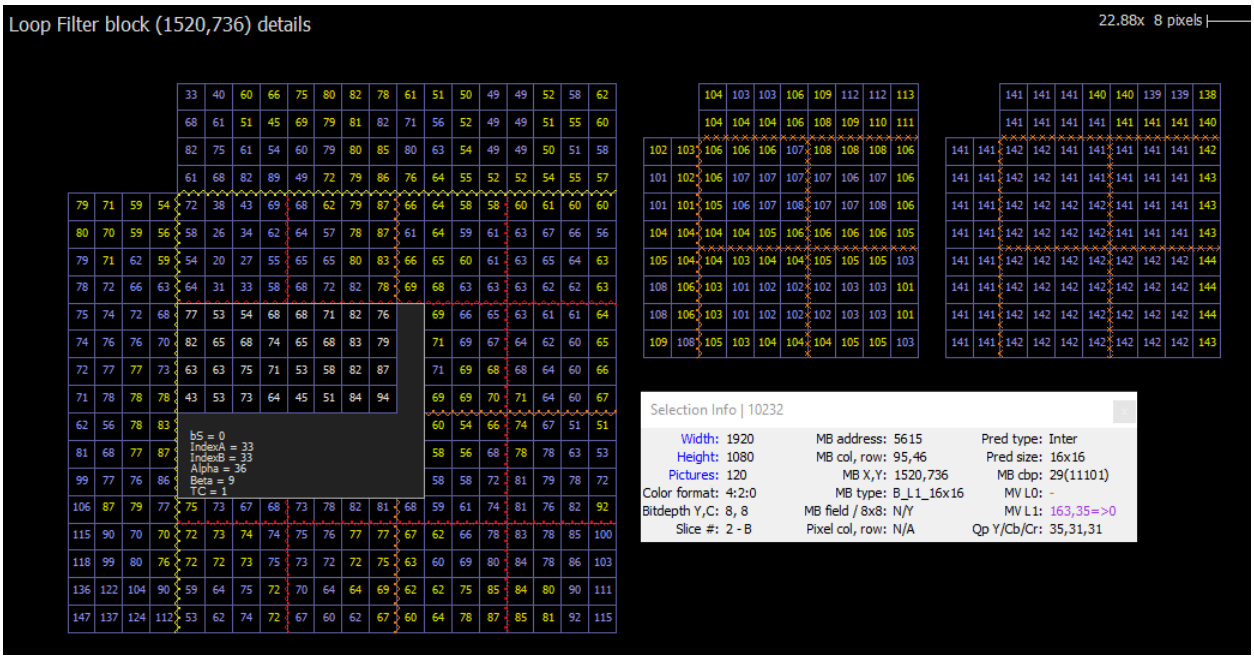
Width:	1920	MB address:	6922	Pred type:	Intra
Height:	1080	MB col, row:	82,57	Pred size:	16x16
Pictures:	120	MB X,Y:	1312,912	MB cbp:	16(10000)
Color format:	4:2:0	MB type:	I16x16	MV L0:	N/A
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/N	MV L1:	N/A
Slice #:	3 - B	Pixel col, row:	N/A	Qp Y/Cb/Cr:	28,25,25

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

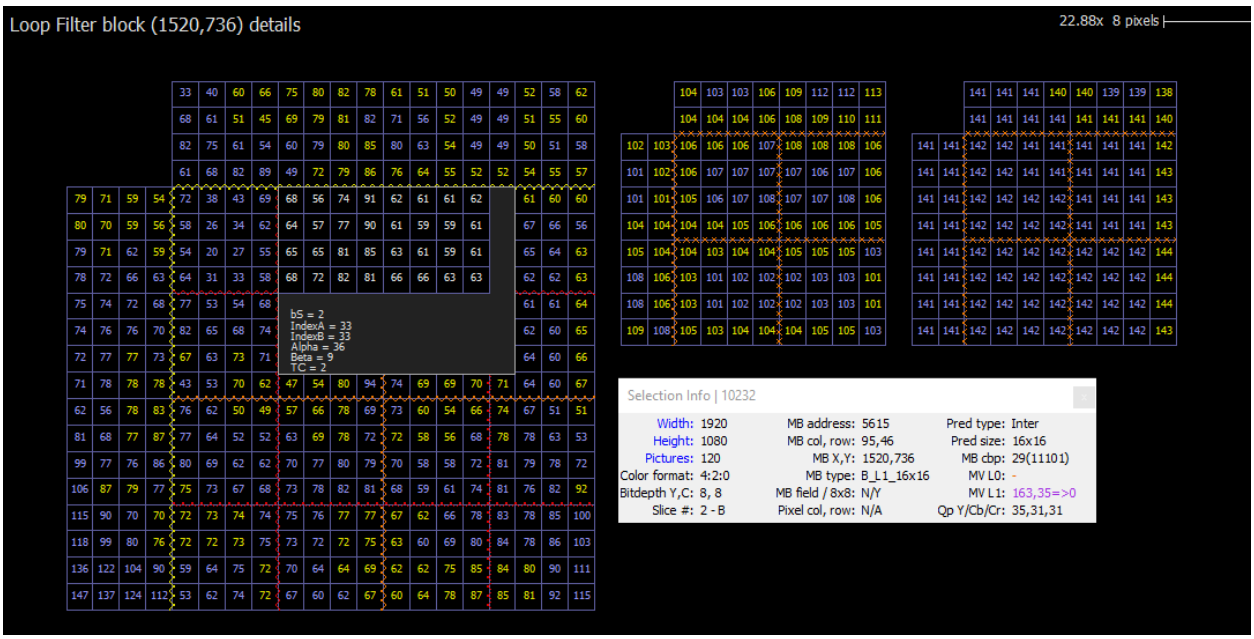


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

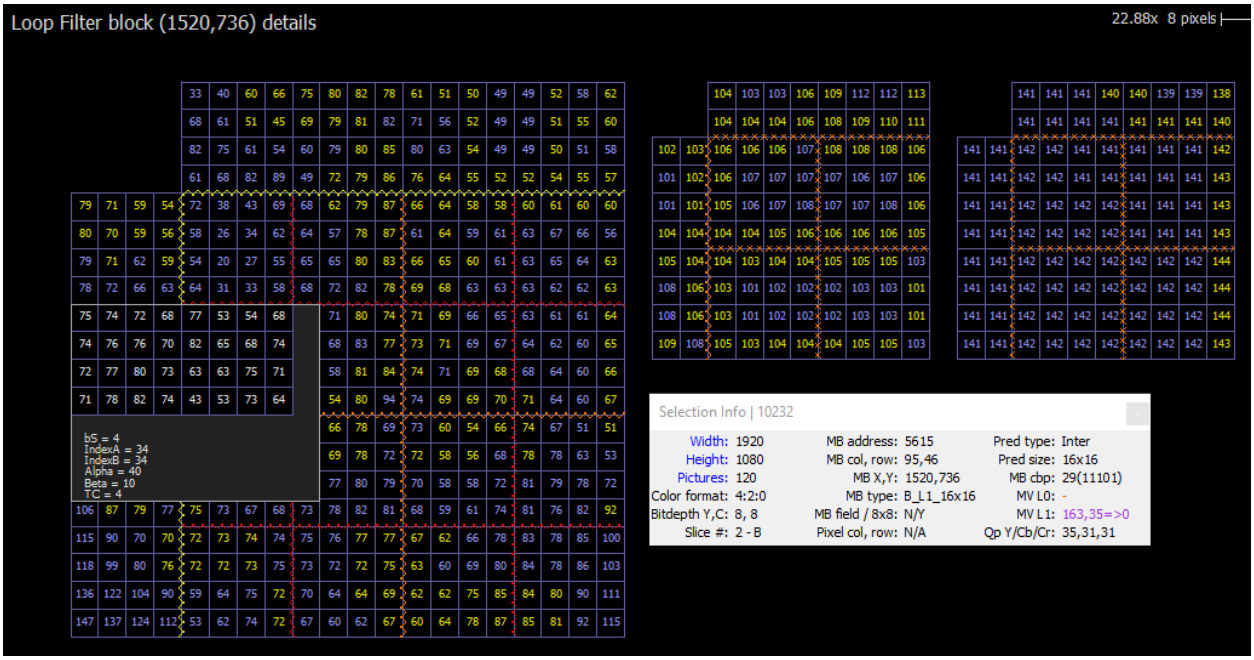
344. When encoding Twitch.tv video, on information and belief, Amazon performs examining, by the adaptive block boundary filter, the type of the first prediction encoding method and the type of the second prediction encoding method, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



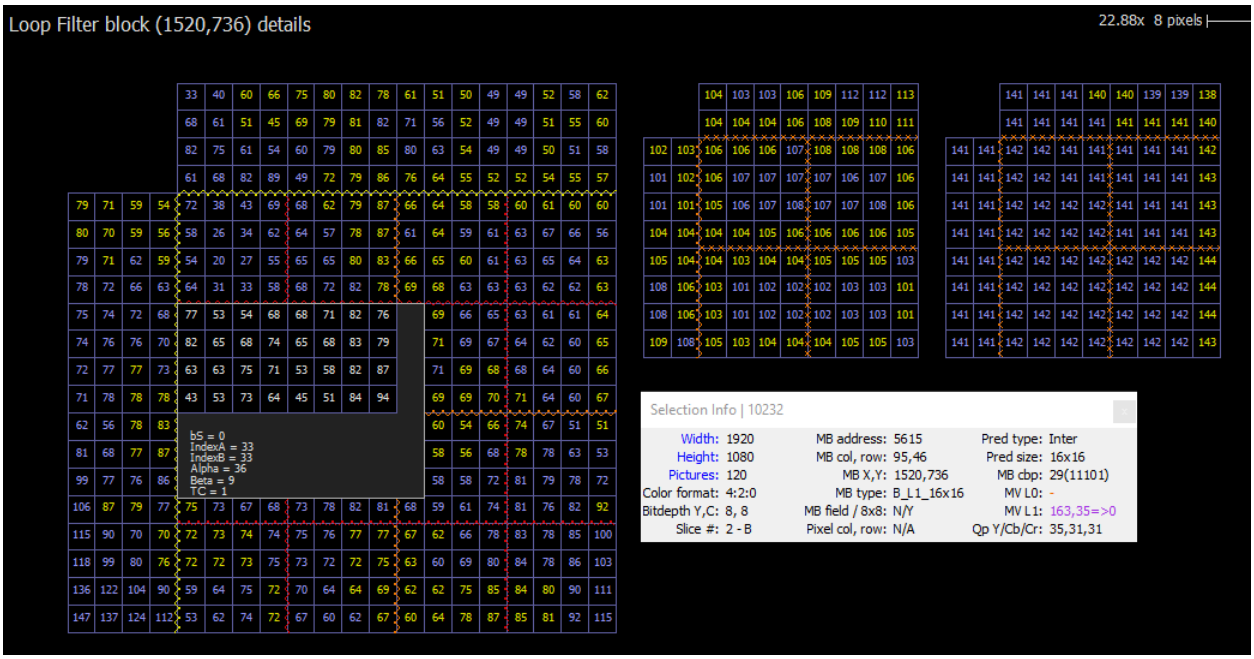
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



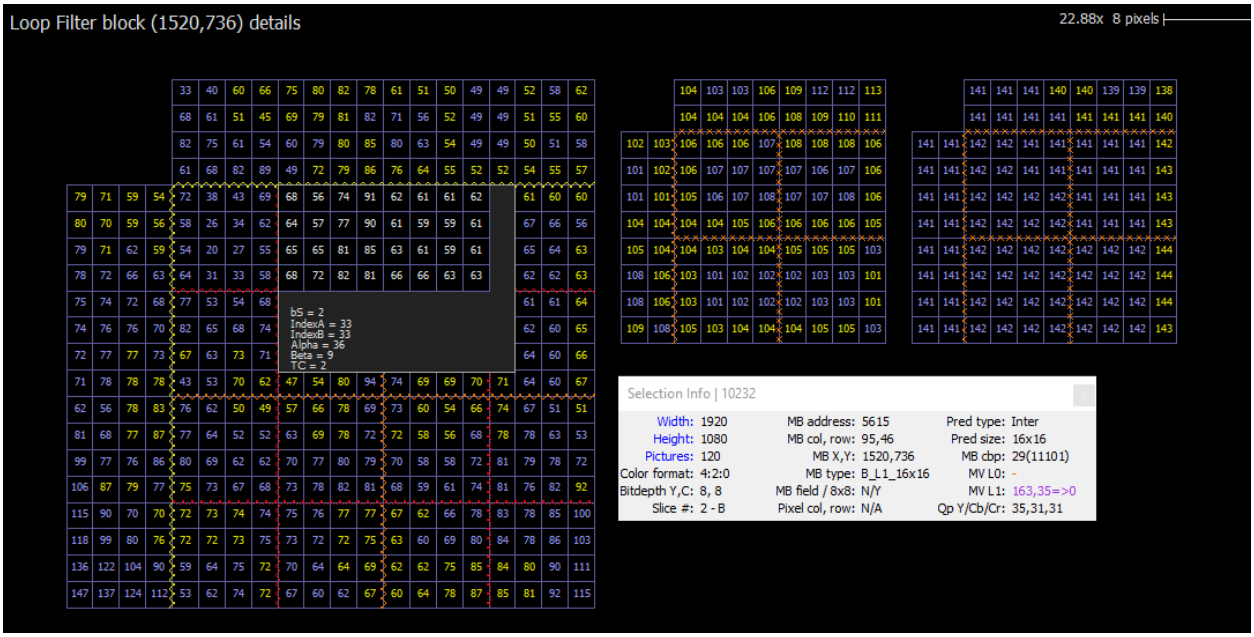
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

345. When encoding Twitch.tv video, on information and belief, Amazon performs determining, by the adaptive block boundary filter, a first number of pixels to be examined on the

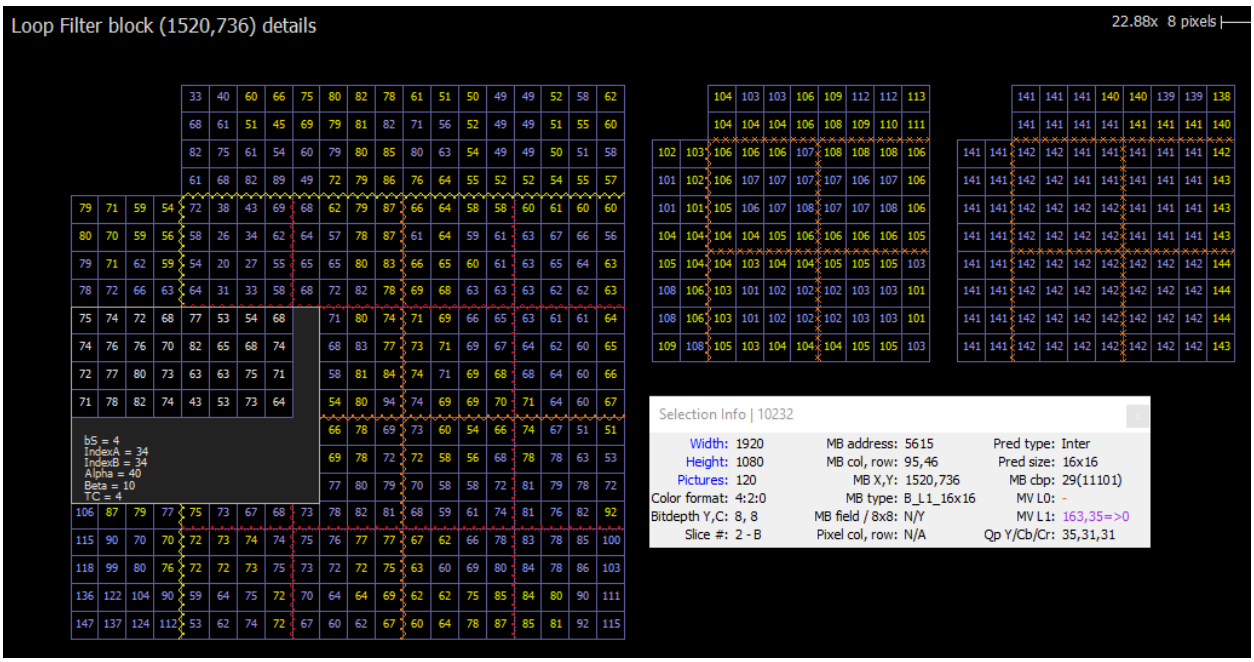
first side of the block boundary and a second number of pixels to be examined on the second side of the block boundary, as a parameter of the filtering operation, based on the types of the first and second prediction encoding methods, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

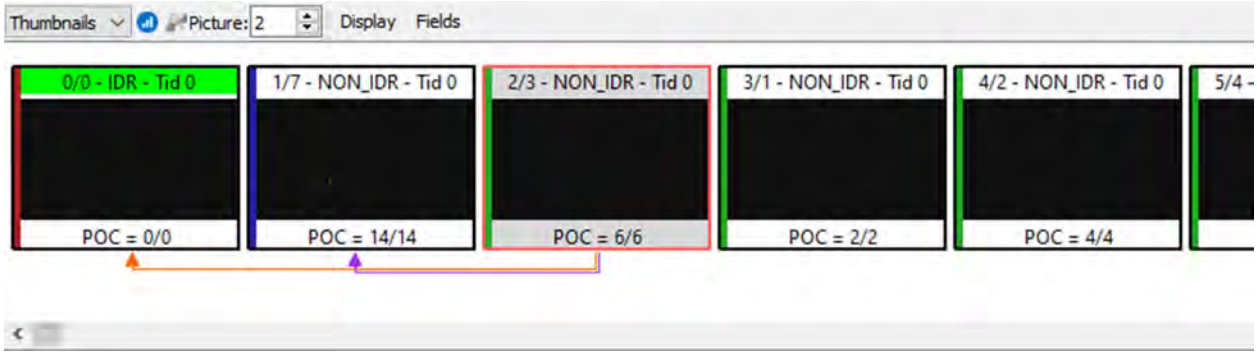
I. Amazon Makes, Imports, Uses, Sells, and/or Offers for Sale Products and Services that Infringe the ’005 Patent

346. The Accused Products infringe one or more claims of the ’005 Patent, including, for example, claim 1.

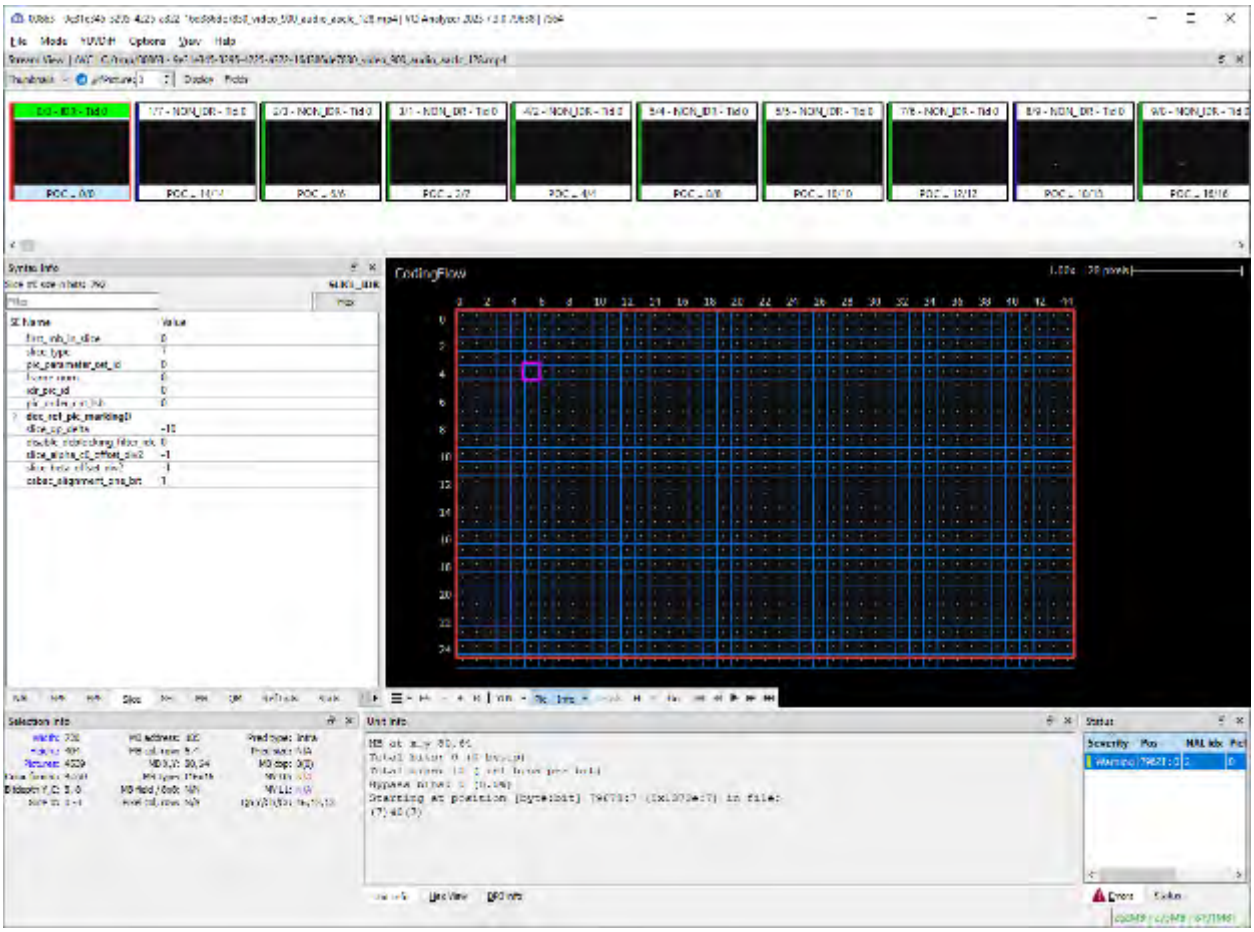
347. As just one example of infringement, on information and belief, Amazon performs a method of encoding a video signal in a manner that is covered by claim 1 of the ’005 Patent for Amazon Prime Video content (such as trailers), as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 212.

348. When encoding Amazon Prime Video trailers, Amazon performs encoding a video signal representing a sequence of pictures to form an encoded video signal comprising temporally independent INTRA pictures and temporally predicted pictures, wherein the INTRA pictures and at least some of the temporally predicted pictures are used to form reference pictures for the

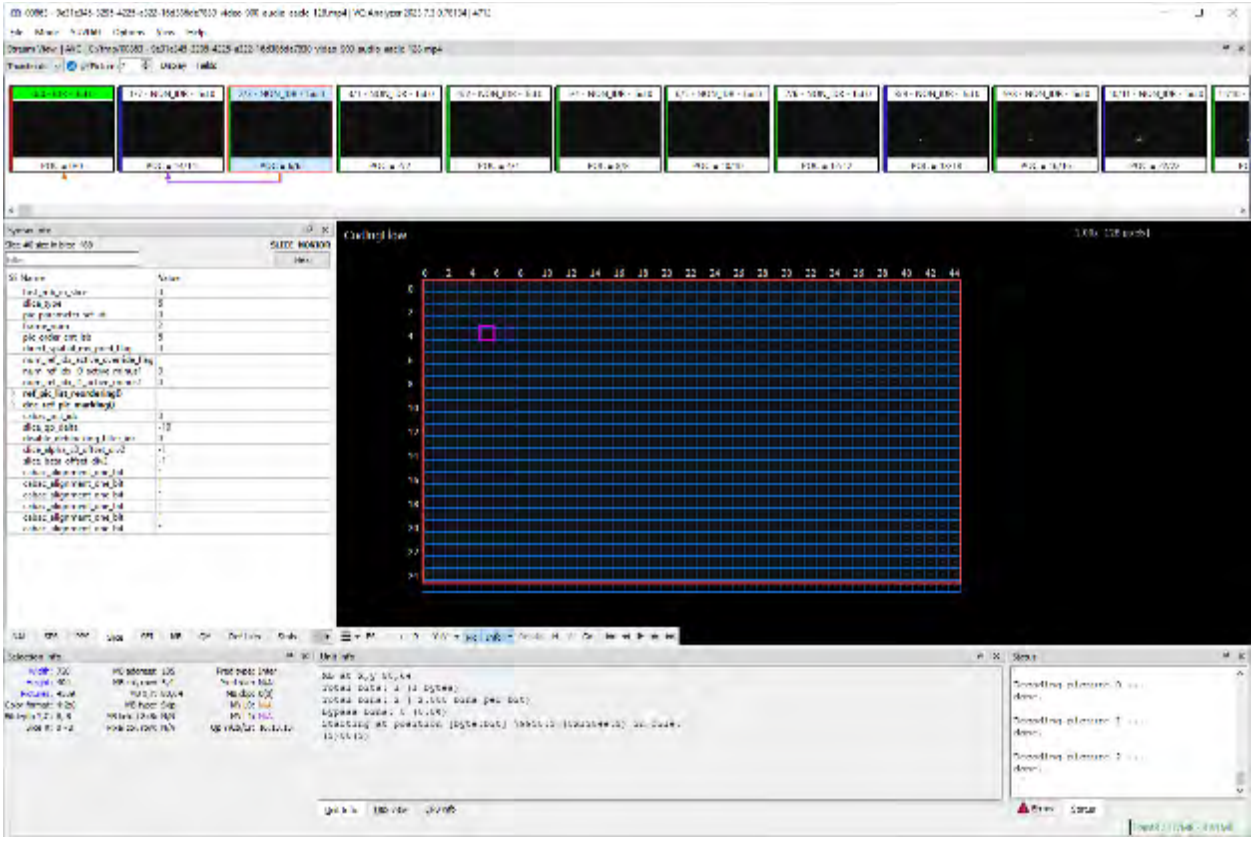
temporal prediction of other pictures in the video sequence, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

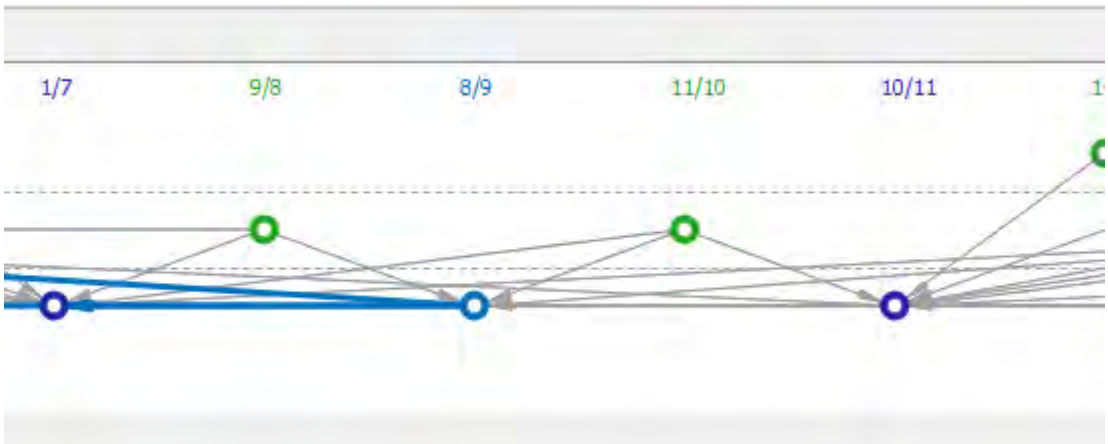


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

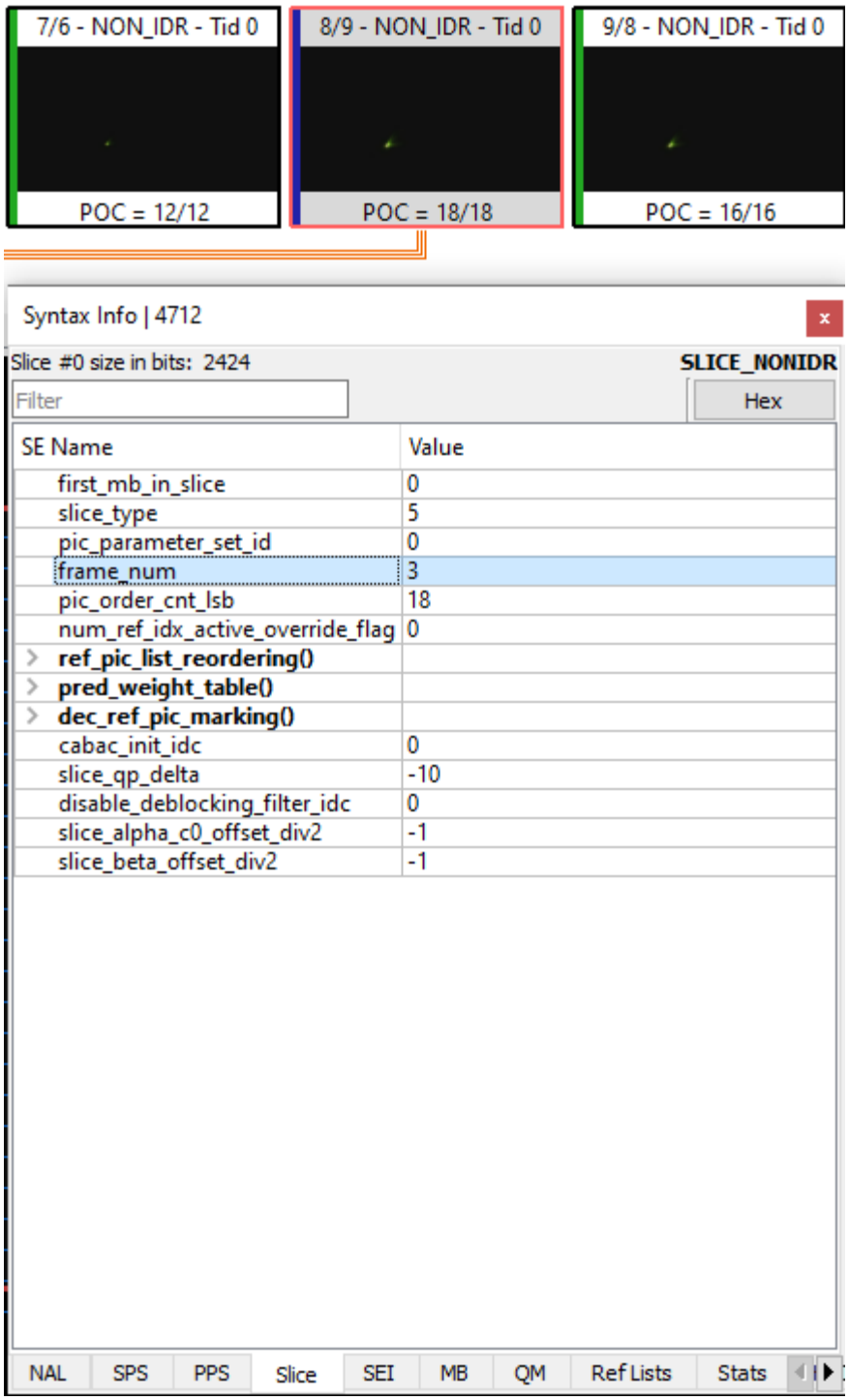


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

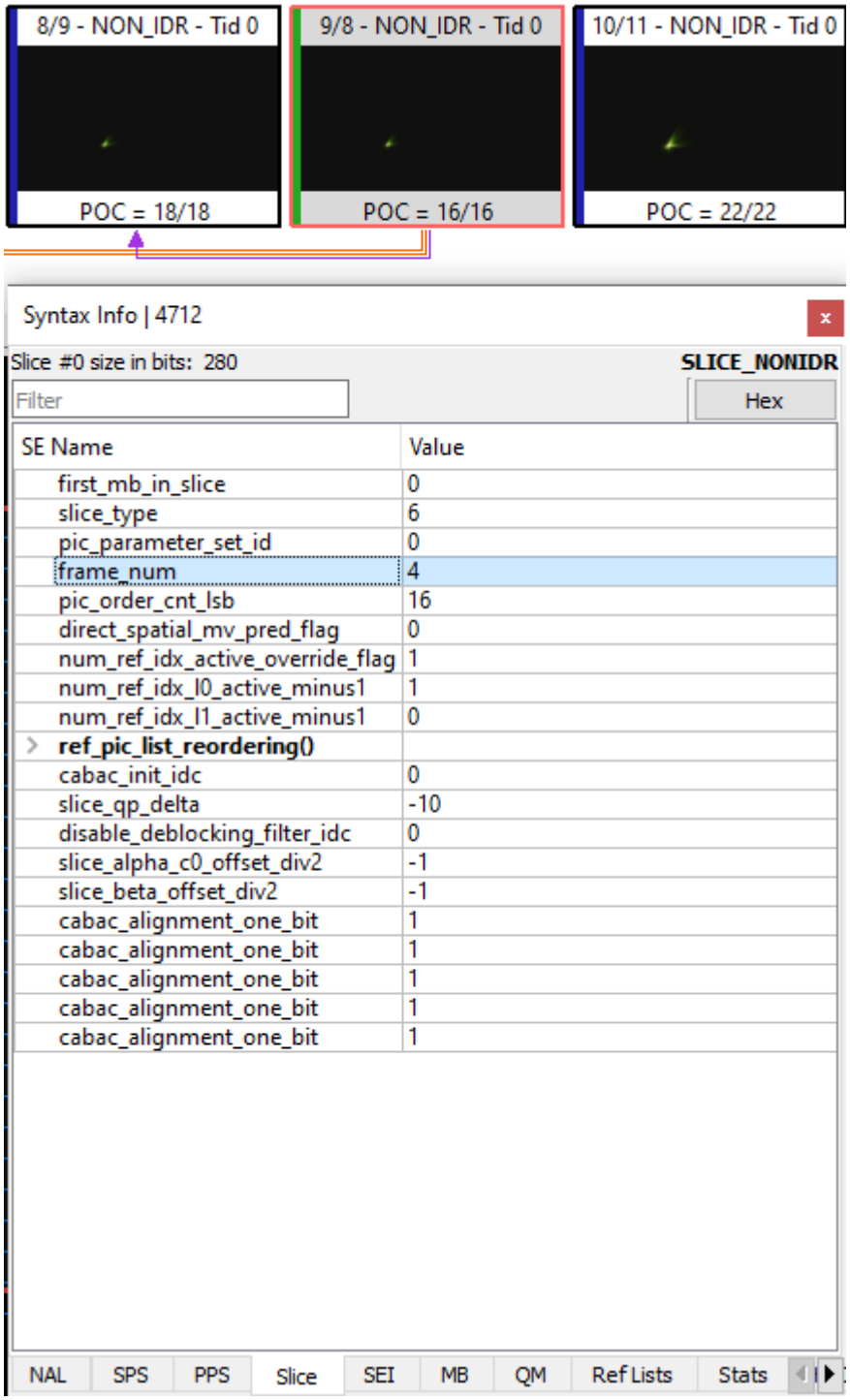
349. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs indicating an encoding order of those pictures used to form reference pictures in the encoded video signal with a sequence indicator having an independent numbering scheme, such that consecutive pictures used to form reference pictures in encoding order are assigned sequence indicator values that differ with respect to each other by a predetermined amount independent of the number of non-reference pictures encoded between successive reference pictures, as demonstrated in the screenshots below using VQ Analyzer software, where the frame number for reference pictures increases by 1 when there is one non-reference picture between reference pictures.



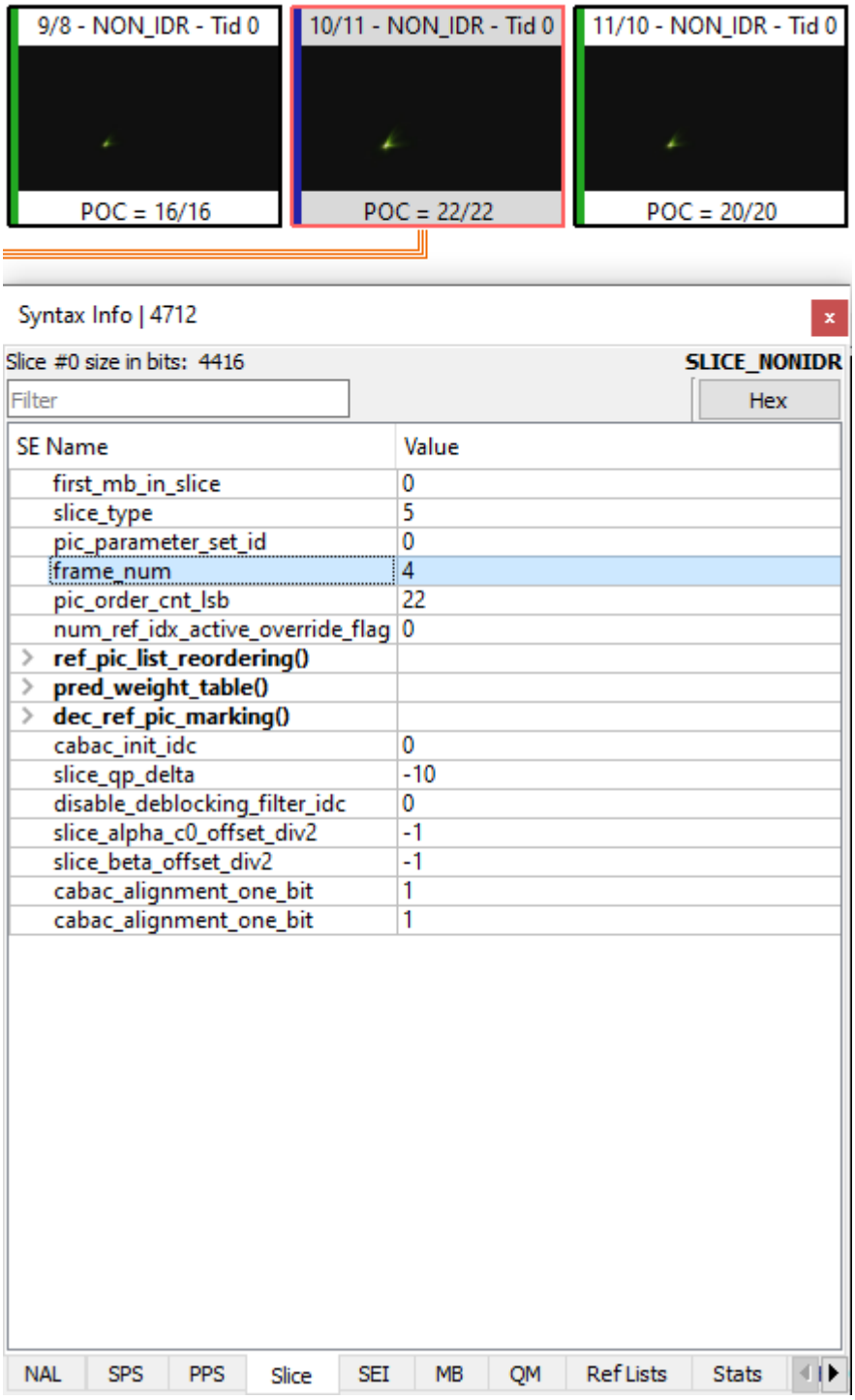
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



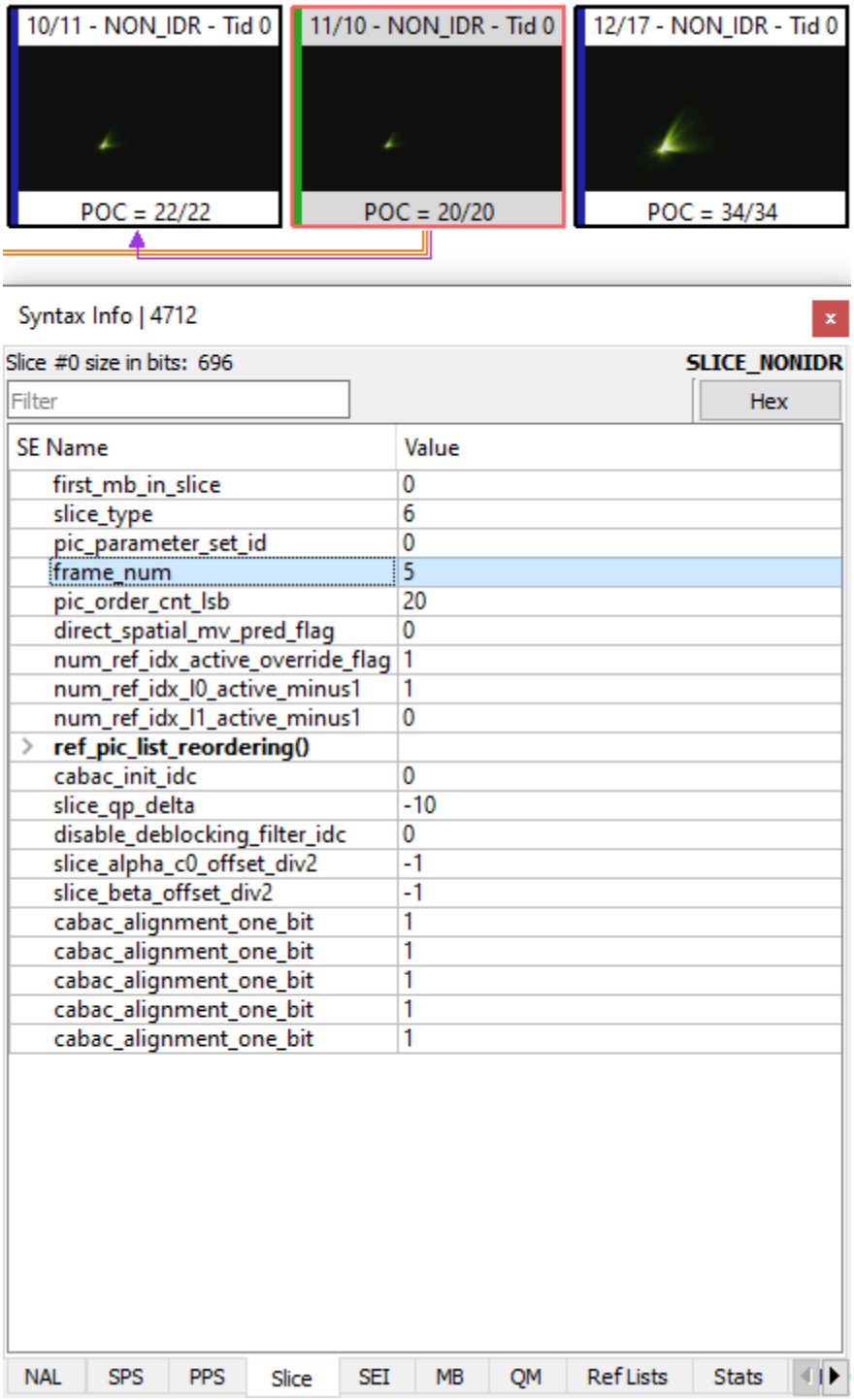
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



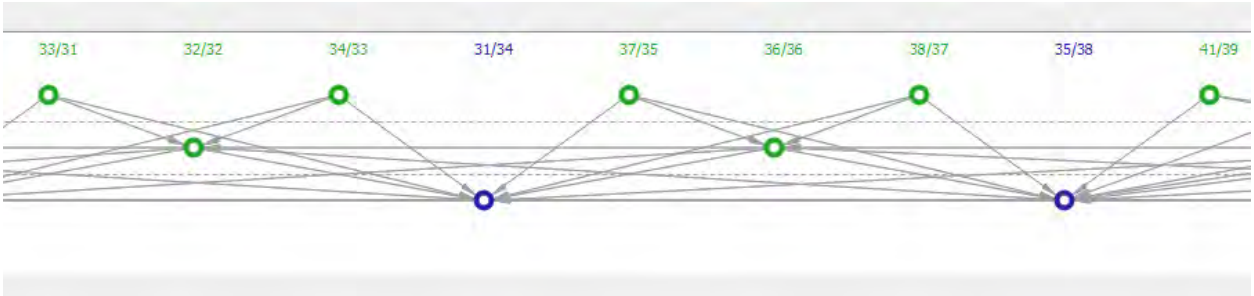
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



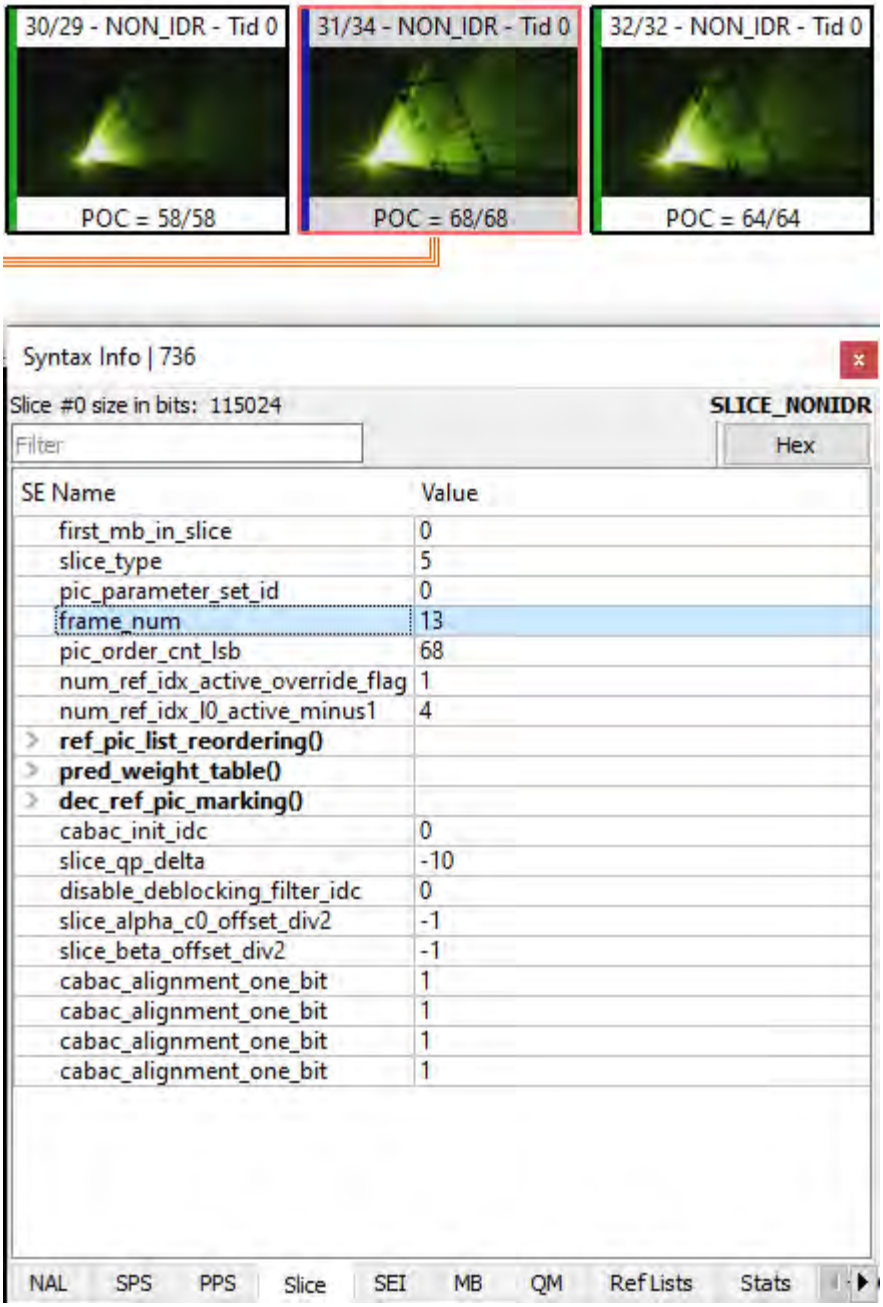
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

350. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs indicating an encoding order of those pictures used to form reference pictures in the

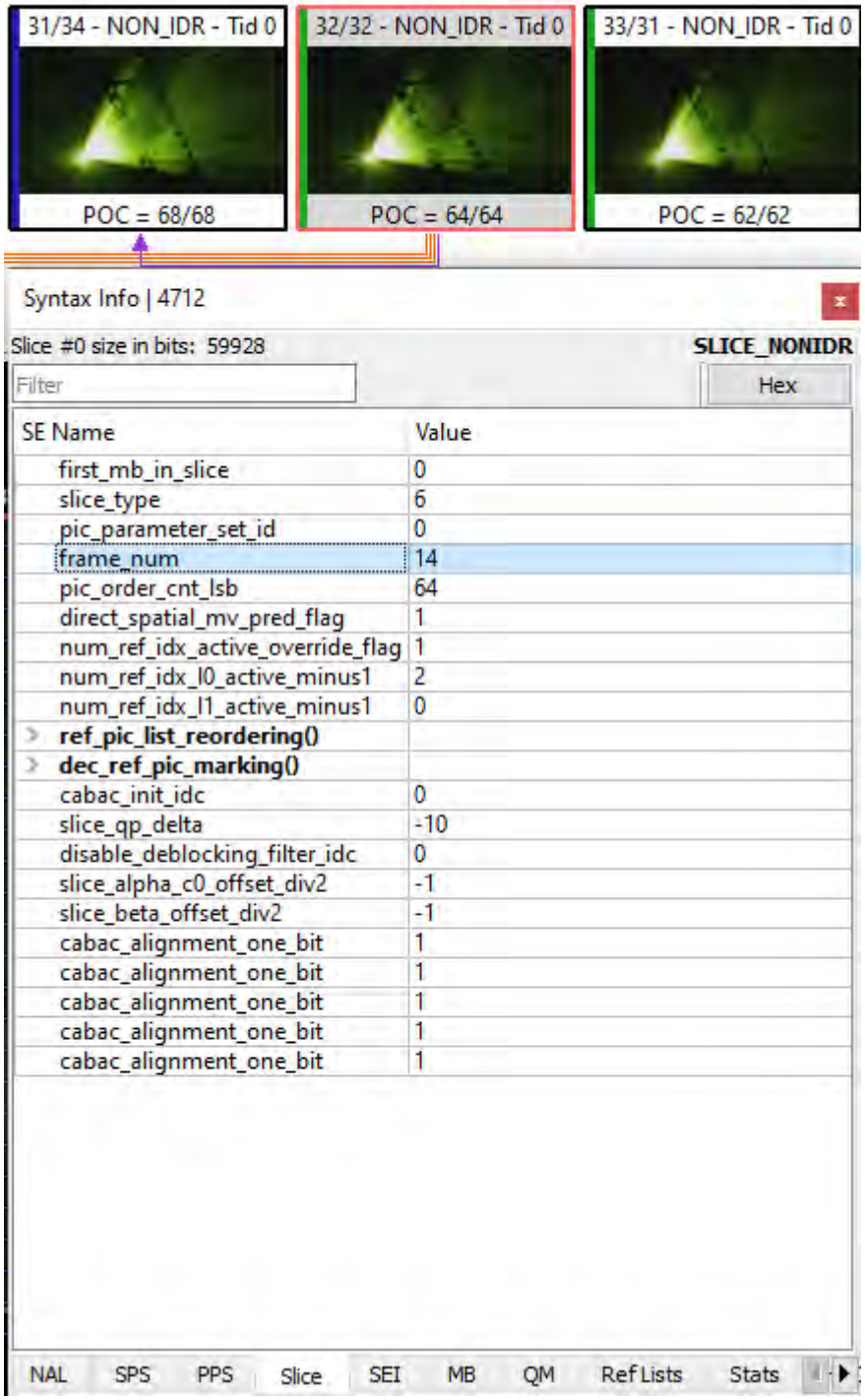
encoded video signal with a sequence indicator having an independent numbering scheme, such that consecutive pictures used to form reference pictures in encoding order are assigned sequence indicator values that differ with respect to each other by a predetermined amount independent of the number of non-reference pictures encoded between successive reference pictures, as demonstrated in the screenshots below using VQ Analyzer software, where the frame number for reference pictures increases by 1 when there are two non-reference pictures between reference pictures.



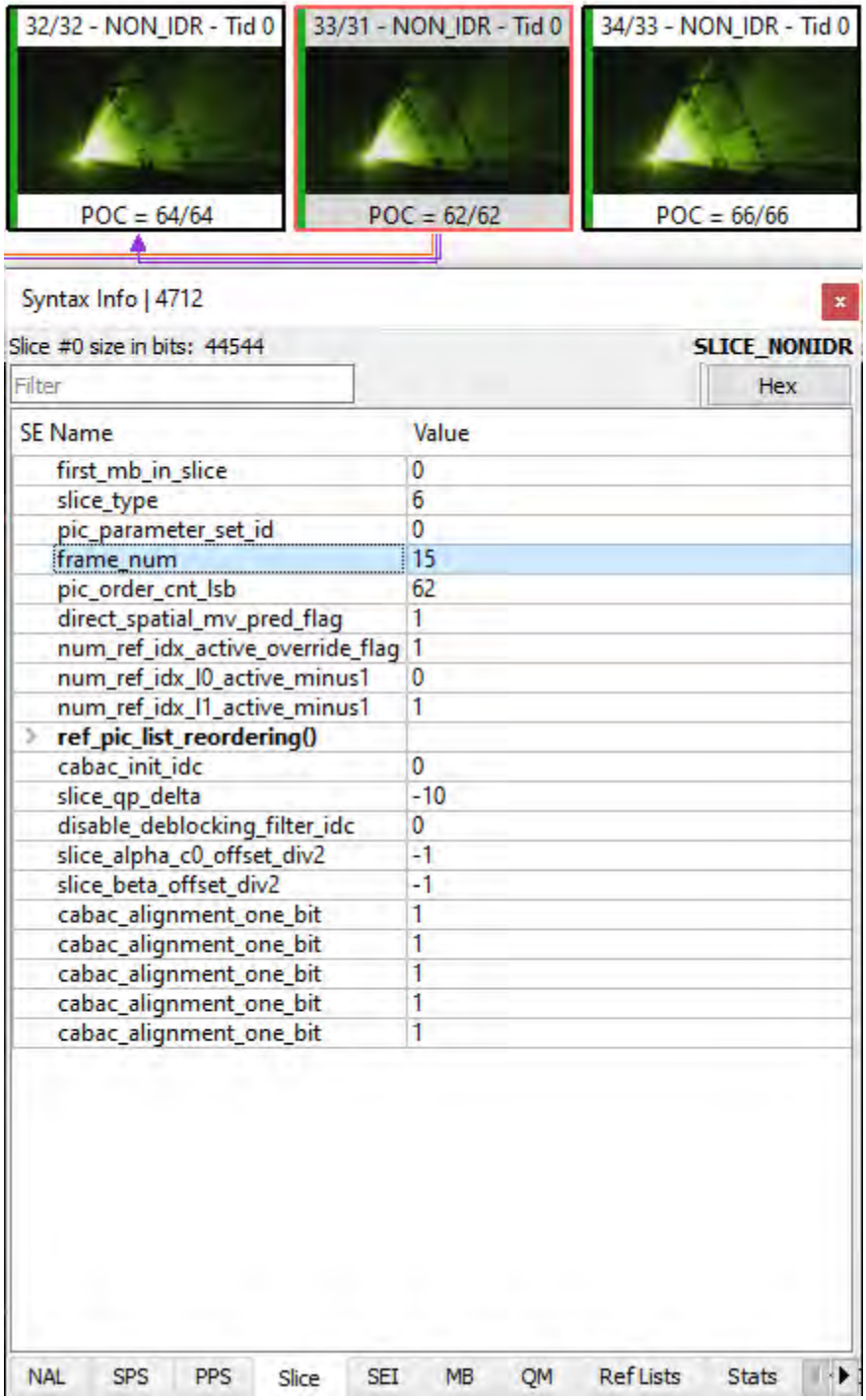
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



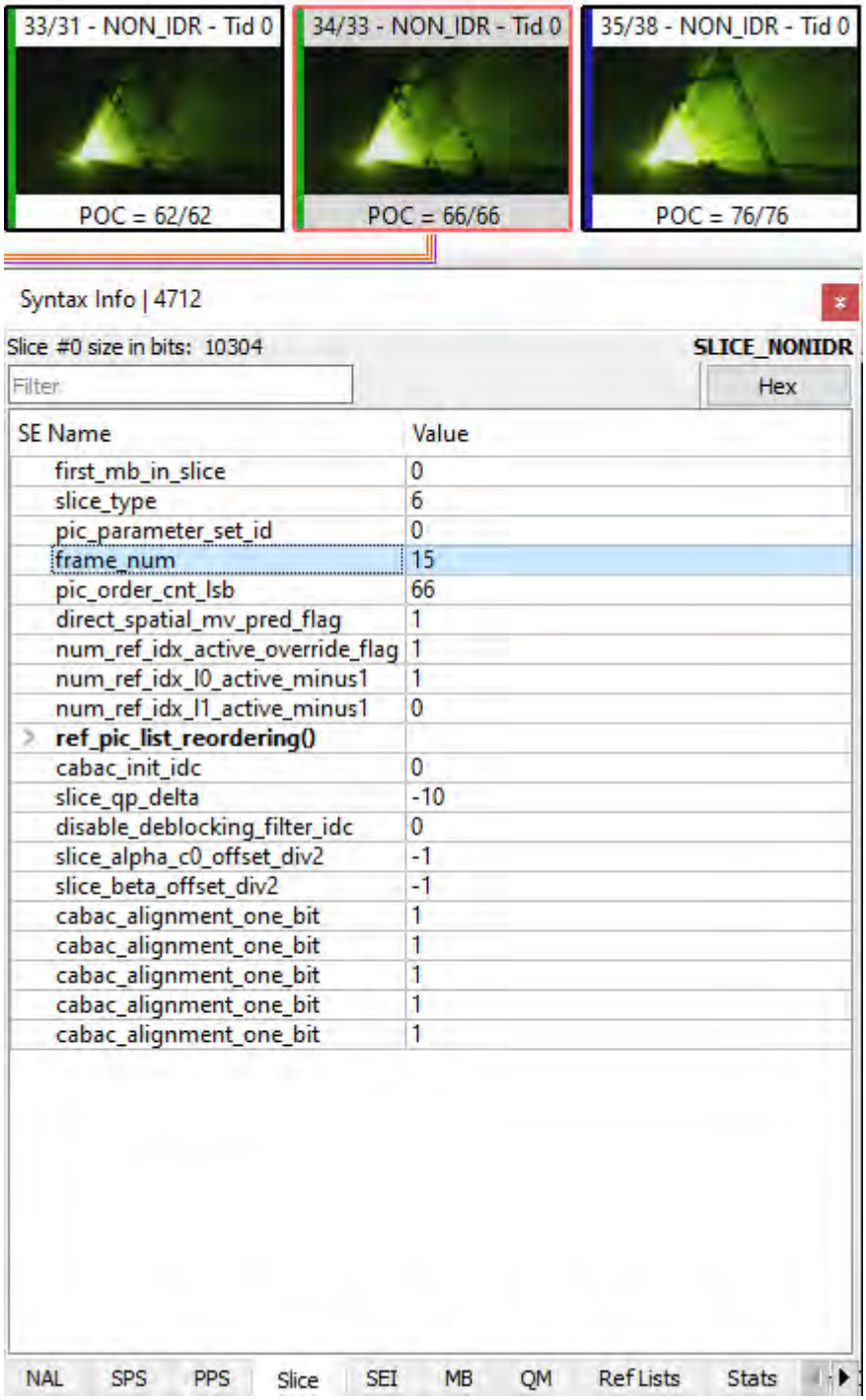
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



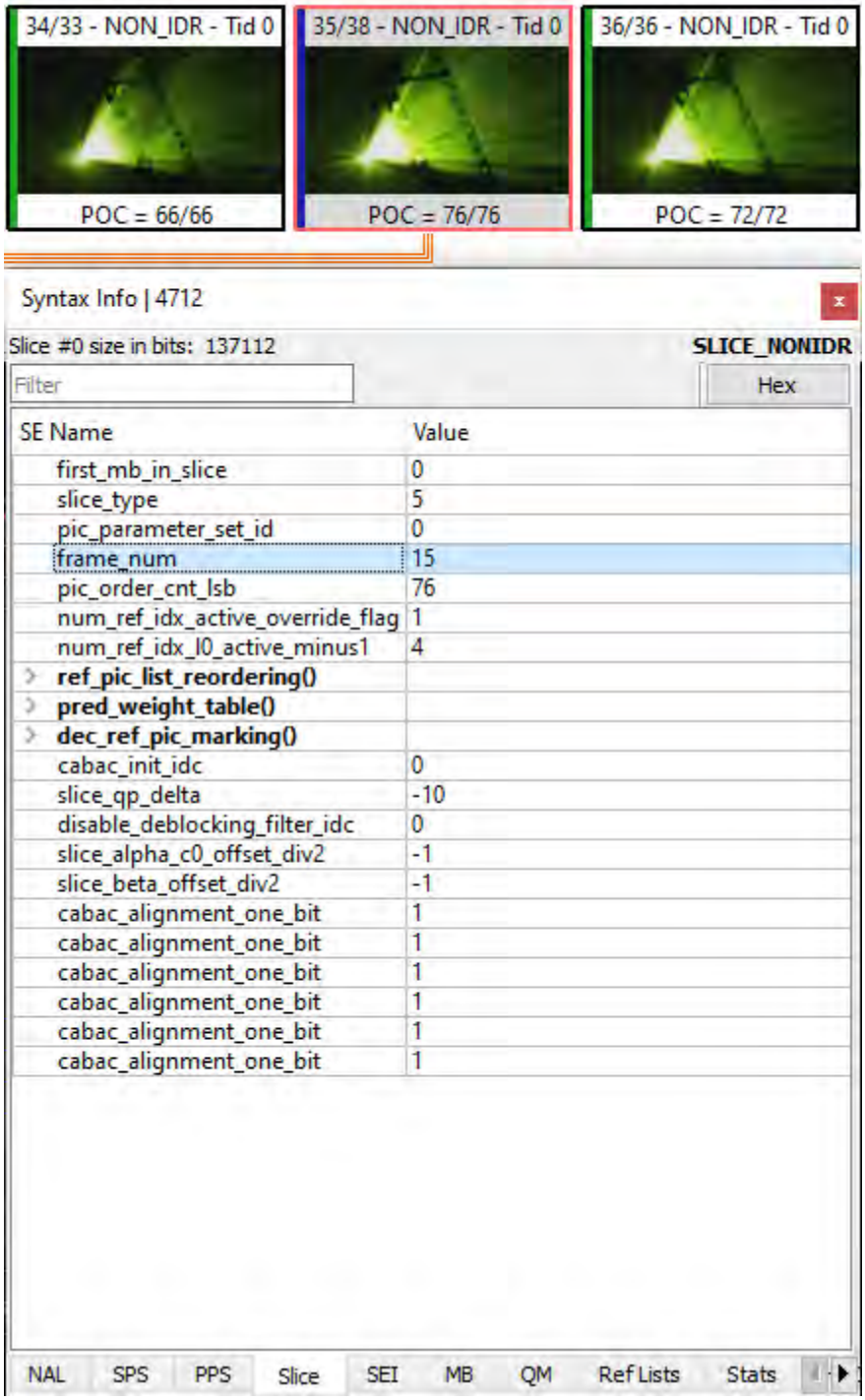
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



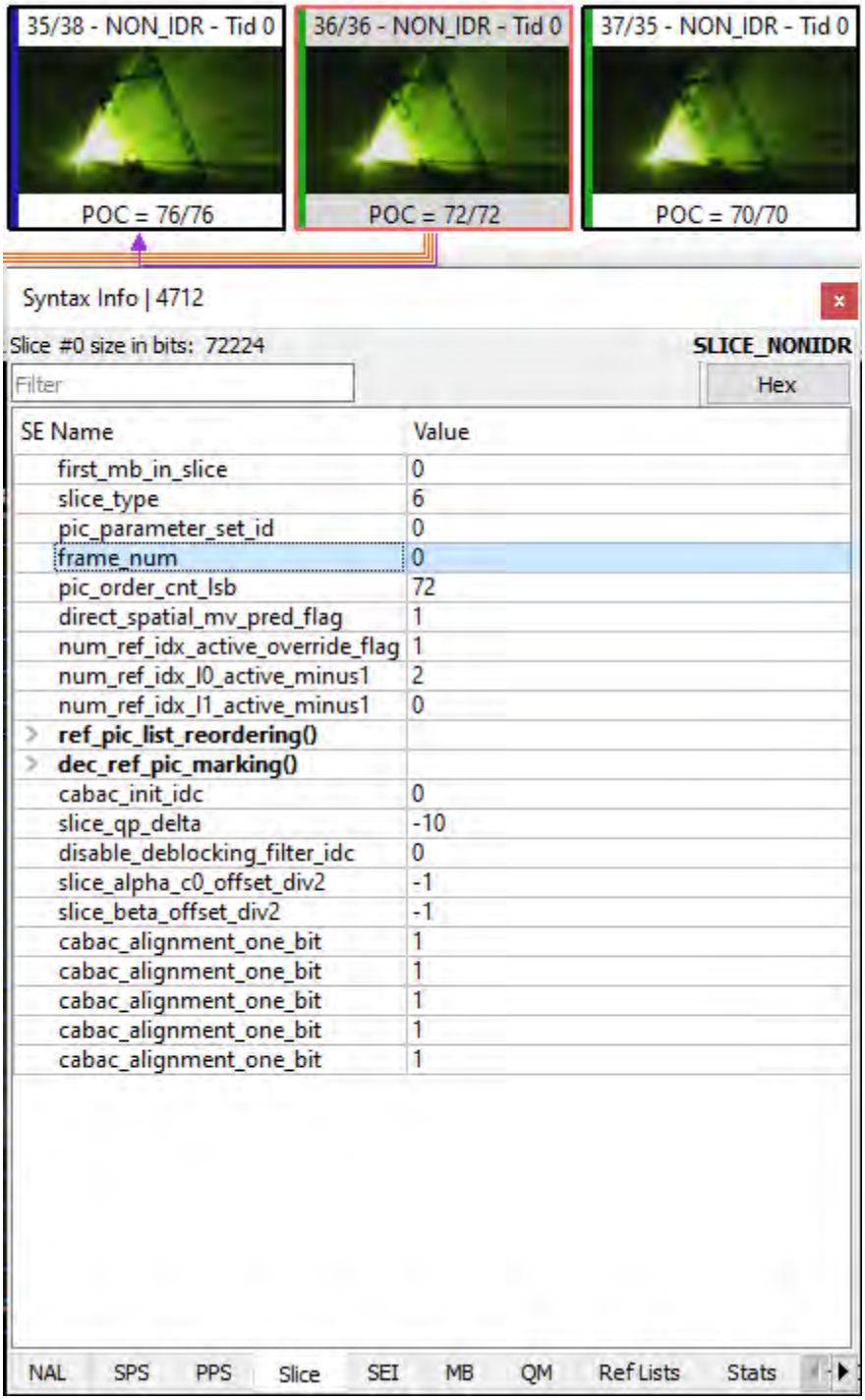
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

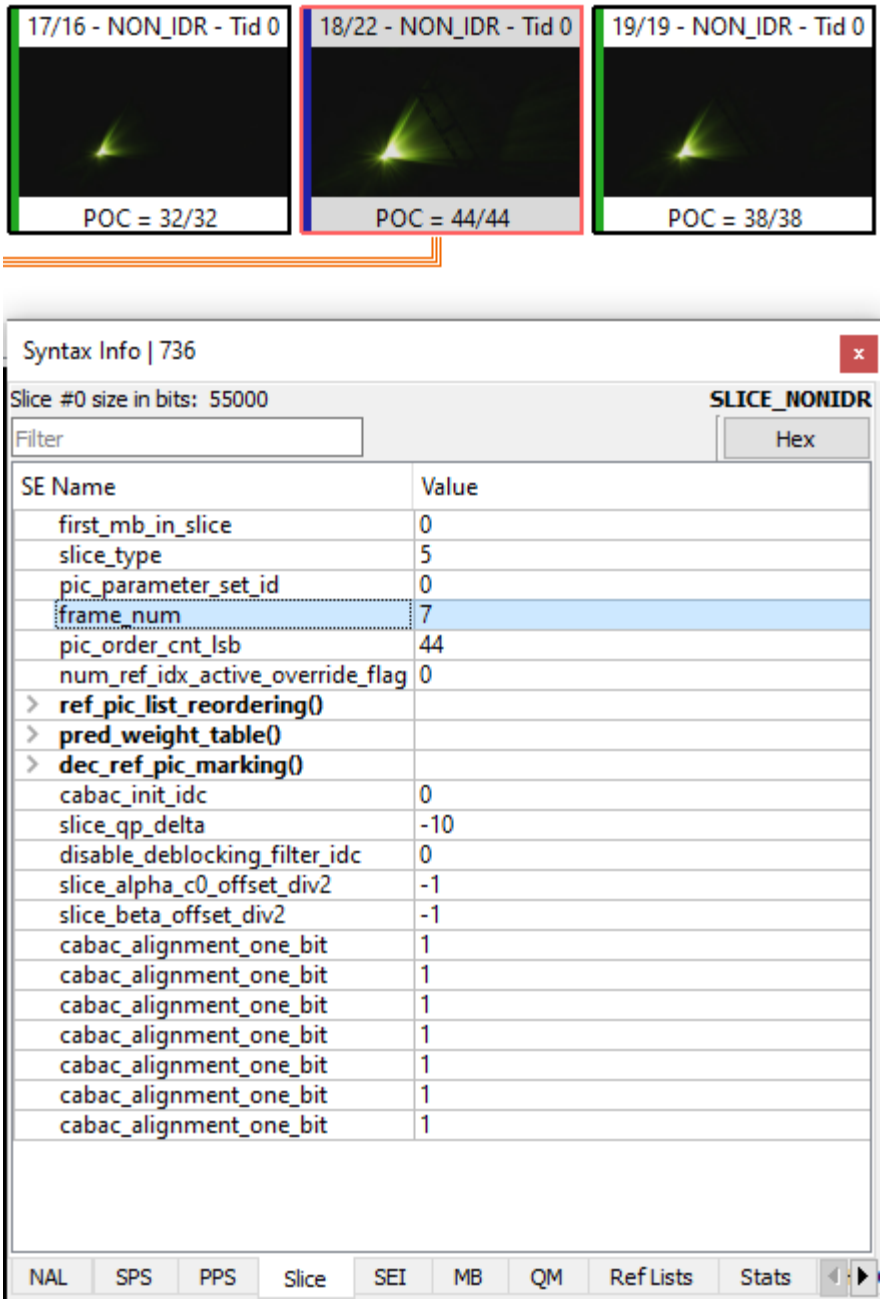


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

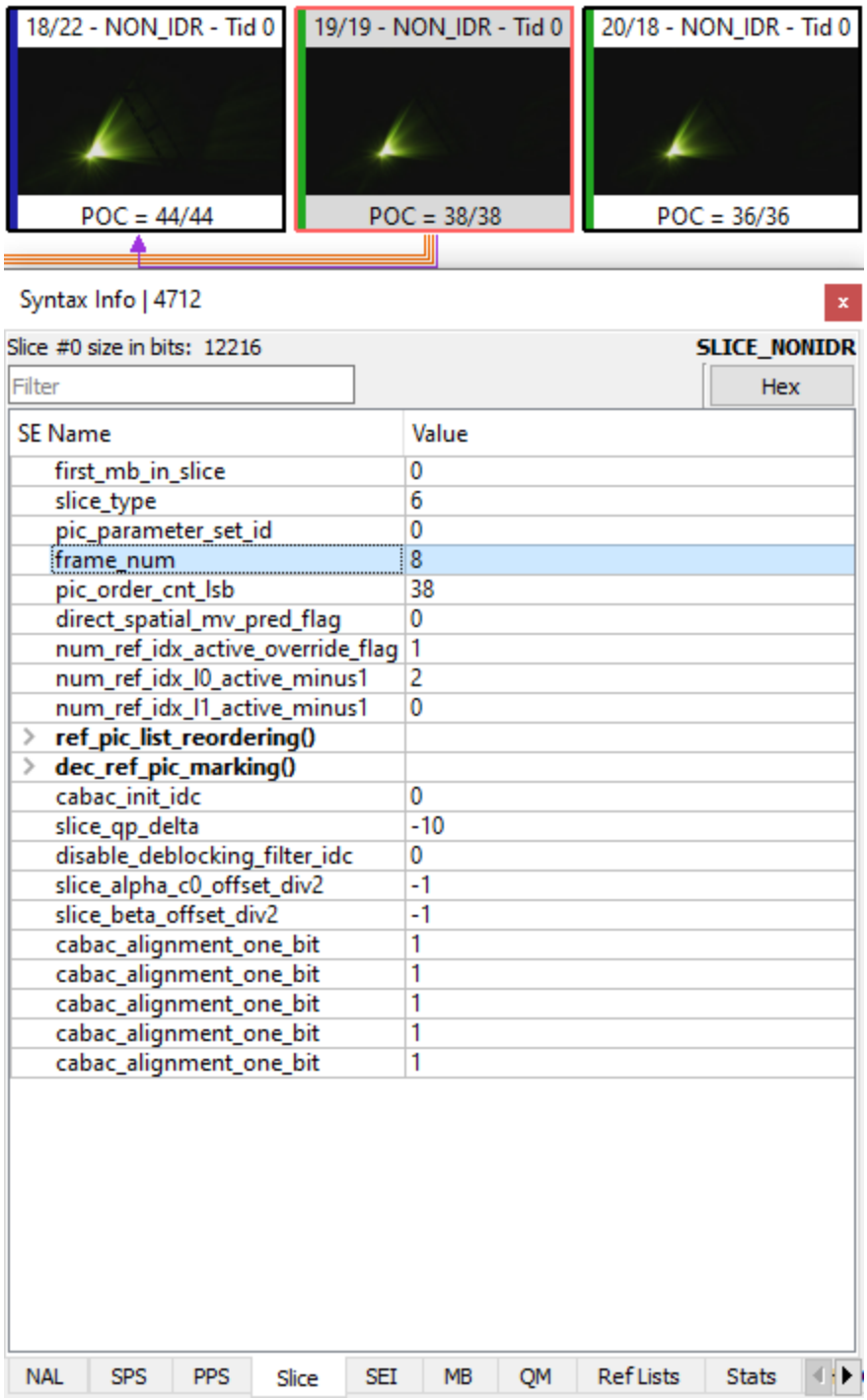
351. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs indicating an encoding order of those pictures used to form reference pictures in the

encoded video signal with a sequence indicator having an independent numbering scheme, such that consecutive pictures used to form reference pictures in encoding order are assigned sequence indicator values that differ with respect to each other by a predetermined amount independent of the number of non-reference pictures encoded between successive reference pictures, as demonstrated in the screenshots below using VQ Analyzer software, where the frame number for reference pictures increases by 1 when there are three non-reference pictures between reference pictures.

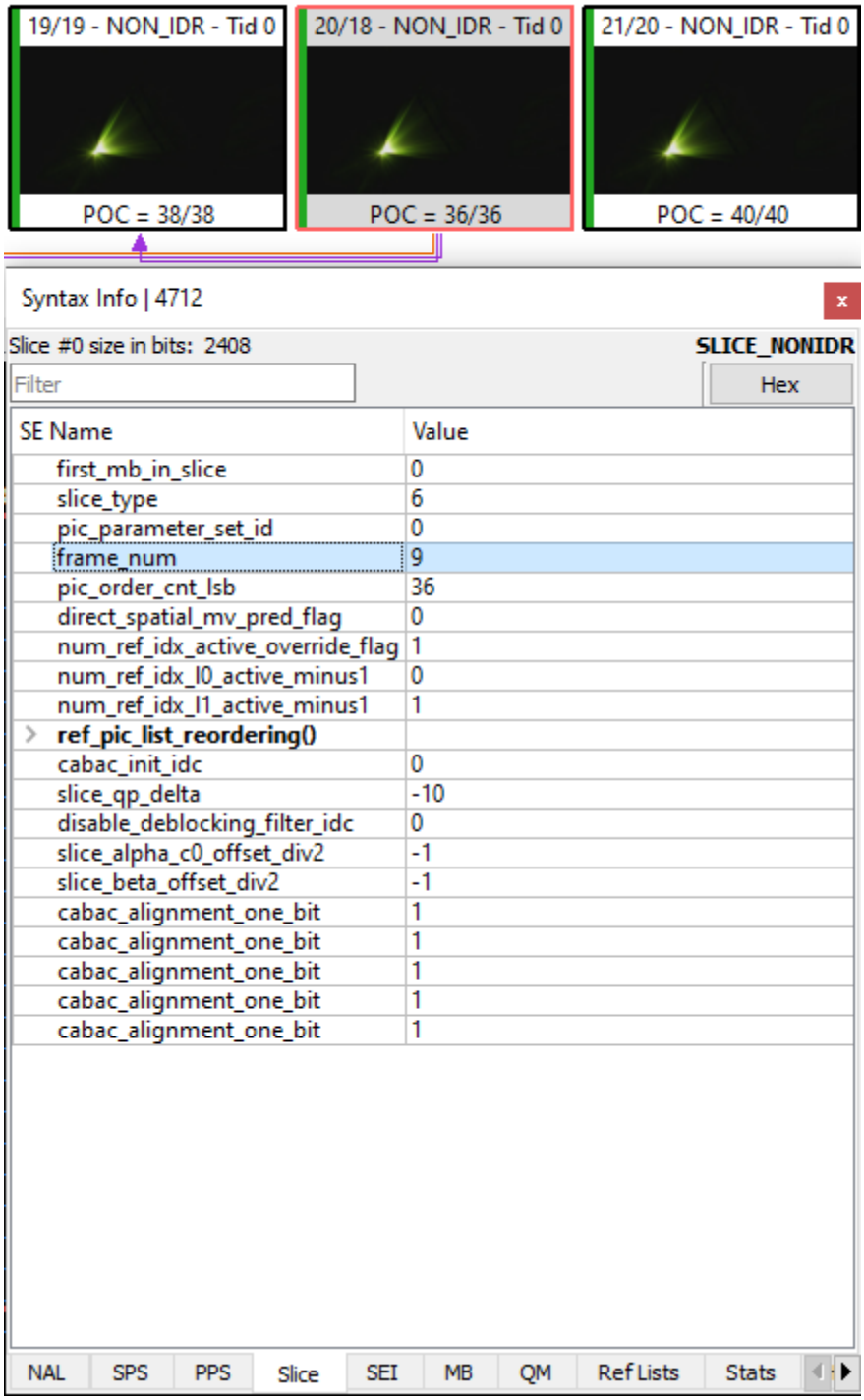
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



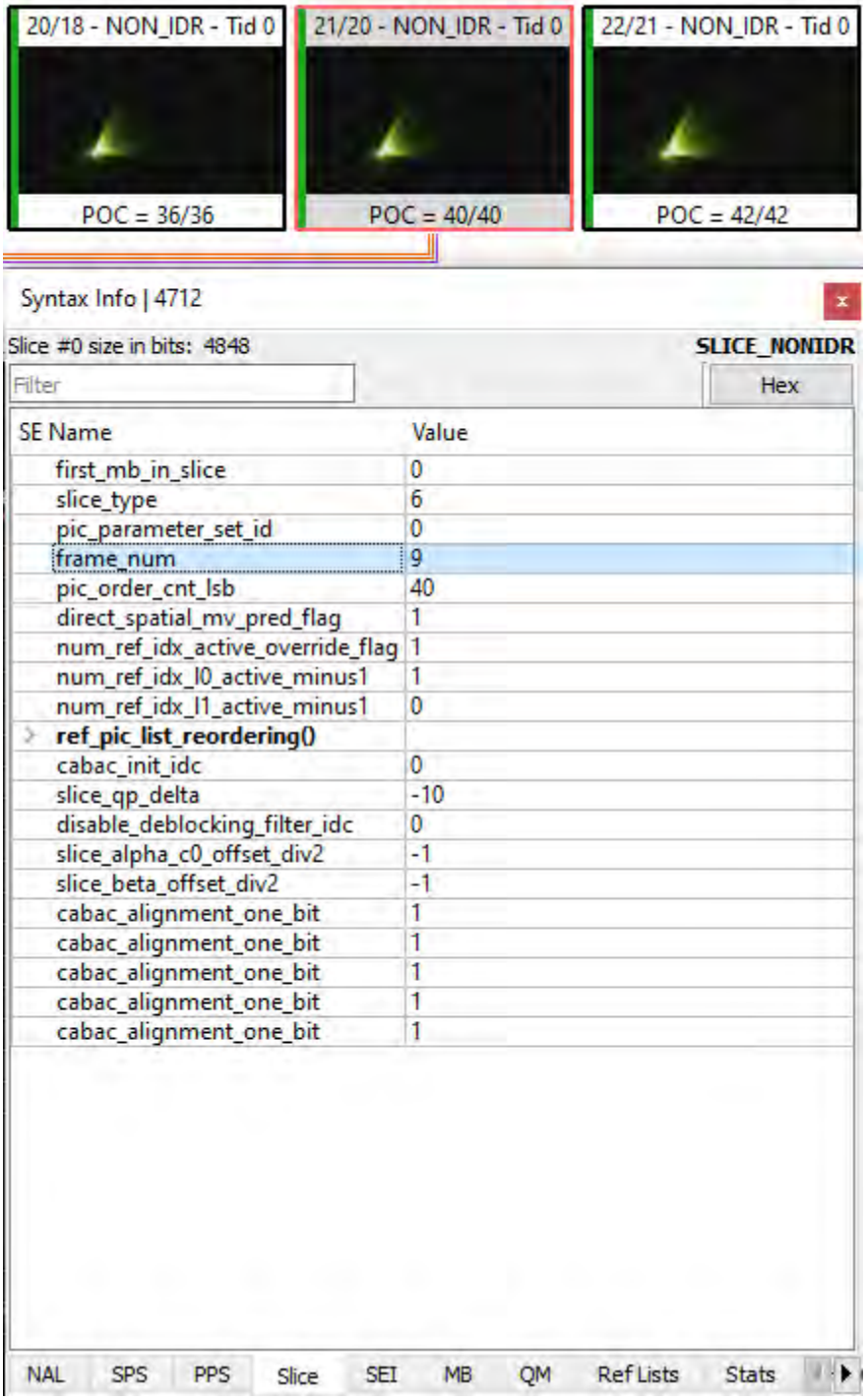
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



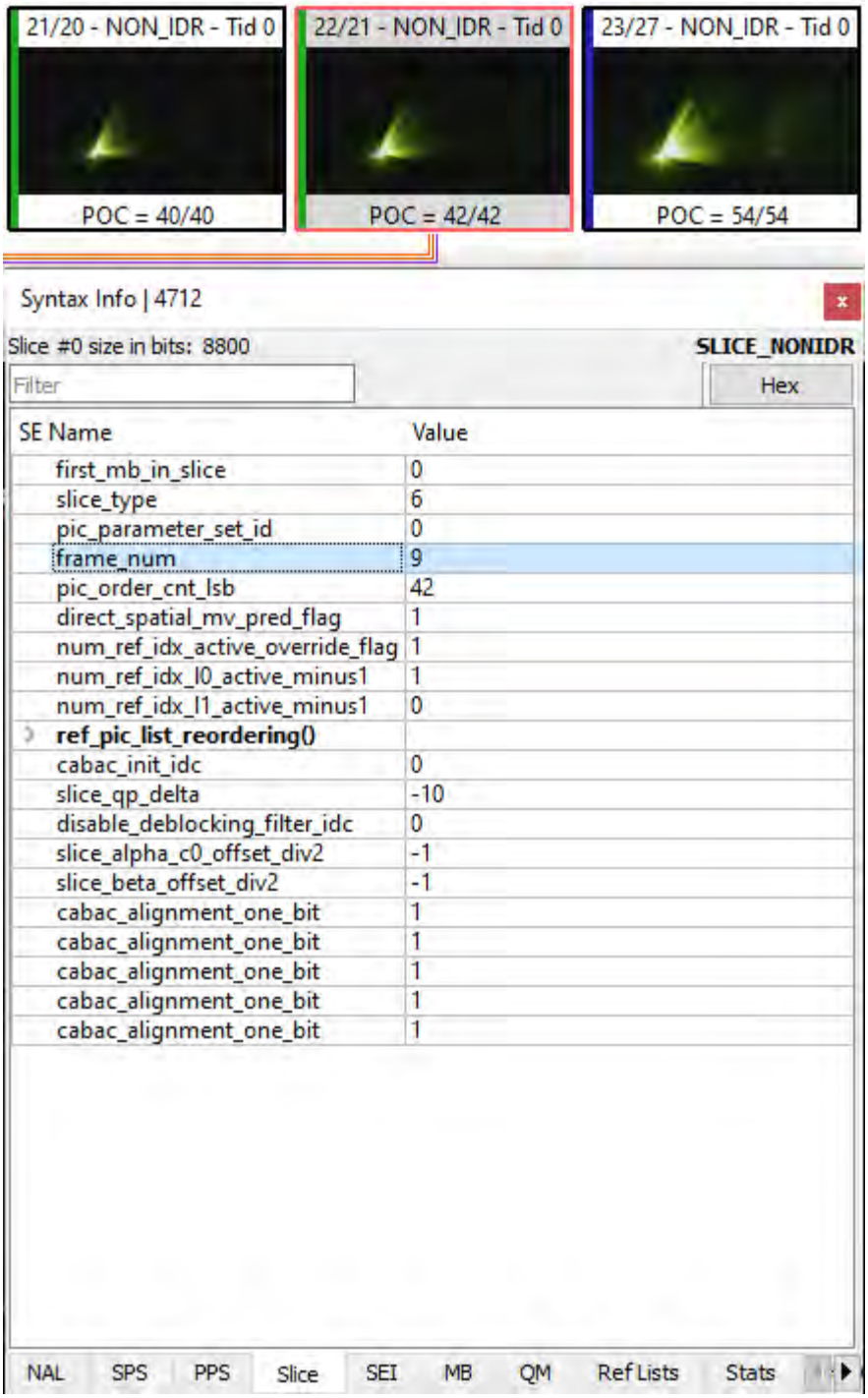
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



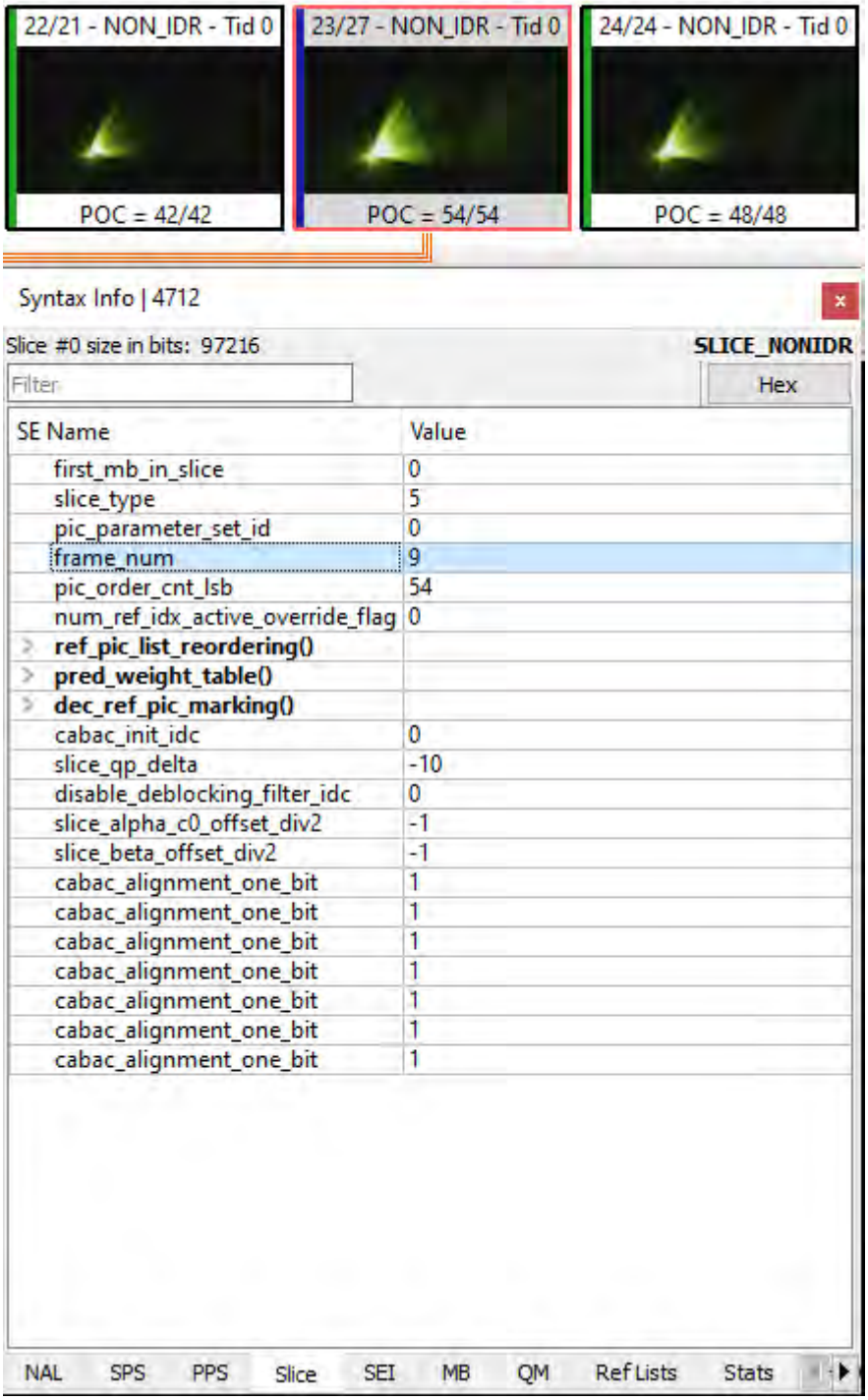
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



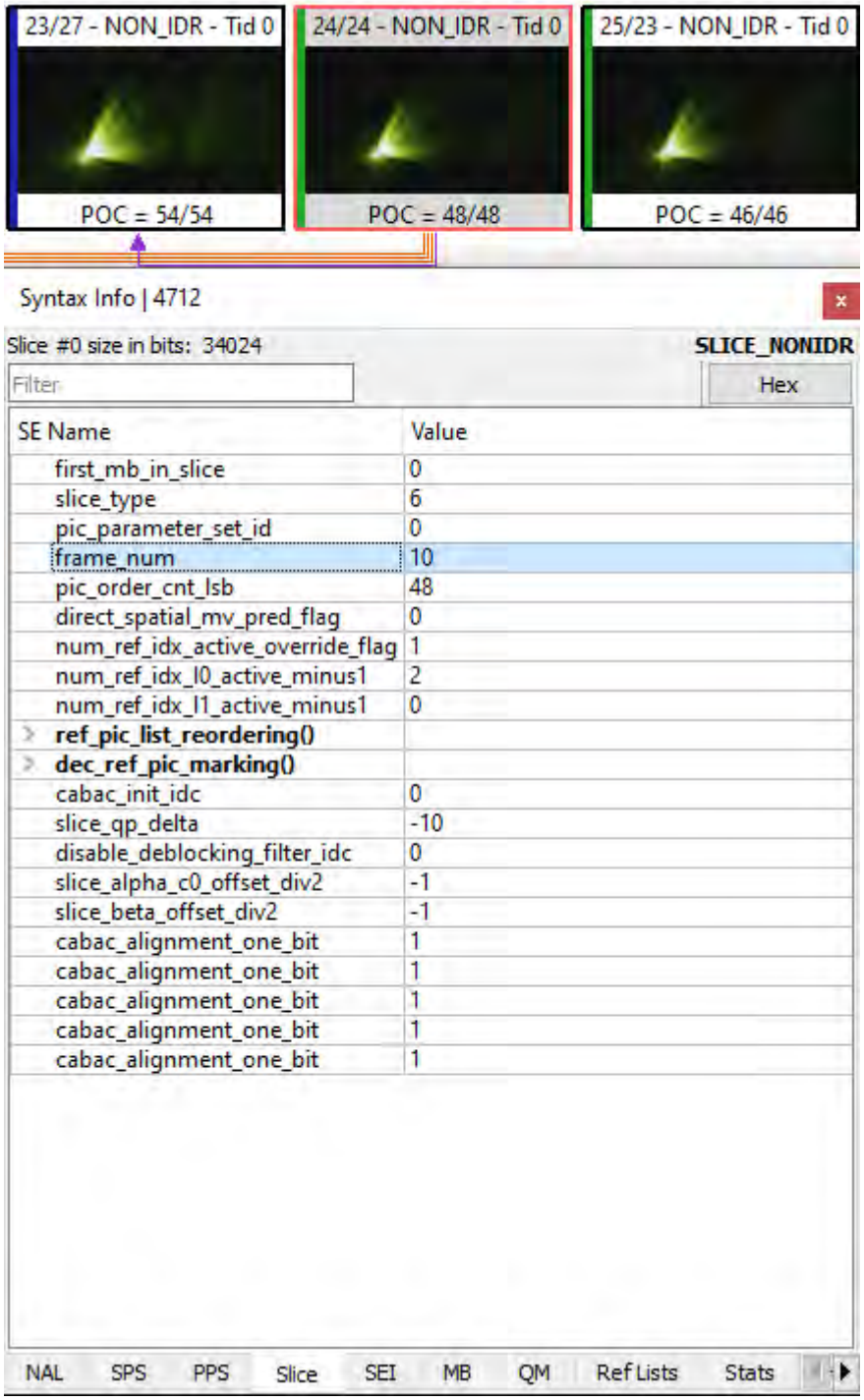
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

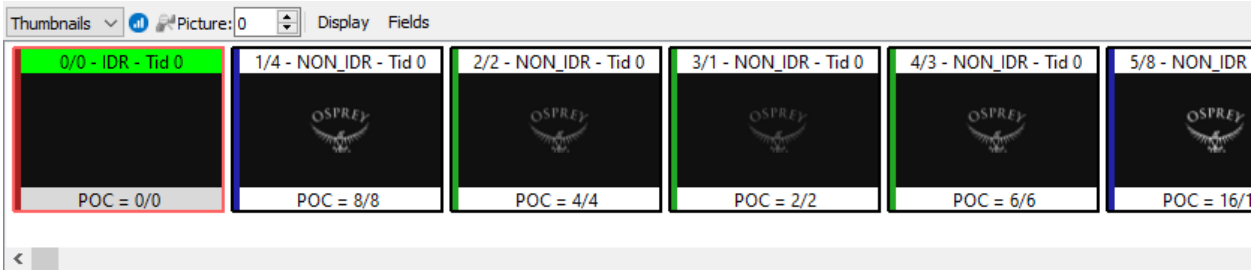


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

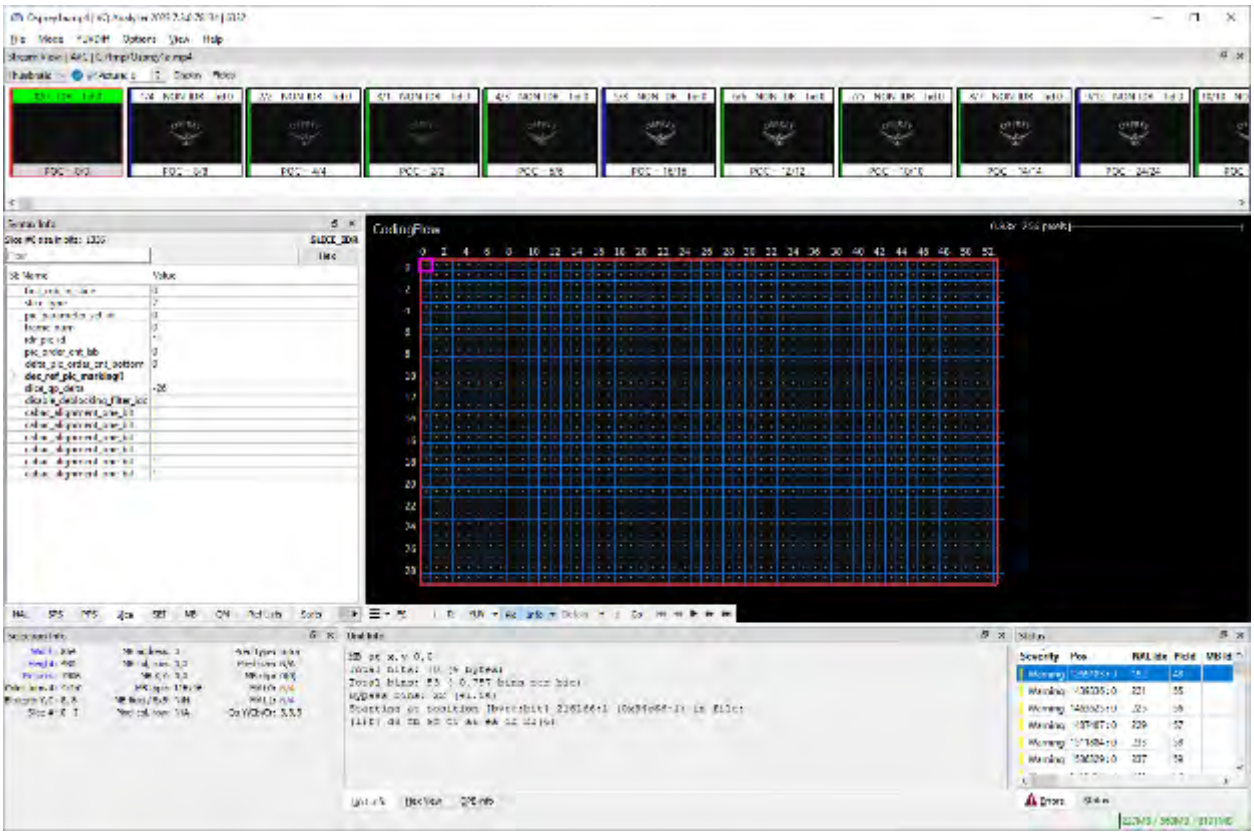
352. For another example, on information and belief, Amazon performs a method of encoding a video signal in a manner that is covered by claim 1 of the ’005 Patent for Amazon.com

advertisements, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 217.

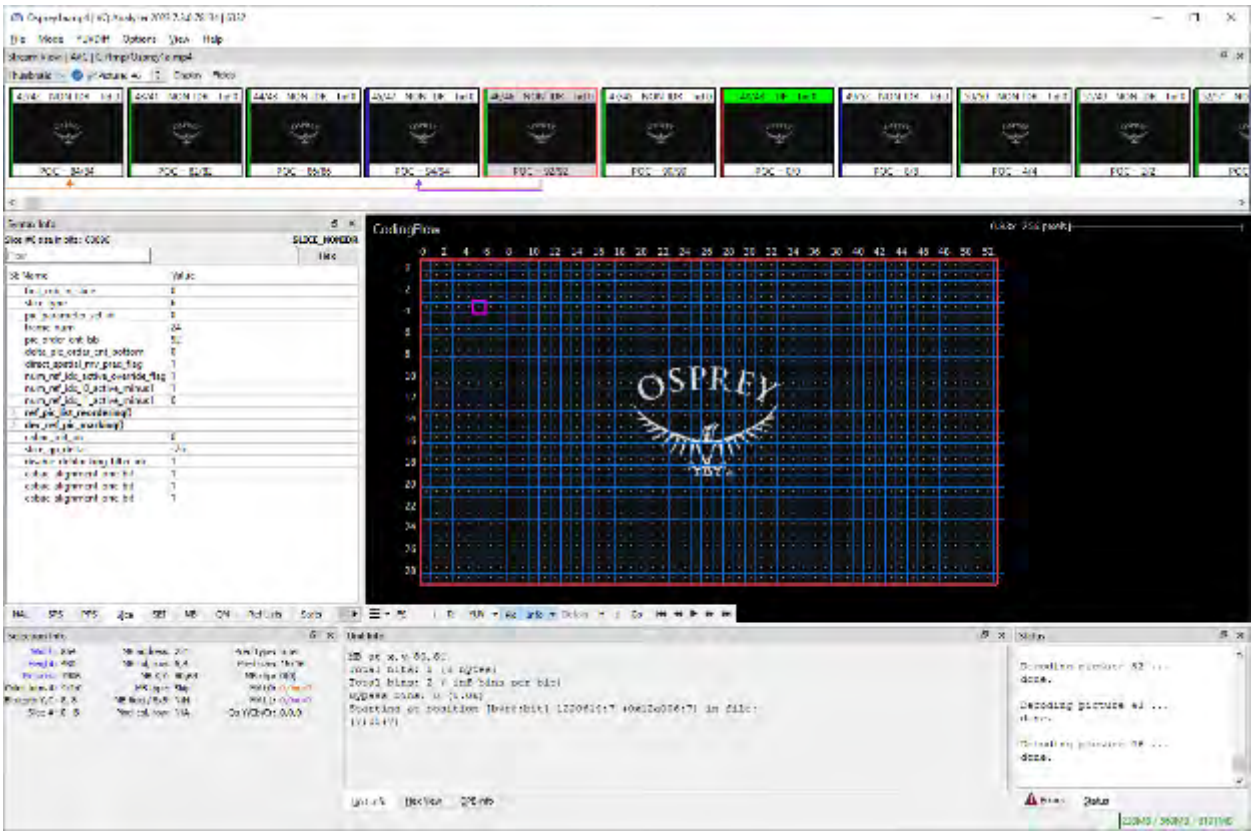
353. When encoding Amazon.com advertisements, on information and belief, Amazon performs encoding a video signal representing a sequence of pictures to form an encoded video signal comprising temporally independent INTRA pictures and temporally predicted pictures, wherein the INTRA pictures and at least some of the temporally predicted pictures are used to form reference pictures for the temporal prediction of other pictures in the video sequence, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

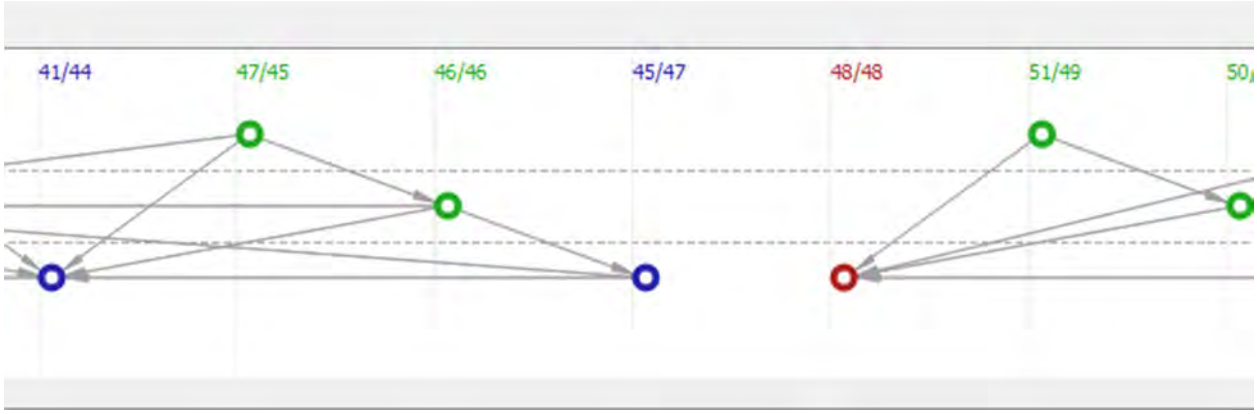


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

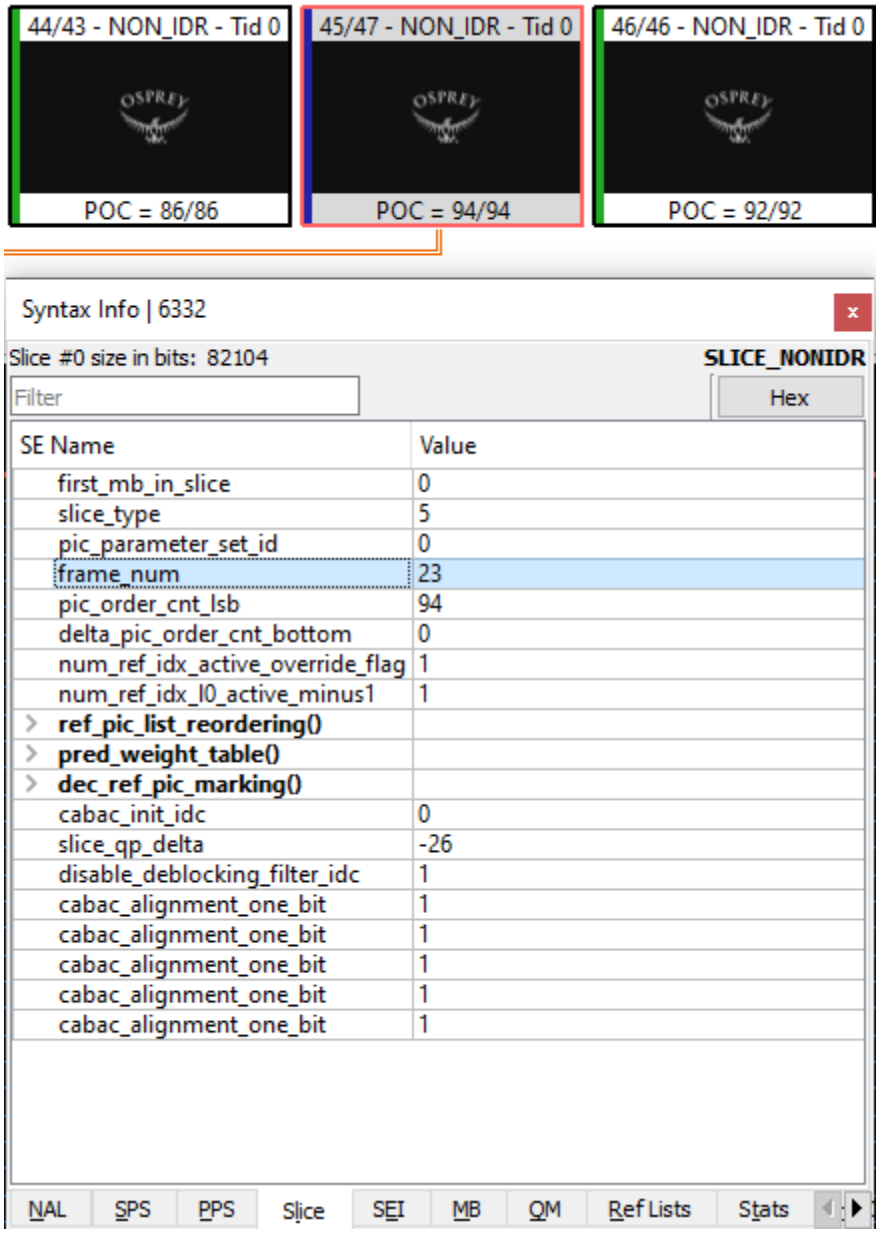


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

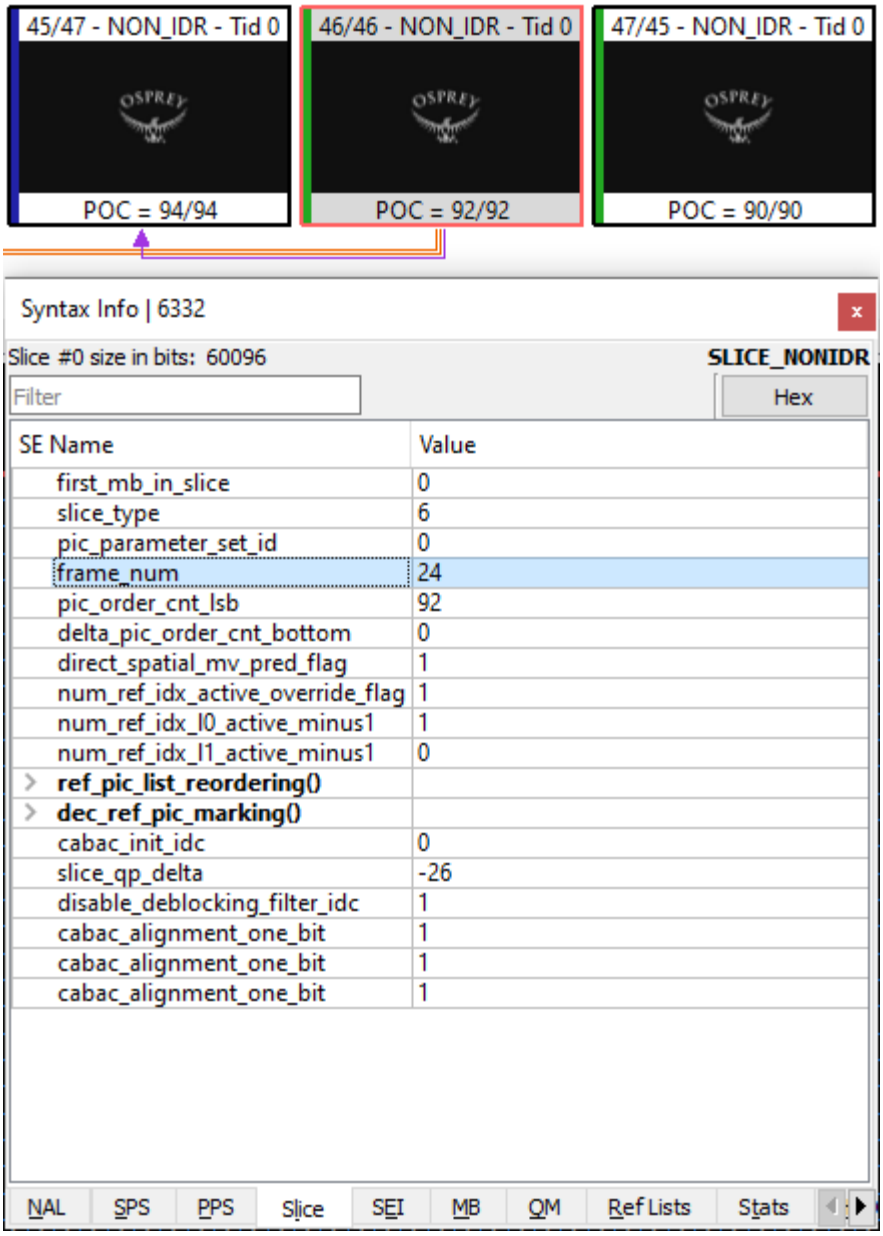
354. When encoding Amazon.com advertisements, on information and belief, Amazon performs indicating an encoding order of those pictures used to form reference pictures in the encoded video signal with a sequence indicator having an independent numbering scheme, such that consecutive pictures used to form reference pictures in encoding order are assigned sequence indicator values that differ with respect to each other by a predetermined amount independent of the number of non-reference pictures encoded between successive reference pictures, as demonstrated in the screenshots below using VQ Analyzer software, where the frame number for reference pictures increases by 1 when there is one non-reference picture between reference pictures.



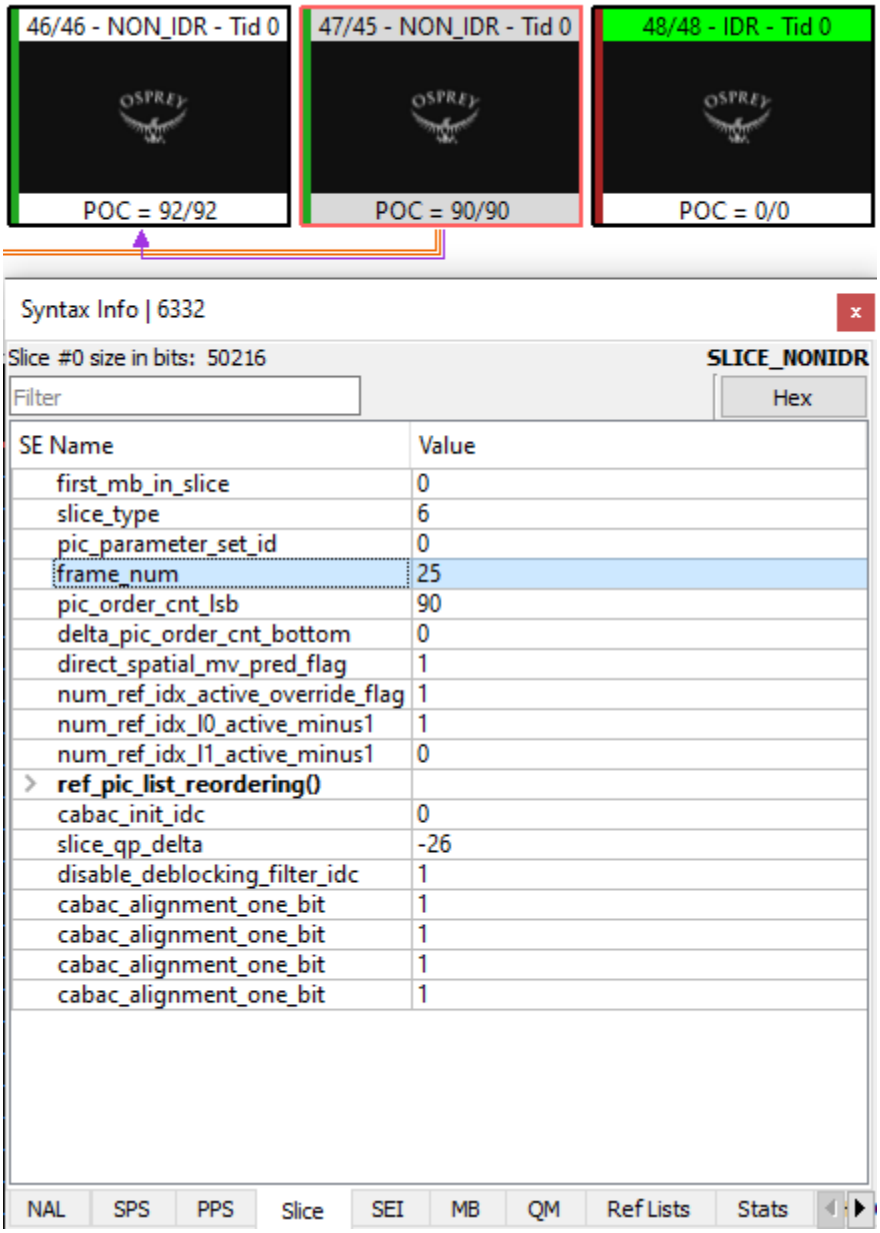
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



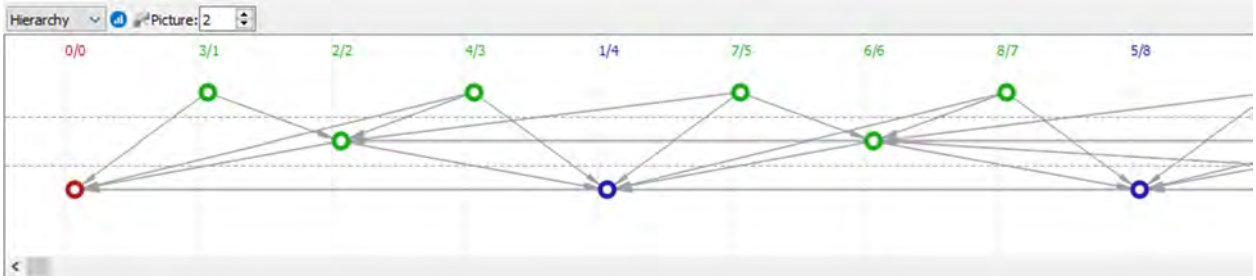
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



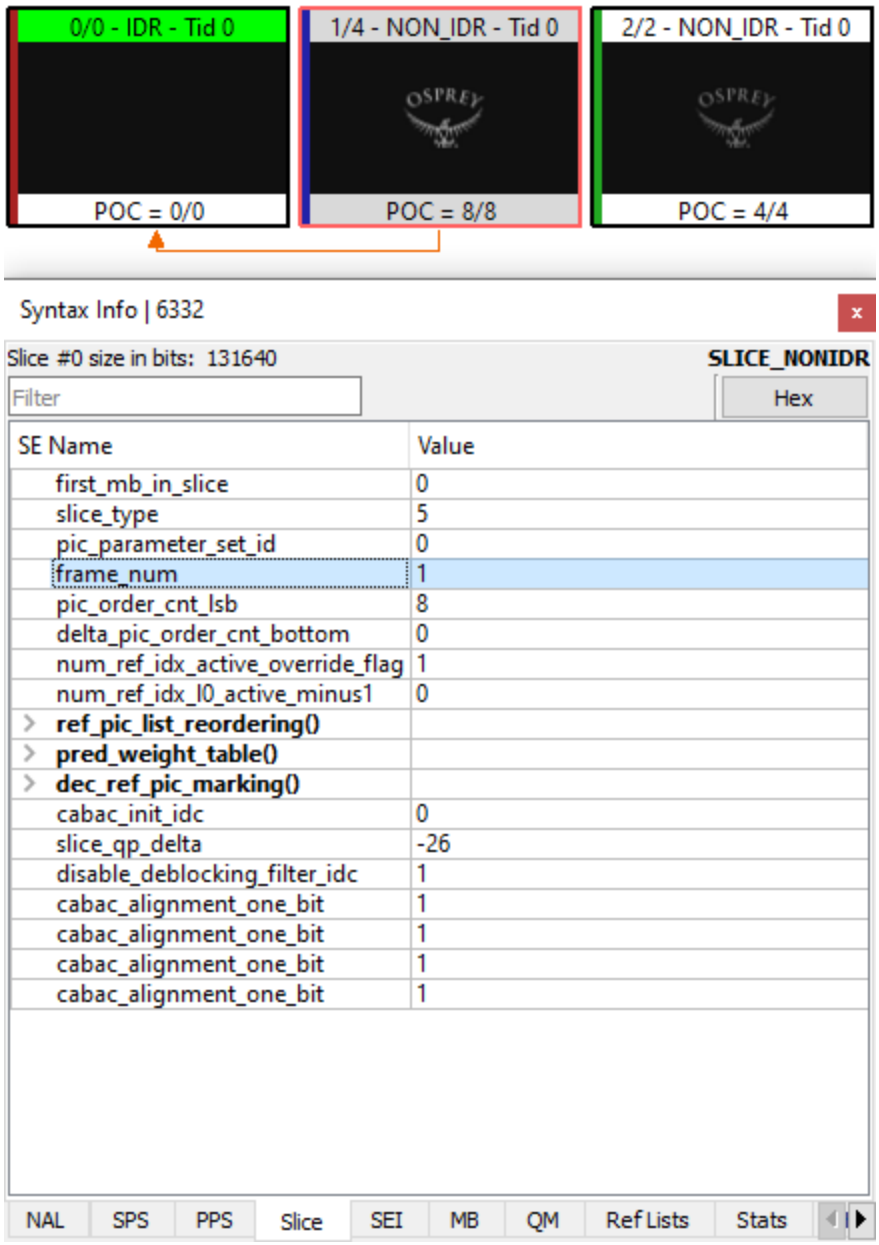
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

355. When encoding Amazon.com advertisements, on information and belief, Amazon performs indicating an encoding order of those pictures used to form reference pictures in the encoded video signal with a sequence indicator having an independent numbering scheme, such that consecutive pictures used to form reference pictures in encoding order are assigned sequence indicator values that differ with respect to each other by a predetermined amount independent of

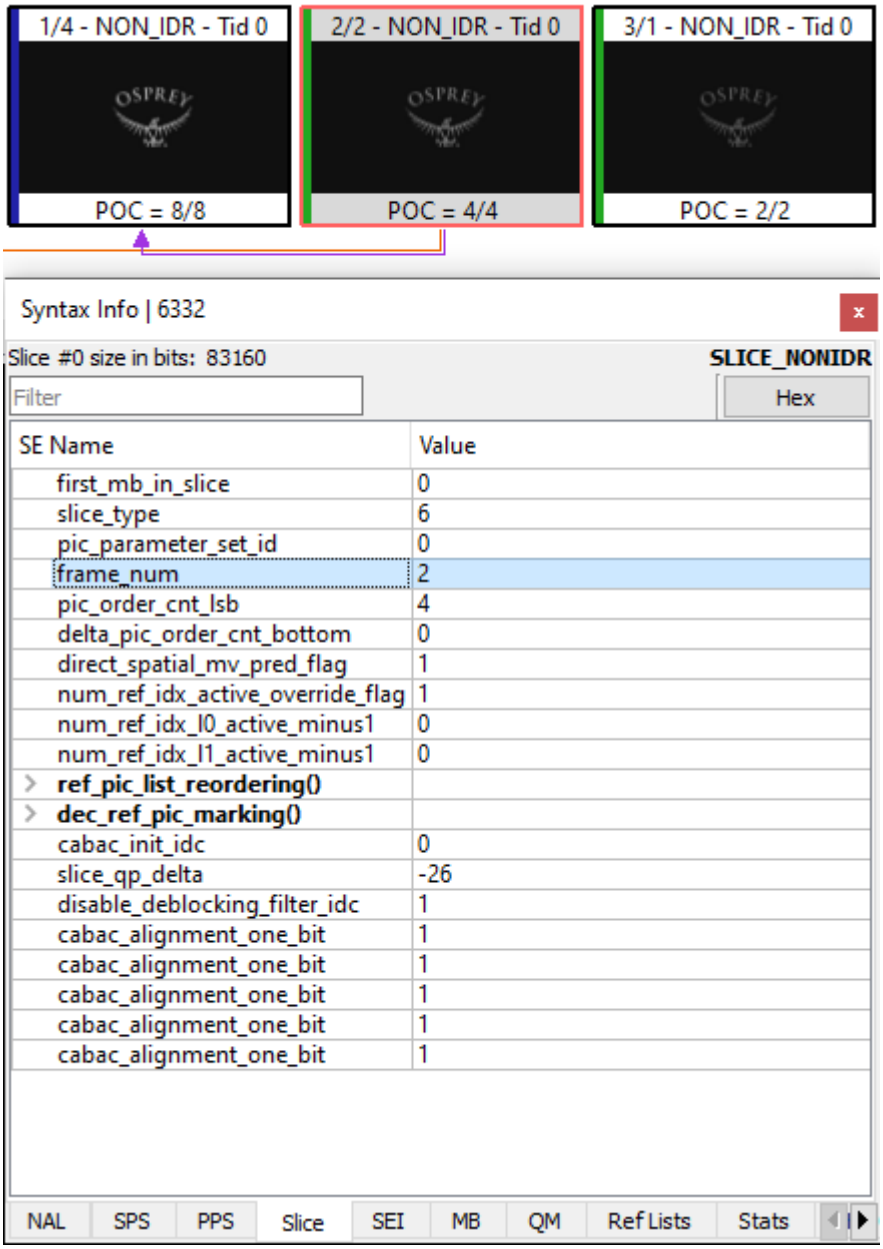
the number of non-reference pictures encoded between successive reference pictures, as demonstrated in the screenshots below using VQ Analyzer software, where the frame number for reference pictures increases by 1 when there are two non-reference pictures between reference pictures.



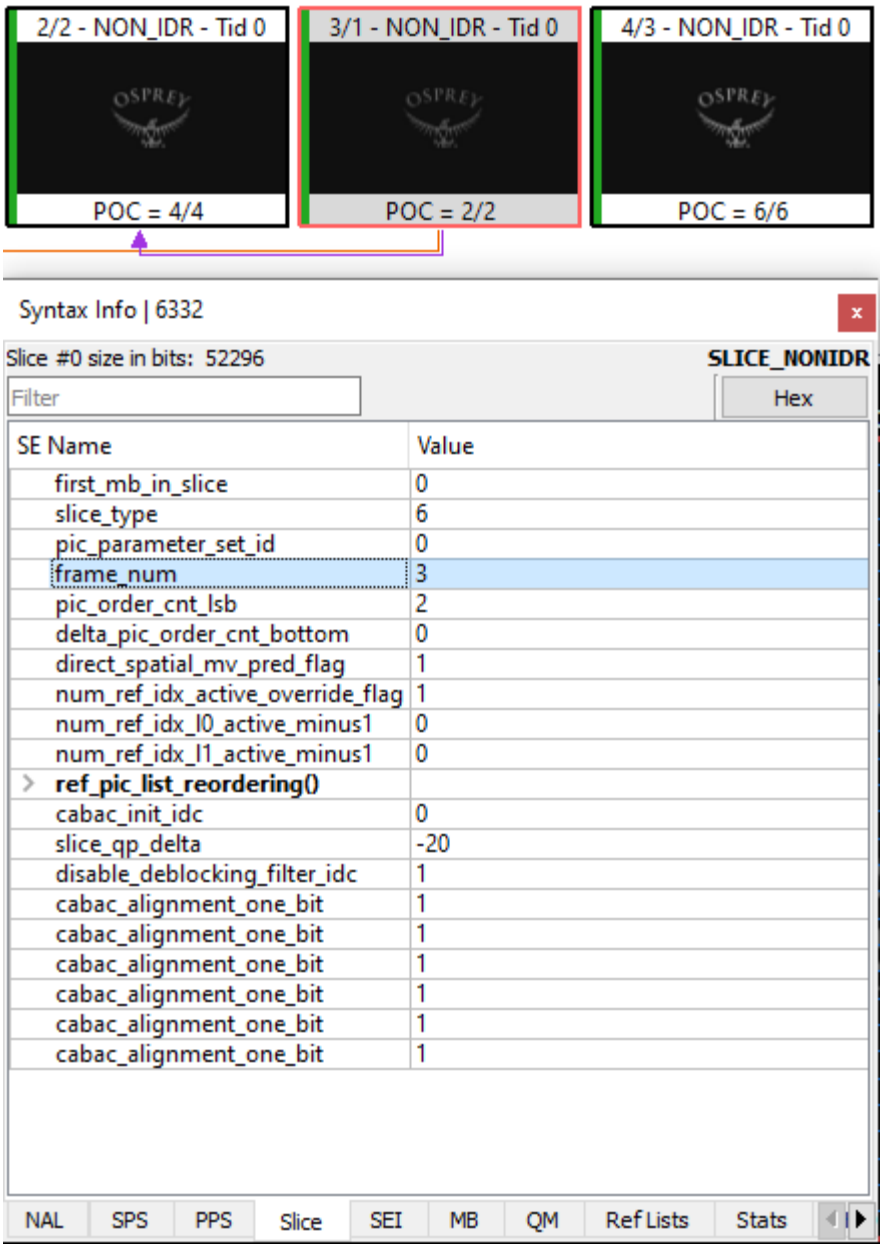
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



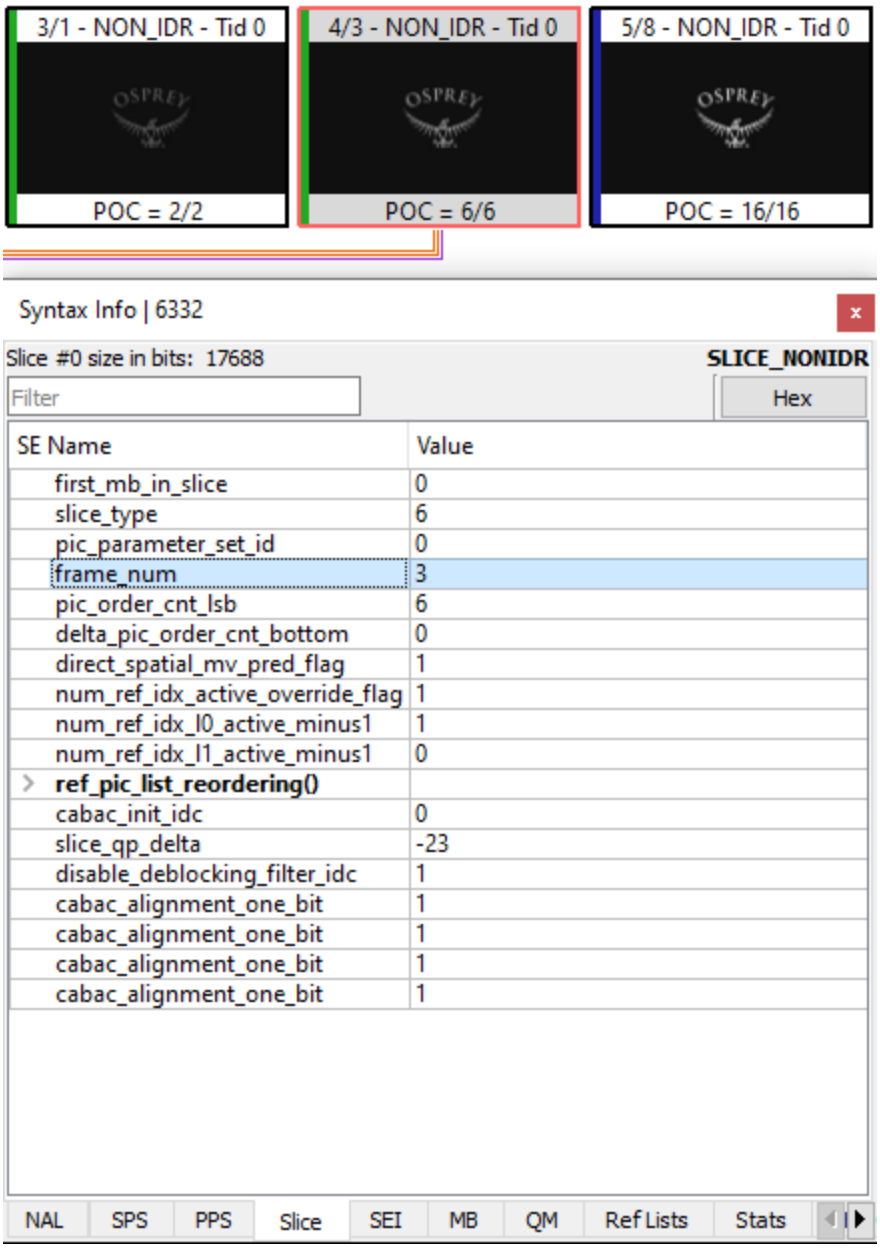
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



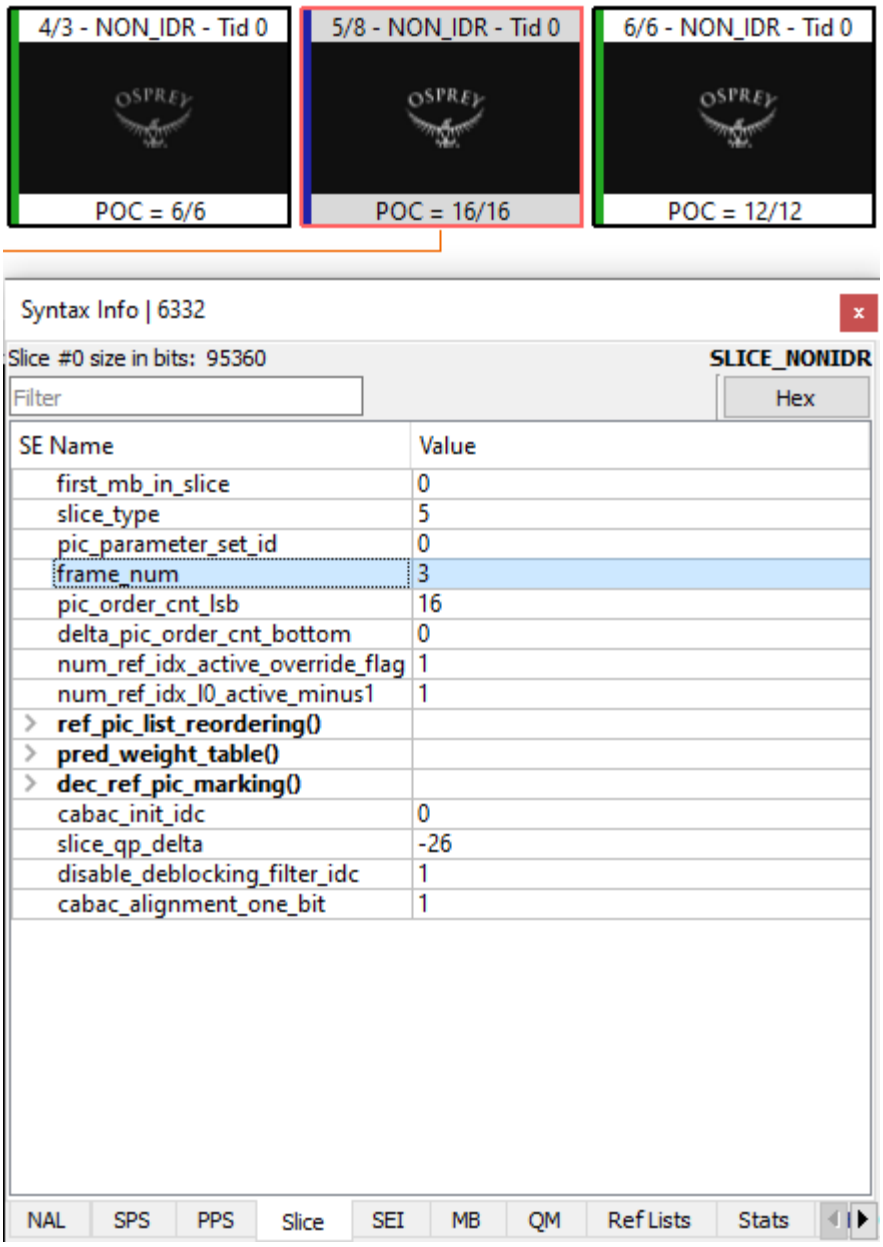
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



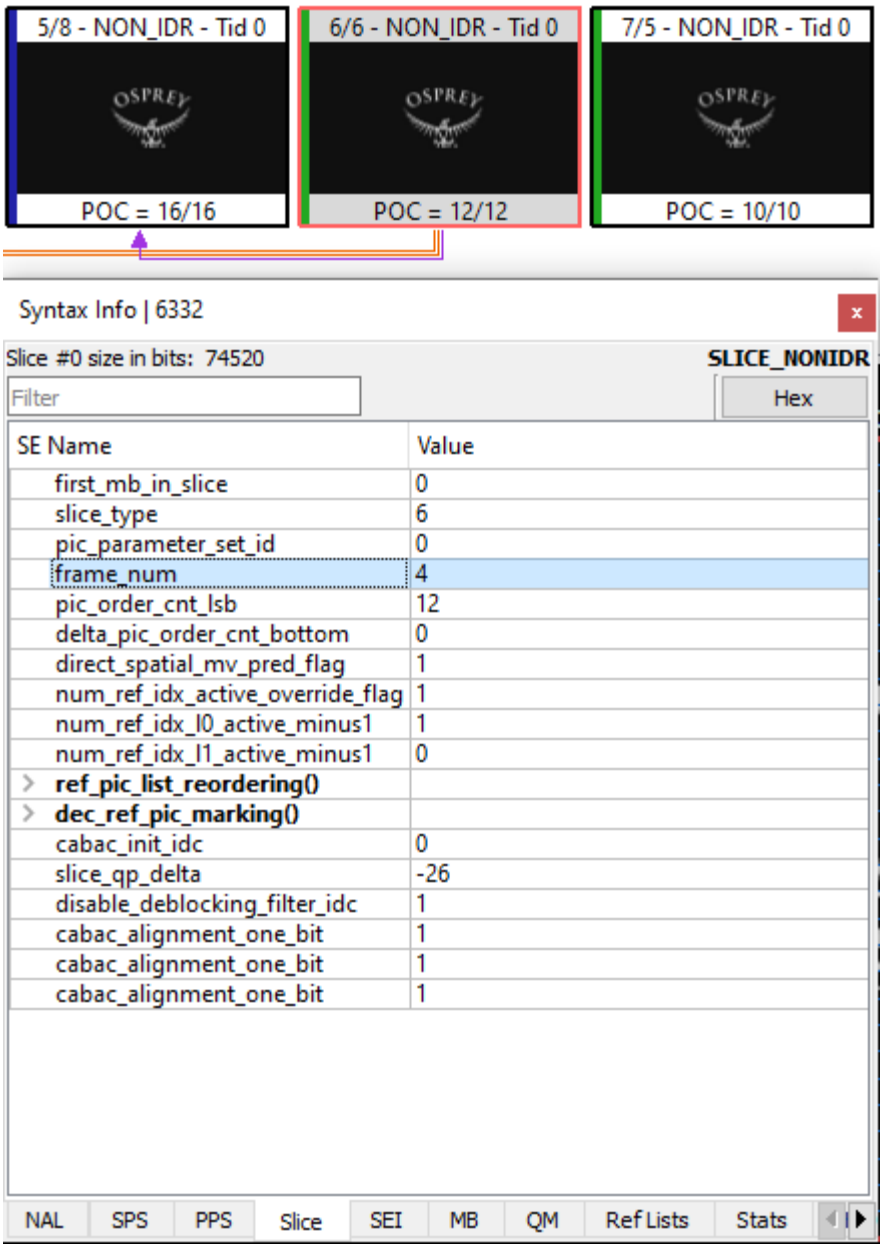
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



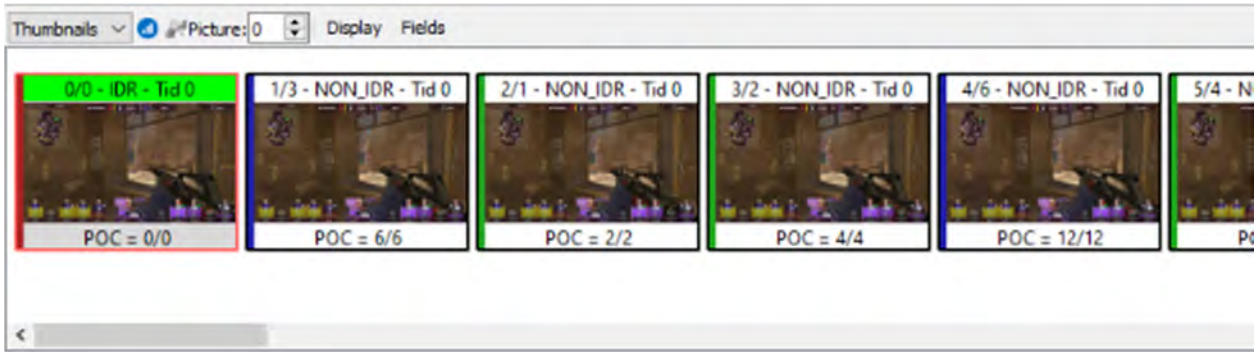
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



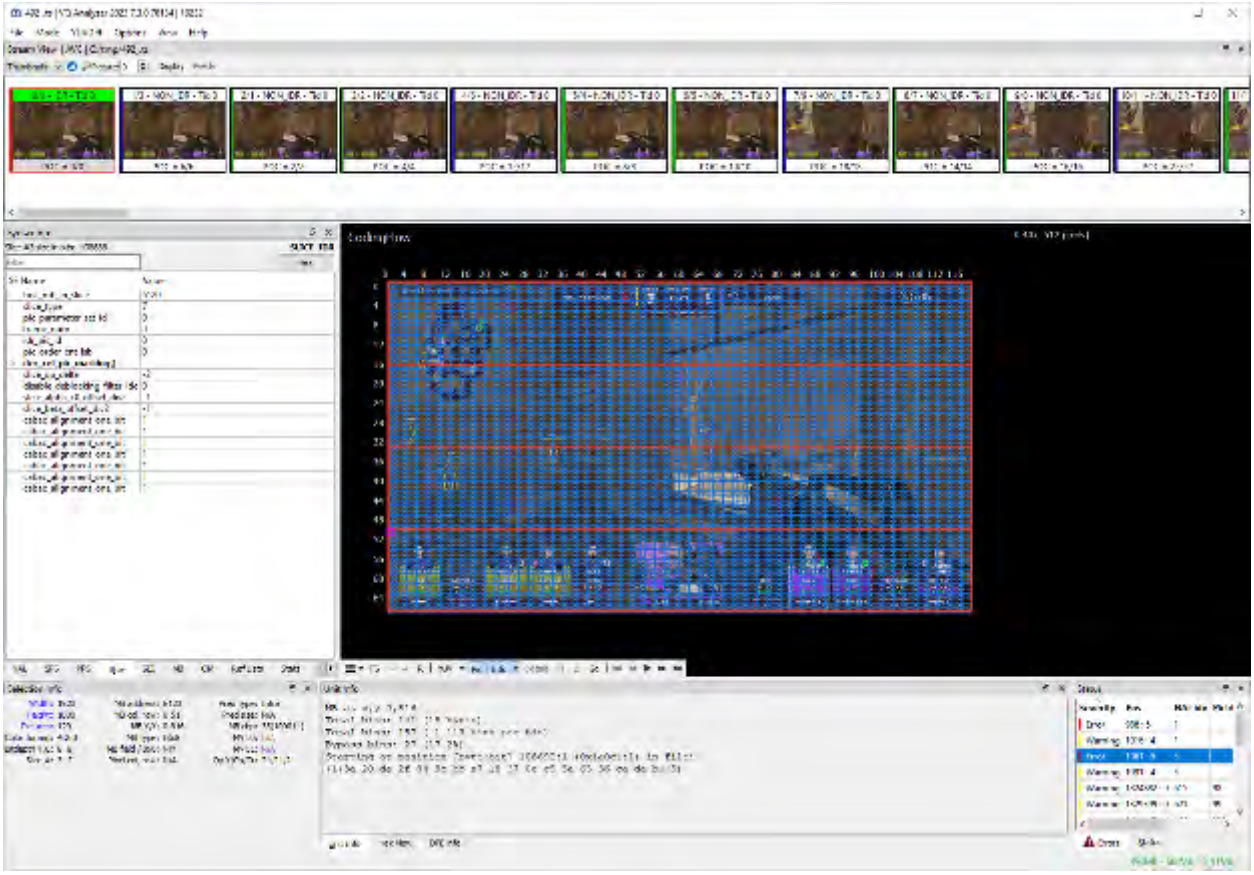
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

356. For another example, on information and belief, Amazon performs a method of encoding a video signal in a manner that is covered by claim 1 of the ’005 Patent for Twitch.tv, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 222.

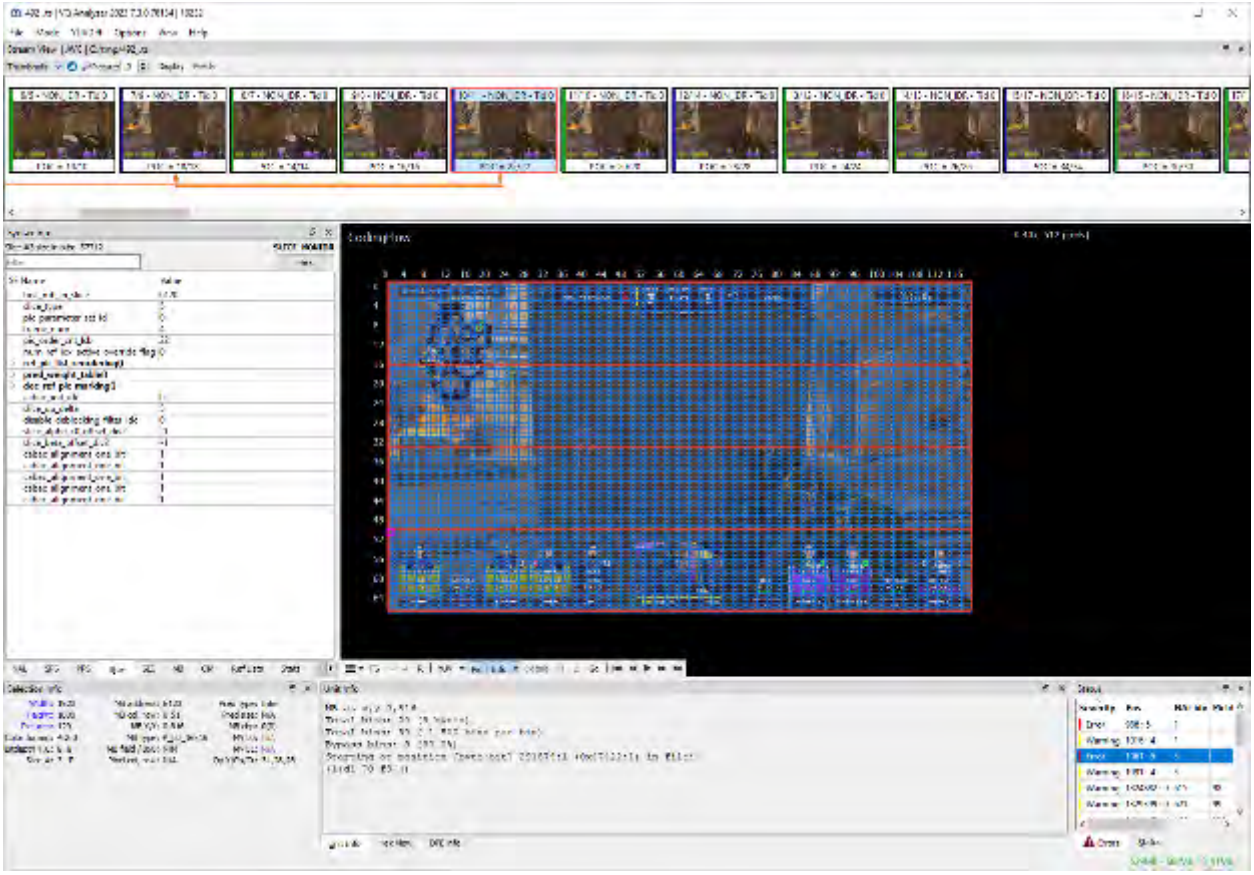
357. When encoding Twitch.tv video, on information and belief, Amazon performs encoding a video signal representing a sequence of pictures to form an encoded video signal comprising temporally independent INTRA pictures and temporally predicted pictures, wherein the INTRA pictures and at least some of the temporally predicted pictures are used to form reference pictures for the temporal prediction of other pictures in the video sequence, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

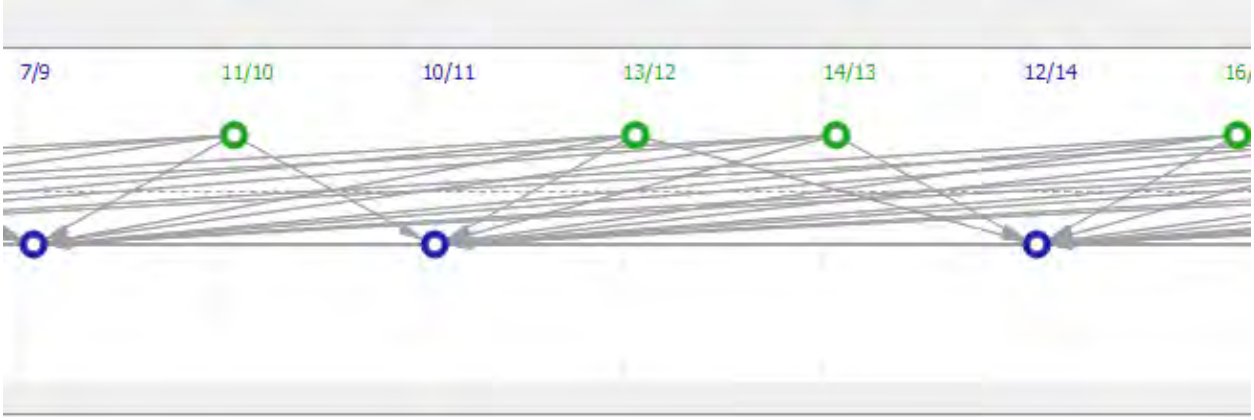


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

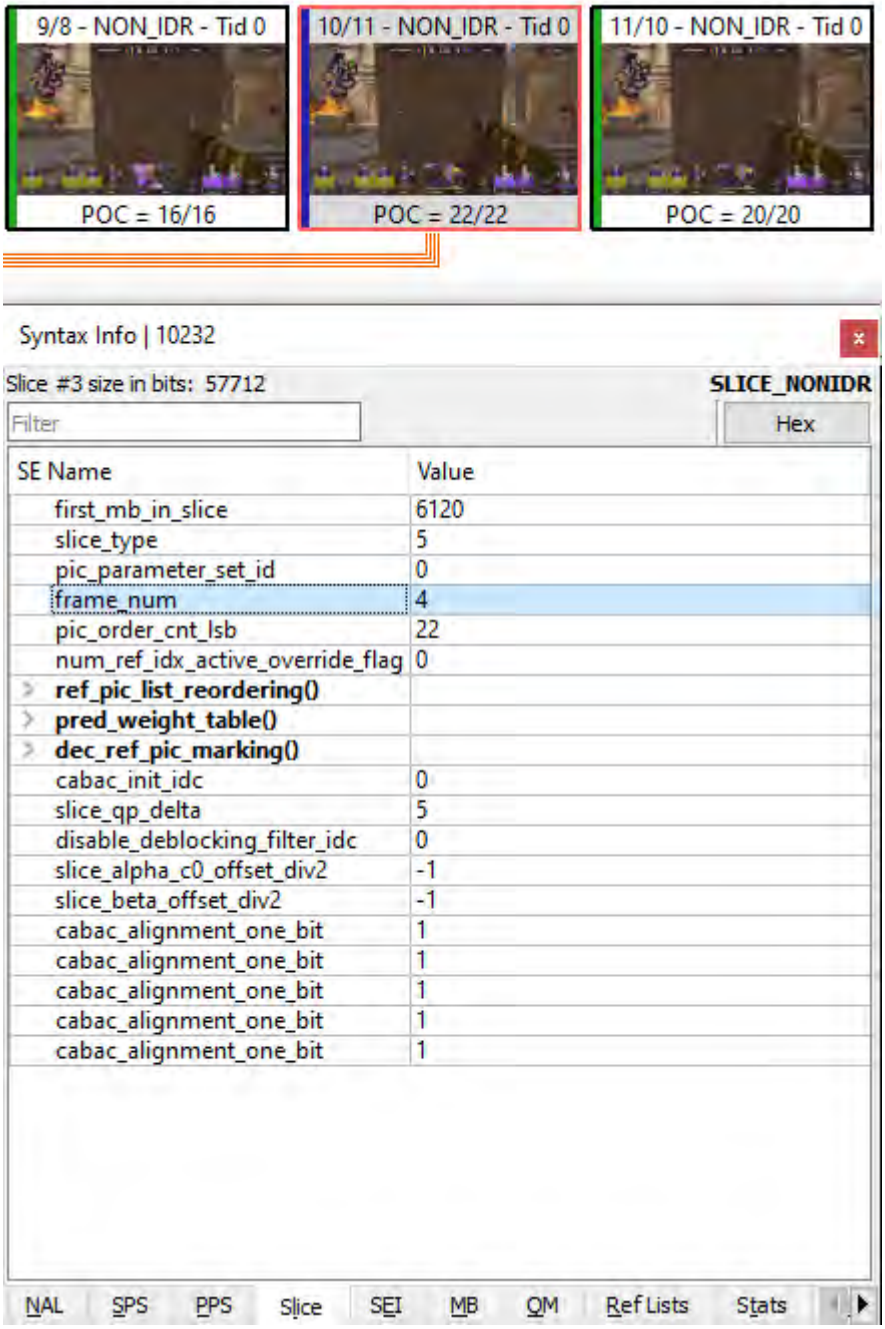


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

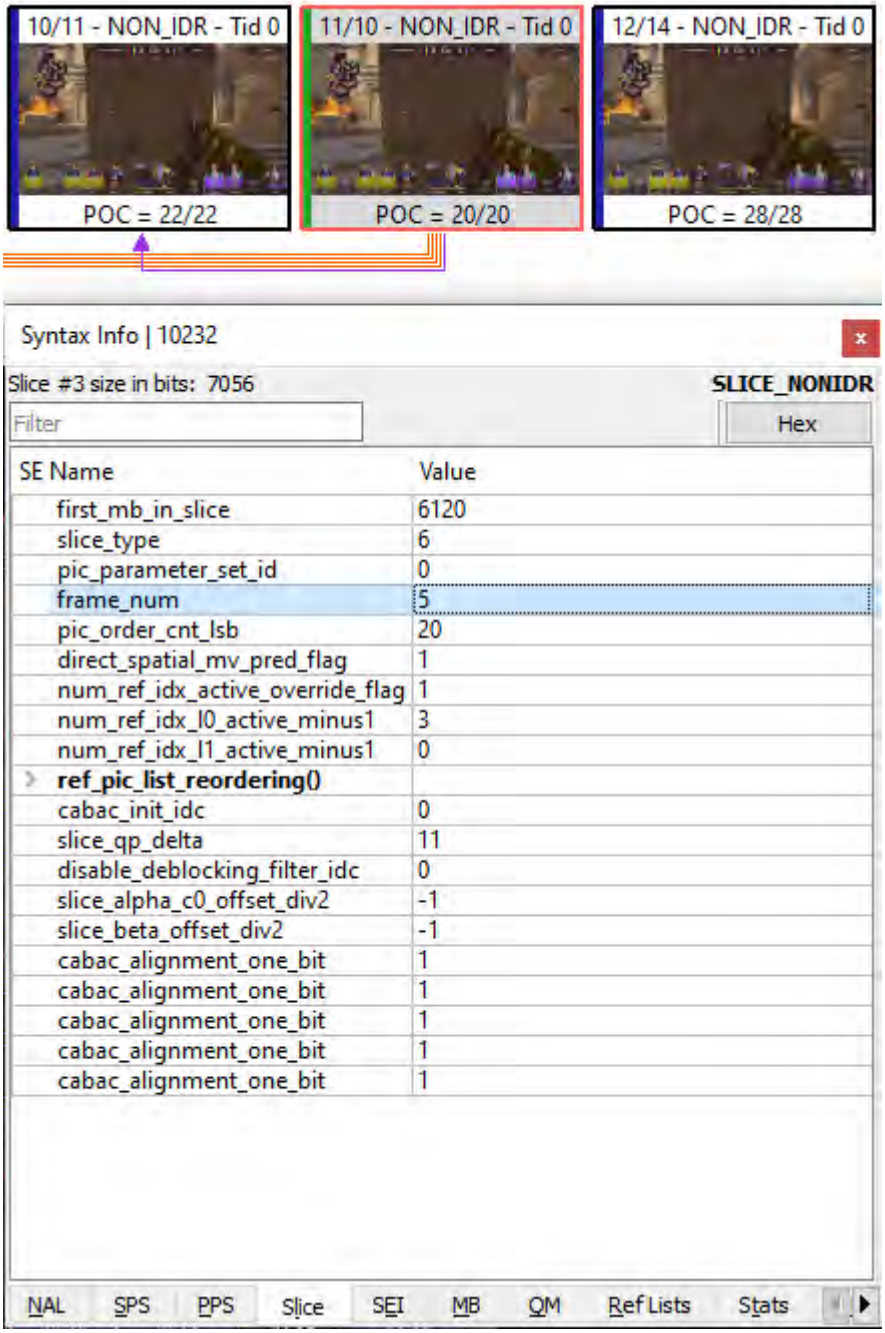
358. When encoding Twitch.tv video, on information and belief, Amazon performs indicating an encoding order of those pictures used to form reference pictures in the encoded video signal with a sequence indicator having an independent numbering scheme, such that consecutive pictures used to form reference pictures in encoding order are assigned sequence indicator values that differ with respect to each other by a predetermined amount independent of the number of non-reference pictures encoded between successive reference pictures, as demonstrated in the screenshots below using VQ Analyzer software, where the frame number for reference pictures increases by 1 when there is one non-reference picture between reference pictures.



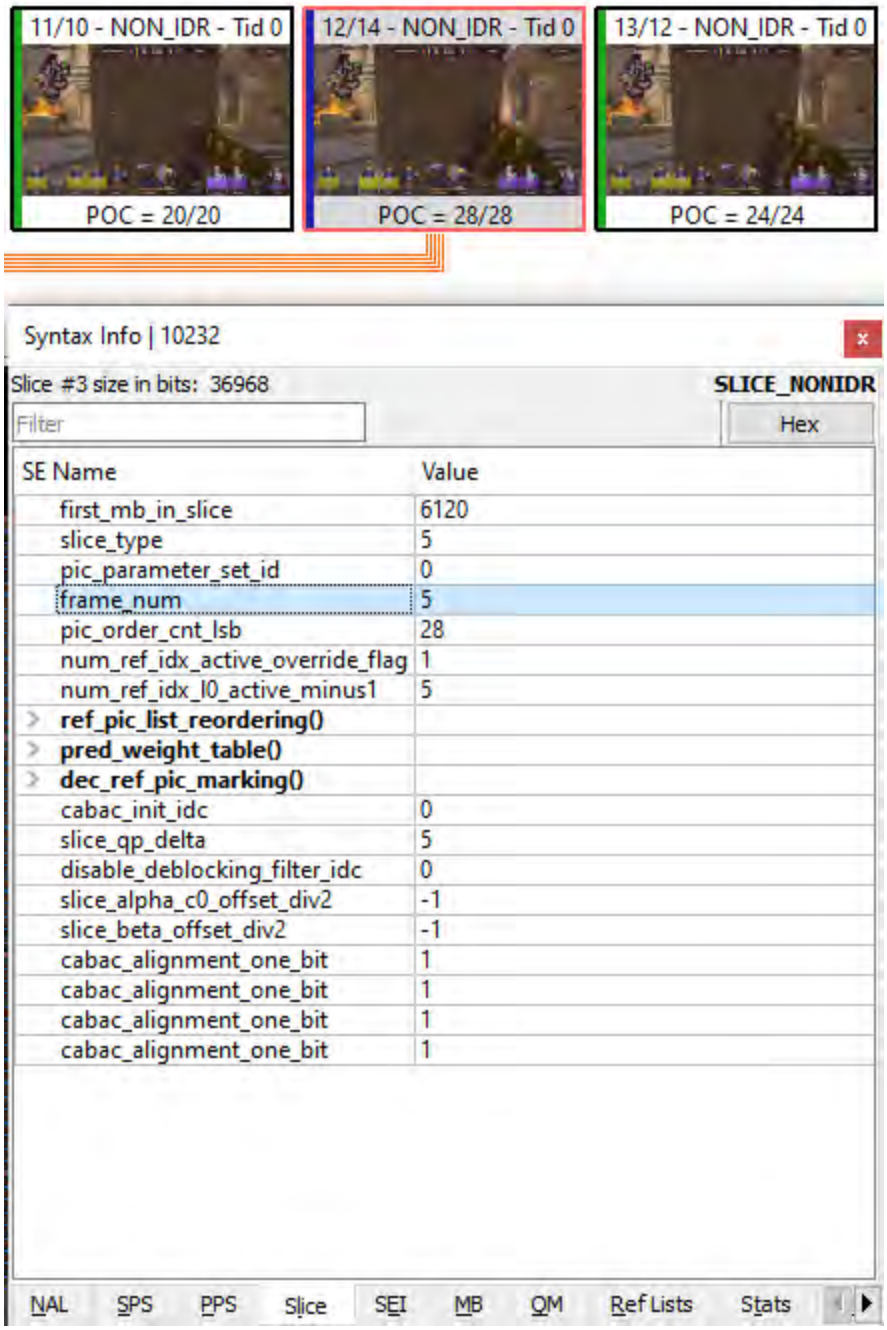
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



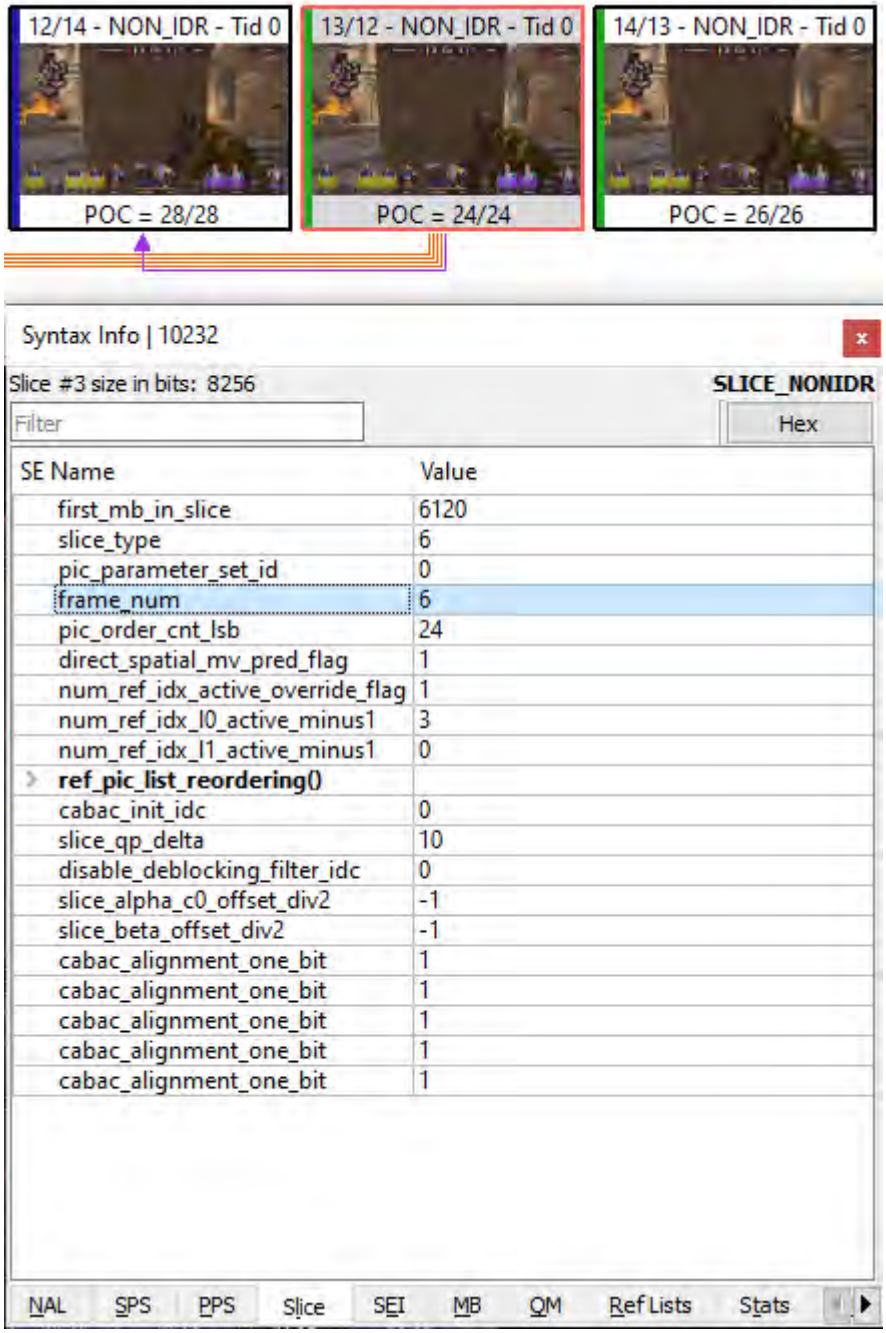
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

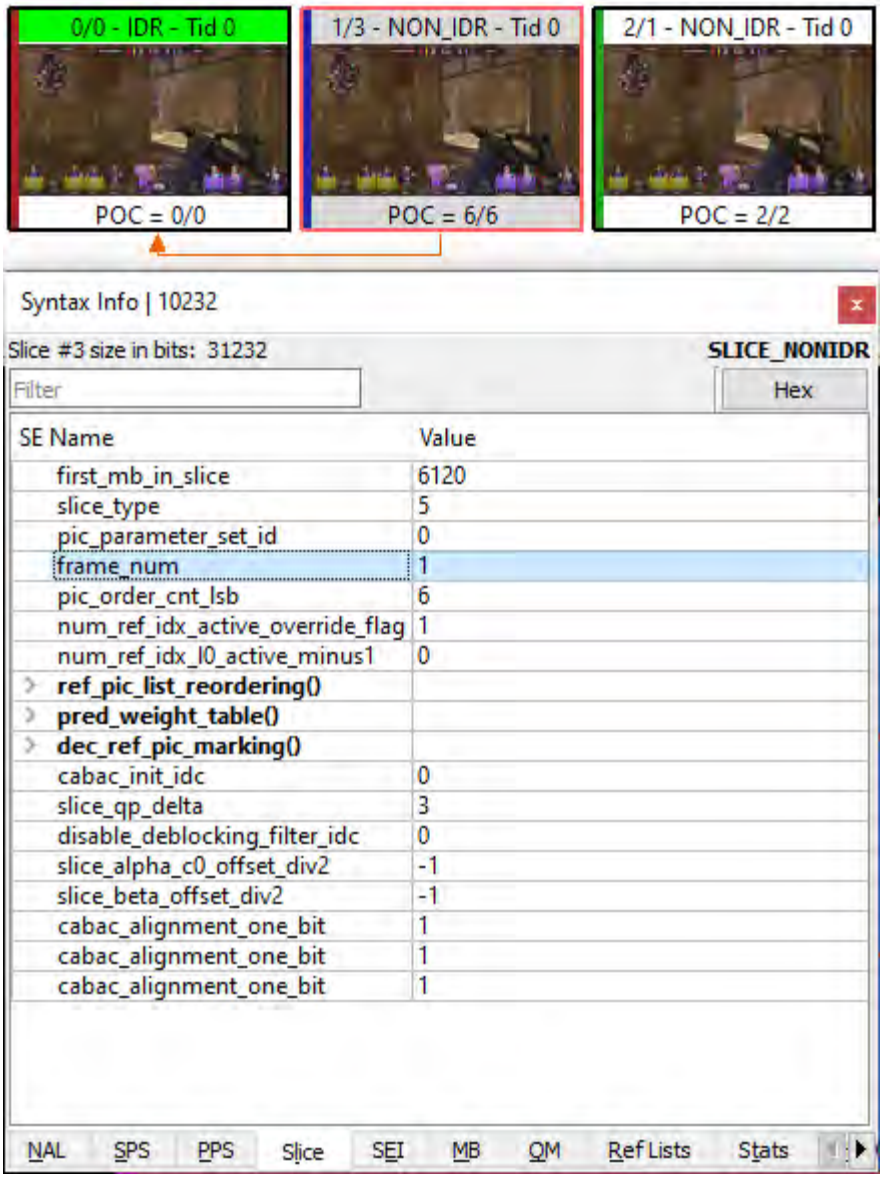


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

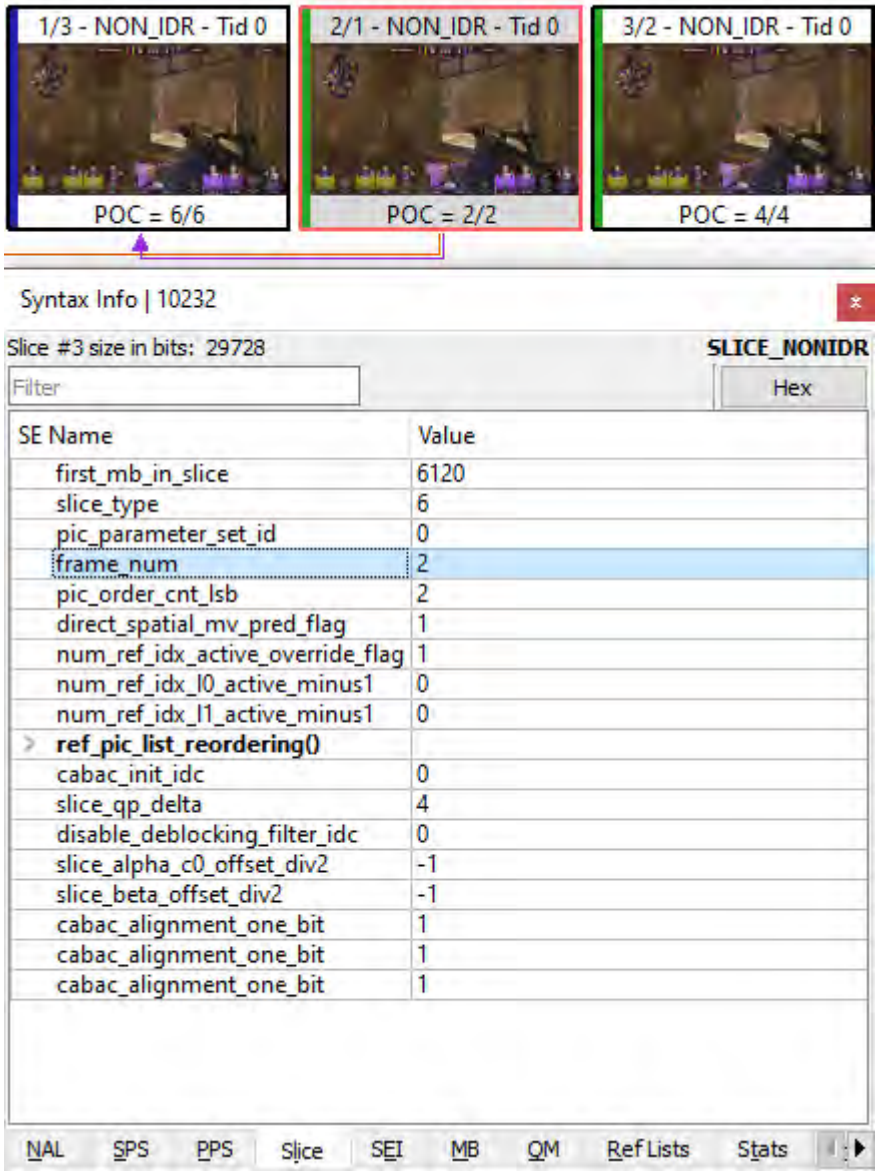
359. When encoding Twitch.tv video, on information and belief, Amazon performs indicating an encoding order of those pictures used to form reference pictures in the encoded video signal with a sequence indicator having an independent numbering scheme, such that consecutive pictures used to form reference pictures in encoding order are assigned sequence

indicator values that differ with respect to each other by a predetermined amount independent of the number of non-reference pictures encoded between successive reference pictures, as demonstrated in the screenshots below using VQ Analyzer software, where the frame number for reference pictures increases by 1 when there are two non-reference pictures between reference pictures.

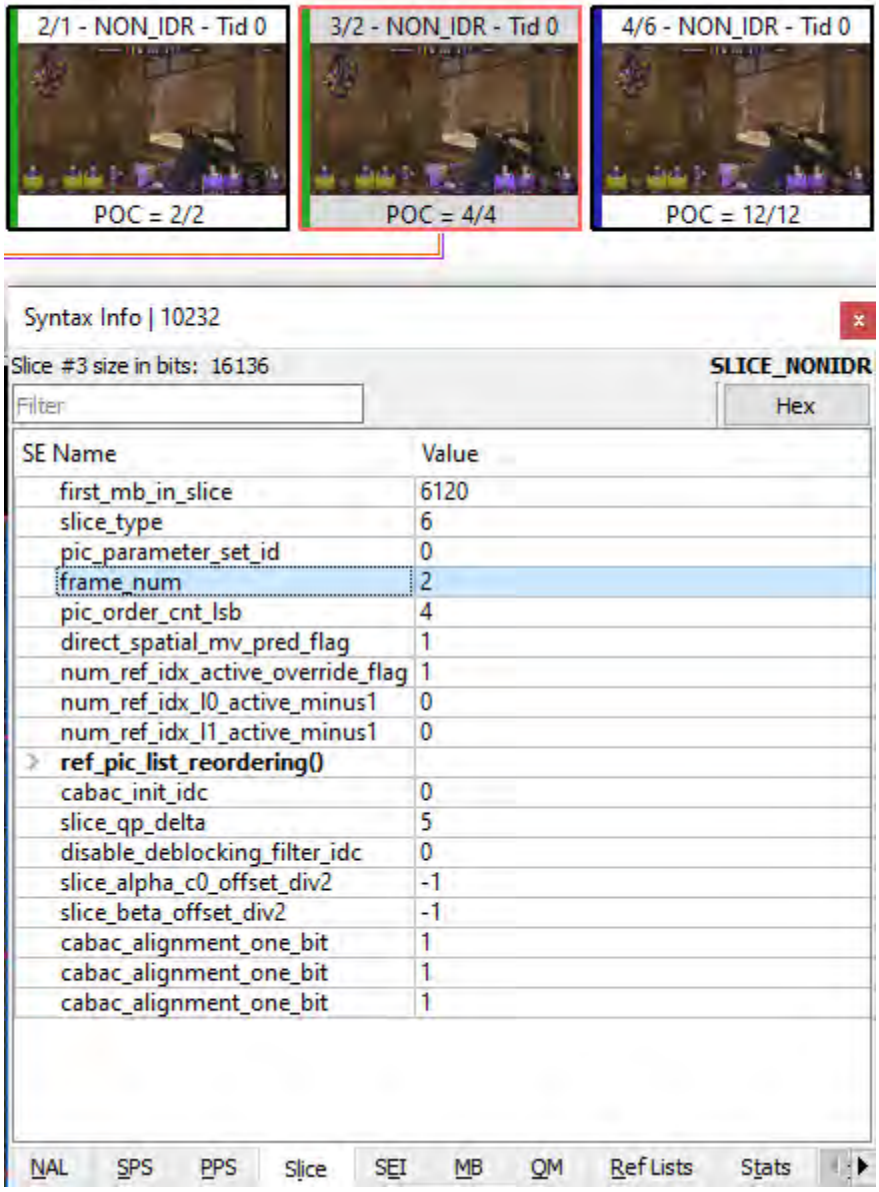
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



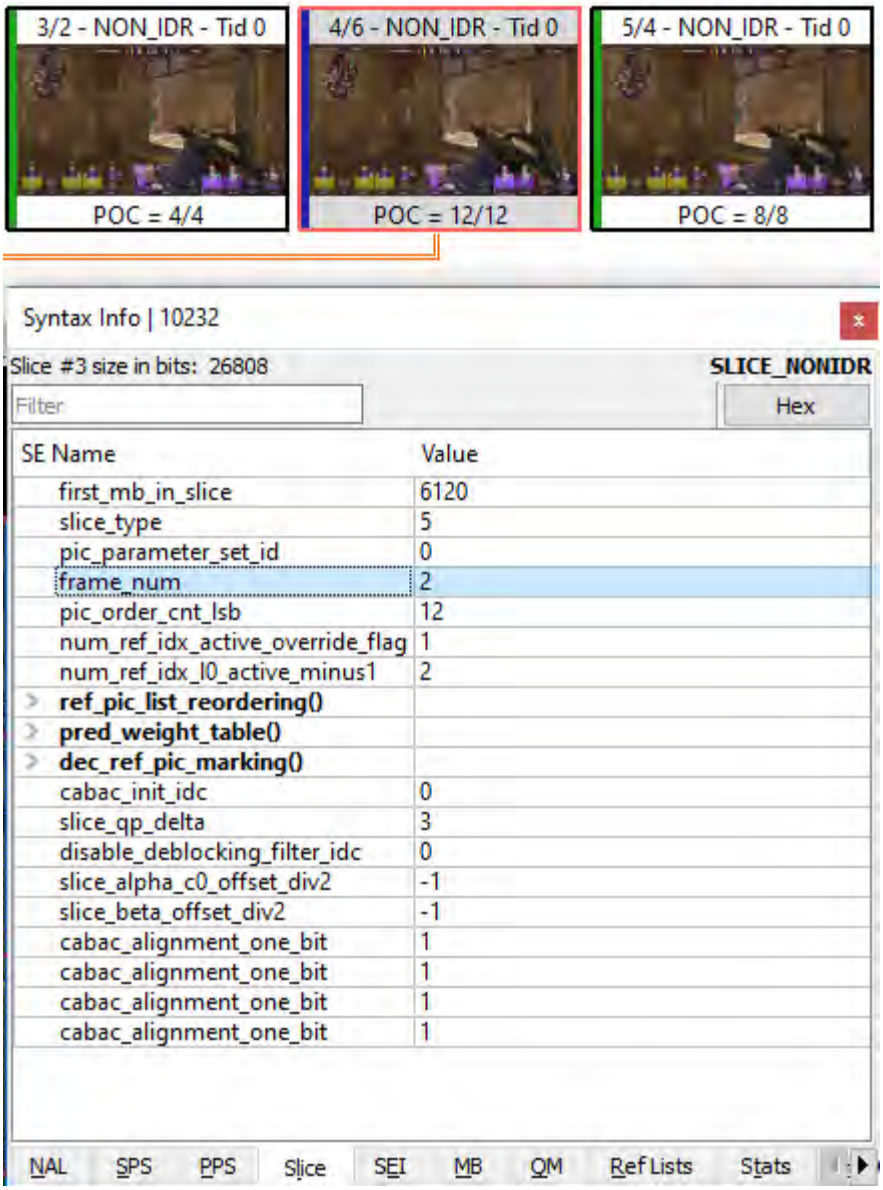
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



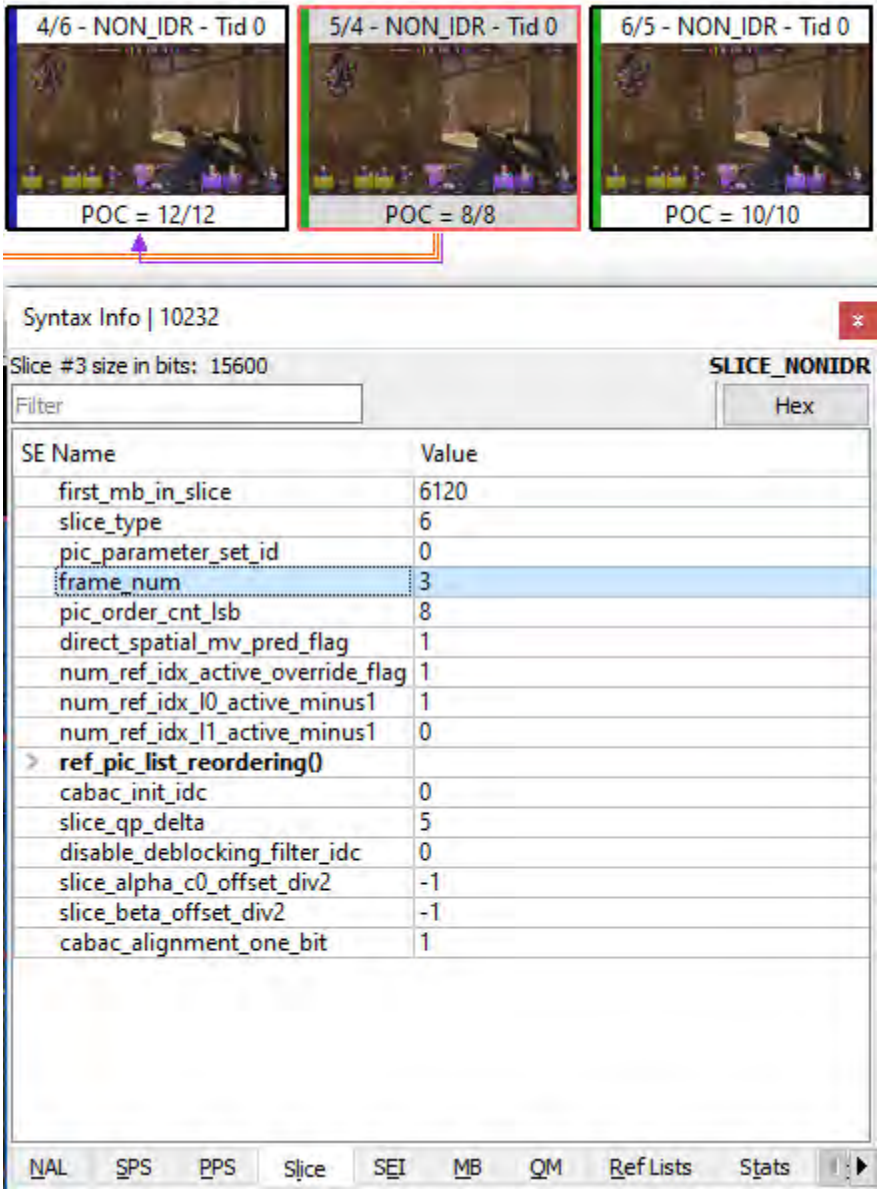
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

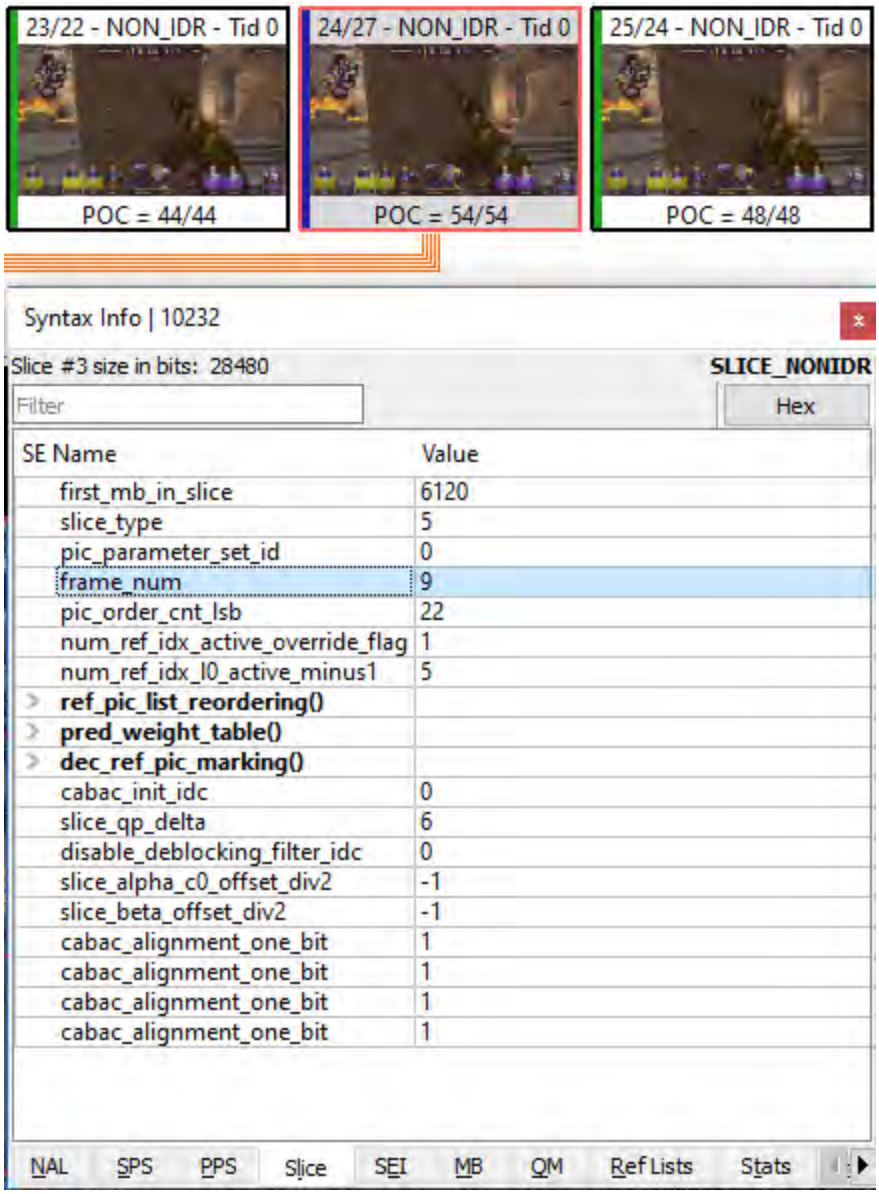


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

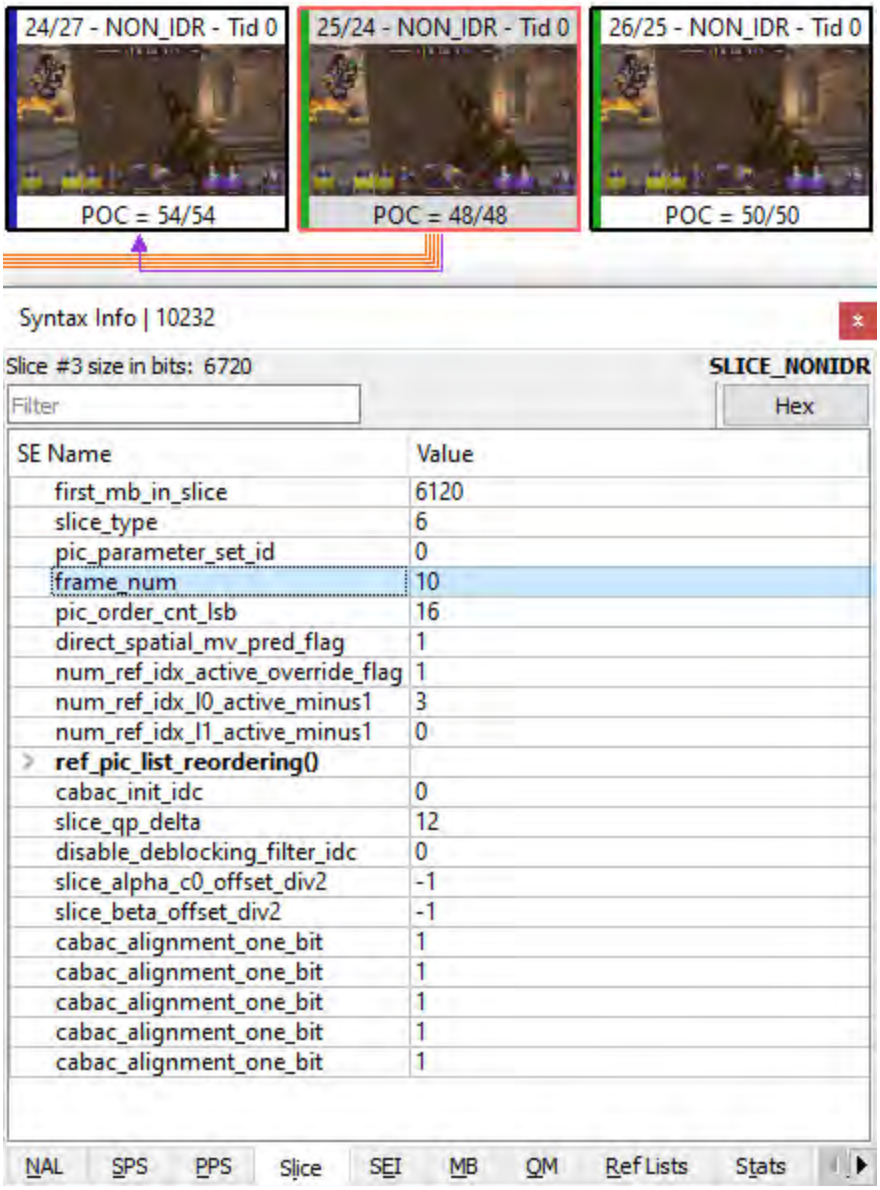
360. When encoding Twitch.tv video, on information and belief, Amazon performs indicating an encoding order of those pictures used to form reference pictures in the encoded video signal with a sequence indicator having an independent numbering scheme, such that consecutive pictures used to form reference pictures in encoding order are assigned sequence indicator values that differ with respect to each other by a predetermined amount independent of the number of non-reference pictures encoded between successive reference pictures, as

demonstrated in the screenshots below using VQ Analyzer software, where the frame number for reference pictures increases by 1 when there are three non-reference pictures between reference pictures.

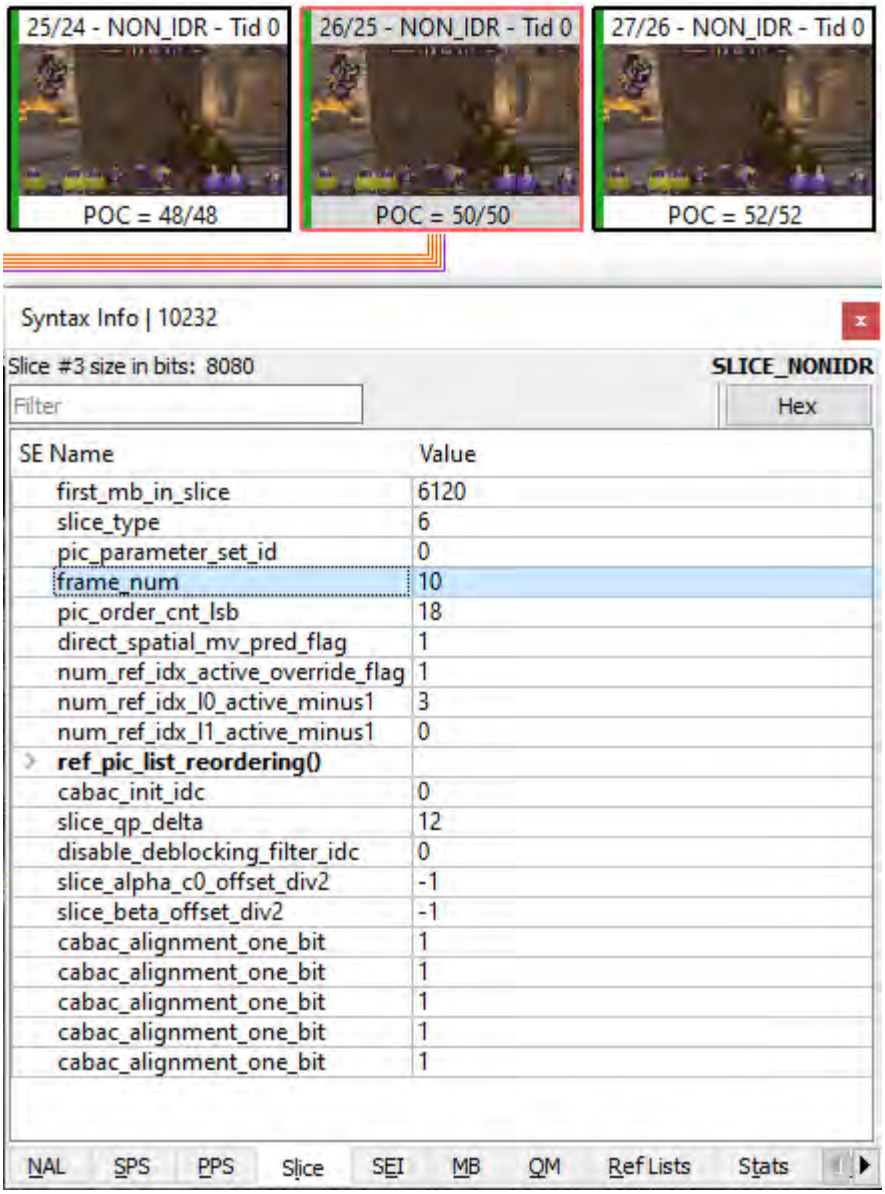
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



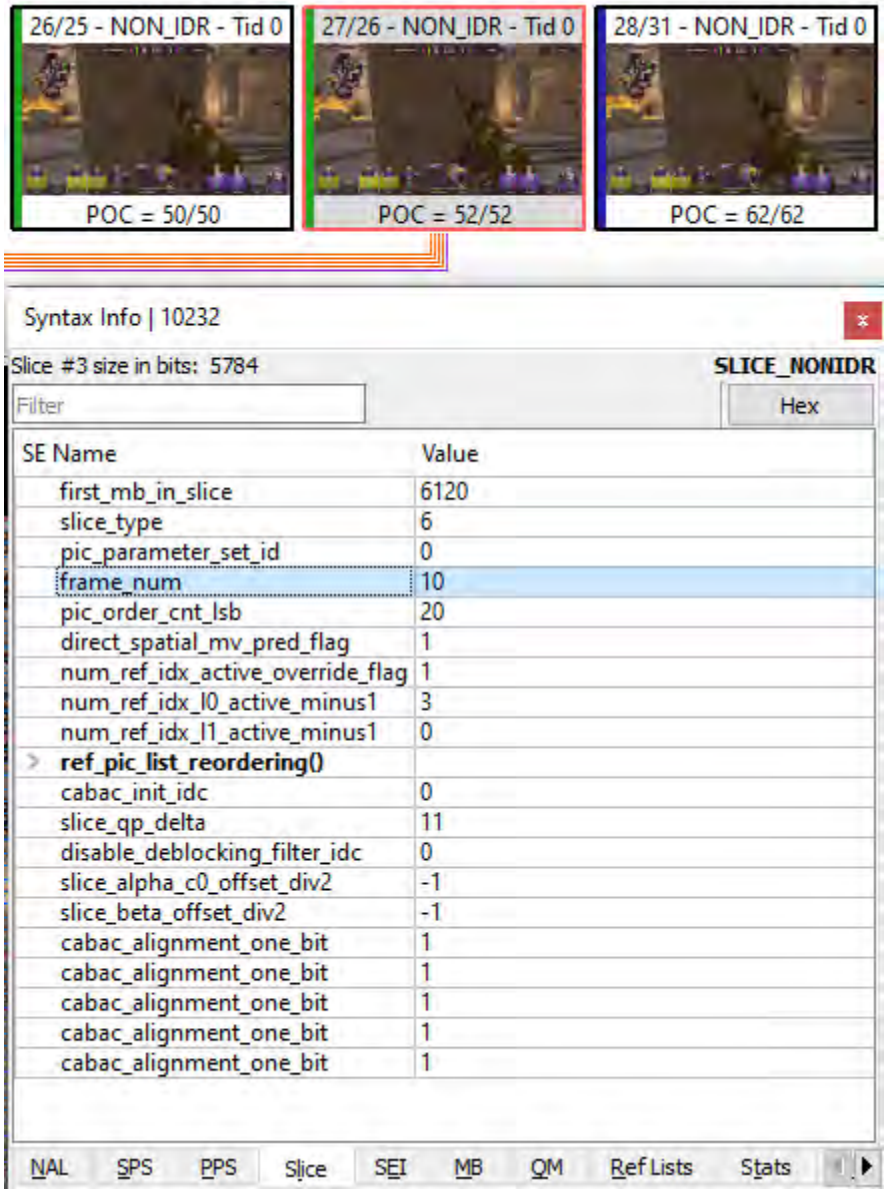
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



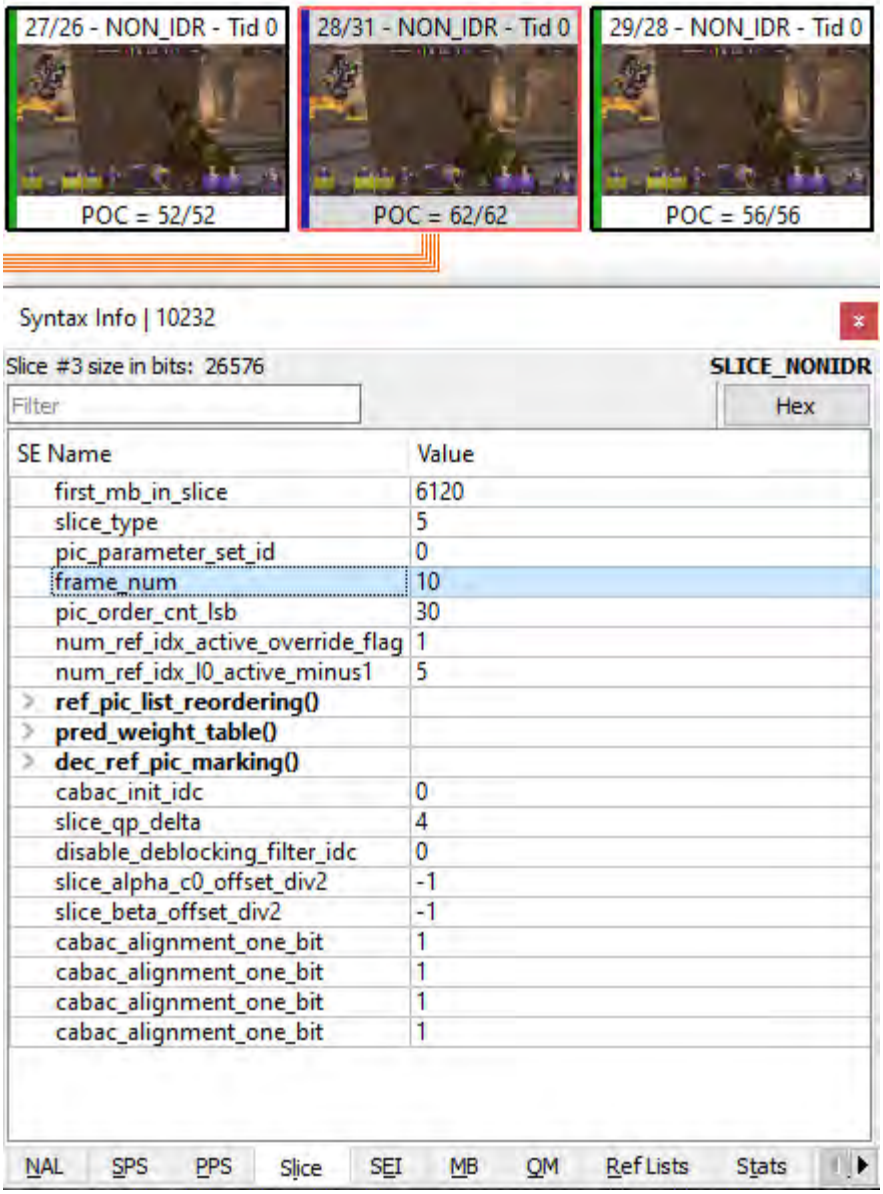
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



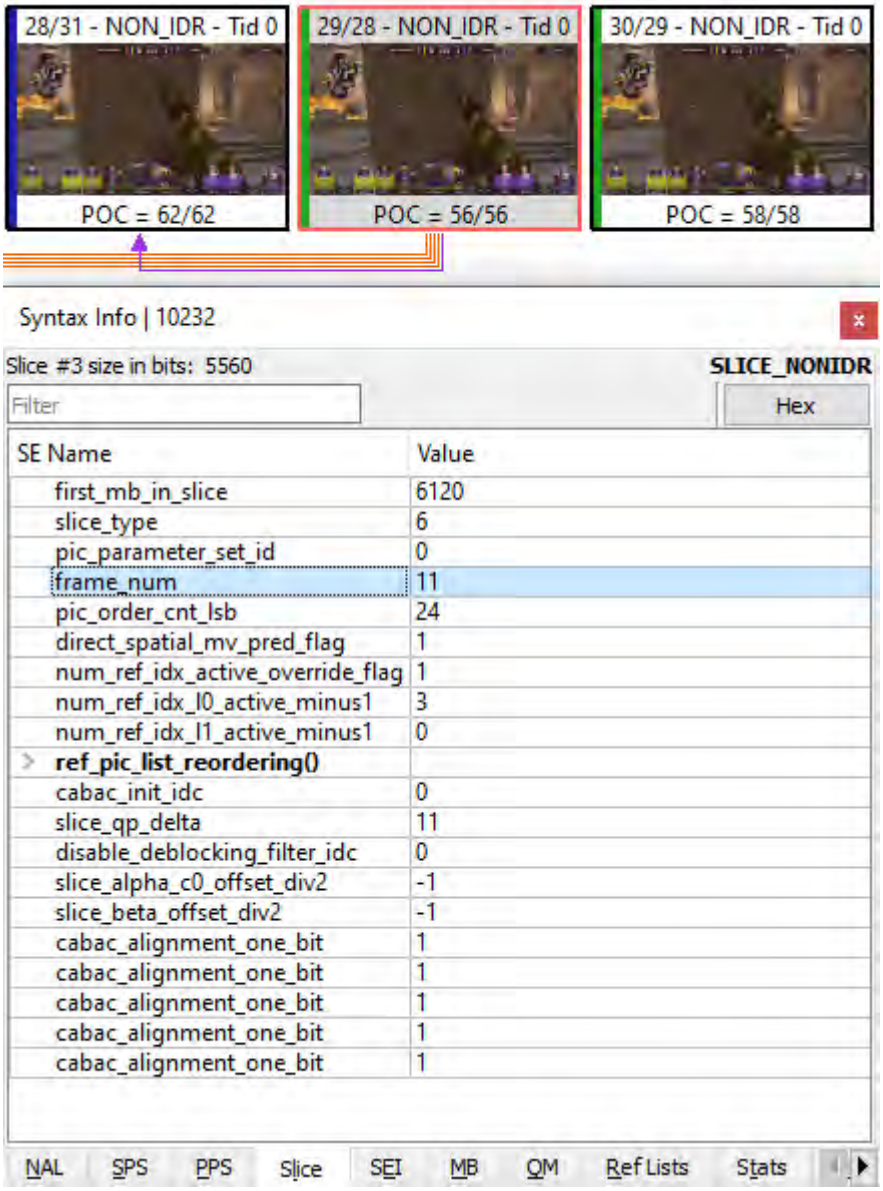
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

361. For another example of infringement, on information and belief, Amazon performs a method of decoding an encoded video signal in a manner that is covered by claim 5 of the ’005 Patent for Amazon Prime Video contents, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder, and as performed, for example, during internal testing of video quality and transcoding. *See supra* at paragraph 212.

Three years ago, the Video Quality Analysis (VQA) group in Prime Video started using machine learning to identify defects in captured content from devices, such as gaming consoles, TVs, and set-top boxes, to validate new application releases or offline changes to encoding profiles. More recently, we've been applying the same techniques to problems such as real-time quality monitoring of our thousands of channels and live events and to analyzing new catalogue content at scale.

Source frame

The initial version of Amazon Prime Video's block corruption detector uses a residual neural network to produce a map indicating the probability of corruption at particular image locations, binarizes that map, and computes the ratio between the corrupted area and the total image area.

Our team at VQA trains computer vision models to watch video and spot issues that may compromise the customer viewing experience, such as blocky frames, unexpected black frames, and audio noise. This enables us to process video at the scale of hundreds of thousands of live events and catalogue items.

https://www.amazon.science/blog/how-prime-video-uses-machine-learning-to-ensure-video-quality.

362. When decoding Amazon Prime Video contents, on information and belief, Amazon performs a method of decoding an encoded video signal representing a sequence of pictures to form a decoded video signal, corresponding to the decoding process specified by the H.264 standard.

3 Definitions

For the purposes of this Recommendation | International Standard, the following definitions apply.

3.1 access unit: A set of *NAL units* always containing exactly one *primary coded picture* **. . .** The decoding of an access unit always results in a *decoded picture*.

. . .

3.14 bitstream: A sequence of bits that forms the representation of *coded pictures* and associated data forming one or more *coded video sequences*. Bitstream is a collective term used to refer either to a *NAL unit stream* or a *byte stream*.

. . .

3.27 coded picture: A *coded representation* of a *picture* **. . .**

. . .

3.30 coded video sequence: A sequence of *access units* that consists, in decoding order, of an *IDR access unit* followed by zero or more non-IDR *access units* including all subsequent *access units* up to but not including any subsequent *IDR access unit*.

. . .

3.37 decoded picture: A *decoded picture* is derived by decoding a *coded picture*. A *decoded picture* is either a decoded *frame*, or a decoded *field*. A decoded *field* is either a decoded *top field* or a decoded *bottom field*.

. . .

3.39 decoder: An embodiment of a *decoding process*.

3.40 decoding order: The order in which *syntax elements* are processed by the *decoding process*.

3.41 decoding process: The process specified in this Recommendation | International Standard that reads a *bitstream* and derives *decoded pictures* from it.

. . .

3.48 field: An assembly of alternate rows of a *frame*. A *frame* is composed of two *fields*, a *top field* and a *bottom field*.

. . .

3.53 frame: A *frame* contains an array of *luma* samples and two corresponding arrays of *chroma* samples. A *frame* consists of two *fields*, a *top field* and a *bottom field*.

. . .

3.61 instantaneous decoding refresh (IDR) access unit: An *access unit* in which the *primary coded picture* is an *IDR picture*.

3.62 instantaneous decoding refresh (IDR) picture: A *coded picture* in which all *slices* are *I* or *SI slices* that causes the *decoding process* to mark all *reference pictures* as "unused for reference" immediately after decoding the IDR picture. After the decoding of an IDR picture all following *coded pictures* in *decoding order* can be decoded without *inter prediction* from any *picture* decoded prior to the IDR picture. The first *picture* of each *coded video sequence* is an IDR picture.

. . .

3.87 NAL unit: A syntax structure containing an indication of the type of data to follow and *bytes* containing that data in the form of an *RBSP* interspersed as necessary with *emulation prevention bytes*.

. . .

3.102 picture: A collective term for a *field* or a *frame*.

. . .

3.109 primary coded picture: The coded representation of a *picture* to be used by the *decoding process* for a bitstream conforming to this Recommendation | International Standard. The primary coded picture contains all *macroblocks* of the *picture*. The only *pictures* that have a normative effect on the *decoding process* are primary coded pictures. See also *redundant coded picture*.

ITU-T Rec. H.264 (3/2005) Advanced video coding for generic audiovisual services at pp. 4-9.

363. When decoding Amazon Prime Video contents, on information and belief, Amazon performs a method of decoding that performs receiving an encoded video signal comprising temporally independent INTRA pictures and temporally predicted pictures, corresponding to the decoding process specified by the H.264 standard.

3 Definitions

For the purposes of this Recommendation | International Standard, the following definitions apply.

3.1 access unit: A set of *NAL units* always containing exactly one *primary coded picture* **. . .** The decoding of an access unit always results in a *decoded picture*.

. . .

3.30 coded video sequence: A sequence of *access units* that consists, in decoding order, of an *IDR access unit* followed by zero or more non-IDR *access units* including all subsequent *access units* up to but not including any subsequent *IDR access unit*.

. . .

3.87 NAL unit: A syntax structure containing an indication of the type of data to follow and *bytes* containing that data in the form of an *RBSP* interspersed as necessary with *emulation prevention bytes*.

. . .

3.61 instantaneous decoding refresh (IDR) access unit: An *access unit* in which the *primary coded picture* is an *IDR picture*.

3.62 instantaneous decoding refresh (IDR) picture: A *coded picture* in which all *slices* are *I* or *SI slices* that causes the *decoding process* to mark all *reference pictures* as "unused

for reference" immediately after decoding the IDR picture. After the decoding of an IDR picture all following *coded pictures* in *decoding order* can be decoded without *inter prediction* from any *picture* decoded prior to the IDR picture. The first *picture* of each *coded video sequence* is an IDR picture.

3.63 inter coding: Coding of a *block*, *macroblock*, *slice*, or *picture* that uses *inter prediction*.

3.64 inter prediction: A *prediction* derived from decoded samples of *reference pictures* other than the current *decoded picture*.

. . .

3.66 intra coding: Coding of a *block*, *macroblock*, *slice*, or *picture* that uses *intra prediction*.

3.67 intra prediction: A *prediction* derived from the decoded samples of the same decoded *slice*.

. . .

3.136 slice: An integer number of *macroblocks* or *macroblock pairs* ordered consecutively in the *raster scan* within a particular *slice group* **. . .**

. . .

3.59 I slice: A *slice* that is not an *SI slice* that is decoded using *prediction* only from decoded samples within the same *slice*.

. . .

3.98 P slice: A *slice* that may be decoded using *intra prediction* from decoded samples within the same *slice* or *inter prediction* from previously-decoded *reference pictures*, using at most one *motion vector* and *reference index* to *predict* the sample values of each *block*.

. . .

3.8 B slice: A *slice* that may be decoded using *intra prediction* from decoded samples within the same *slice* or *inter prediction* from previously-decoded *reference pictures*, using at most two *motion vectors* and *reference indices* to *predict* the sample values of each *block*.

ITU-T Rec. H.264 (3/2005) Advanced video coding for generic audiovisual services at pp. 4-11.

8.1 NAL unit decoding process

Inputs to this process are NAL units.

Outputs of this process are the RBSP syntax structures encapsulated within the NAL units.

> The decoding process for each NAL unit extracts the RBSP syntax structure from the NAL unit and then operates the decoding processes specified for the RBSP syntax structure in the NAL unit as follows.

ITU-T Rec. H.264 (3/2005) Advanced video coding for generic audiovisual services at pp. 97.

364. When decoding Amazon Prime Video contents, on information and belief, Amazon performs a method of decoding that performs wherein the INTRA pictures and at least some of the temporally predicted pictures are used to form reference pictures for the temporal prediction of other pictures, corresponding to the decoding process specified by the H.264 standard.

> **3 Definitions**
>
> For the purposes of this Recommendation | International Standard, the following definitions apply.
>
> . . .
>
> **3.94 non-reference picture:** A *picture* coded with nal_ref_idc equal to 0. A *non-reference picture* is not used for *inter prediction* of any other *pictures.*
>
> . . .
>
> **3.121 reference picture:** A *picture* with nal_ref_idc not equal to 0. A *reference picture* contains samples that may be used for *inter prediction* in the *decoding process* of subsequent *pictures* in *decoding order.*

ITU-T Rec. H.264 (3/2005) Advanced video coding for generic audiovisual services at pp. 4, 9-10.

> **7.4.1 NAL unit semantics**
>
> . . .

nal_ref_idc not equal to 0 specifies that the content of the NAL unit contains a sequence parameter set or a picture parameter set or a slice of a reference picture or a slice data partition of a reference picture.

nal_ref_idc equal to 0 for a NAL unit containing a slice or slice data partition indicates that the slice or slice data partition is part of a non-reference picture.

. . .

nal_ref_idc shall not be equal to 0 for IDR NAL units, i.e., NAL units with nal_unit_type equal to 5.

ITU-T Rec. H.264 (3/2005) Advanced video coding for generic audiovisual services at pp. 56.

7.4.3 Slice header semantics

. . .

slice_type specifies the coding type of the slice according to Table 7-6.

Table 7-6 – Name association to slice_type

slice_type	Name of slice_type
0	P (P slice)
1	B (B slice)
2	I (I slice)
3	SP (SP slice)
4	SI (SI slice)
5	P (P slice)
6	B (B slice)
7	I (I slice)
8	SP (SP slice)
9	SI (SI slice)

. . .

When nal_unit_type is equal to 5 (IDR picture), slice_type shall be equal to 2, 4, 7, or 9.

ITU-T Rec. H.264 (3/2005) Advanced video coding for generic audiovisual services at pp. 75.

365. When decoding Amazon Prime Video contents, on information and belief, Amazon performs a method of decoding an encoded video signal further comprising a sequence indicator having an independent numbering scheme such that consecutive reference pictures in

encoding order are assigned sequence indicator values that differ with respect to each other by a predetermined amount independent of the number of non-reference pictures encoded between successive reference pictures, corresponding to the decoding process specified by the H.264 standard.

7.3.3 Slice header syntax

slice_header( ) {	C	Descriptor
first_mb_in_slice	2	ue(v)
slice_type	2	ue(v)
pic_parameter_set_id	2	ue(v)
frame_num	2	u(v)
if(!frame_mbs_only_flag) {		
field_pic_flag	2	u(1)
if(field_pic_flag)		
bottom_field_flag	2	u(1)
}		

ITU-T Rec. H.264 (3/2005) Advanced video coding for generic audiovisual services at pp. 45.

7.4.3 Slice header semantics

. . .

frame_num is used as an identifier for pictures and shall be represented by log2_max_frame_num_minus4 + 4 bits in the bitstream. frame_num is constrained as follows:

The variable PrevRefFrameNum is derived as follows.

– If the current picture is an IDR picture, PrevRefFrameNum is set equal to 0.

. . .

– Otherwise, PrevRefFrameNum is set equal to the value of frame_num for the previous access unit in decoding order that contained a reference picture.

The value of frame_num is constrained as follows.

– If the current picture is an IDR picture, frame_num shall be equal to 0.

– Otherwise (the current picture is not an IDR picture), referring to the primary coded picture in the previous access unit in decoding order that contains a reference picture as the preceding reference picture, the value of frame_num for the current picture shall not be equal to PrevRefFrameNum . . .

. . .

– The value of frame_num is constrained as follows.

– If gaps_in_frame_num_value_allowed_flag is equal to 0, the value of frame_num for the current picture shall be equal to (PrevRefFrameNum + 1) % MaxFrameNum.

ITU-T Rec. H.264 (3/2005) Advanced video coding for generic audiovisual services at pp. 75-76.

366. When decoding Amazon Prime Video contents, on information and belief, Amazon performs a method of decoding that performs decoding received encoded pictures, examining each decoded picture that forms a reference picture to identify the sequence indicator value assigned to the reference picture and comparing the sequence indicator values assigned to consecutively decoded reference pictures to detect loss of a reference picture, corresponding to the decoding process specified by the H.264 standard.

8 Decoding process

Outputs of this process are decoded samples of the current picture (sometimes referred to by the variable CurrPic).

This clause describes the decoding process, given syntax elements and upper-case variables from clause 7.

. . .

An overview of the decoding process is given as follows.

– The decoding of NAL units is specified in subclause 8.1.

– The processes in subclause 8.2 specify decoding processes using syntax elements in the slice layer and above.

. . .

8.1 NAL unit decoding process

Inputs to this process are NAL units.

Outputs of this process are the RBSP syntax structures encapsulated within the NAL units.

> The decoding process for each NAL unit extracts the RBSP syntax structure from the NAL unit and then operates the decoding processes specified for the RBSP syntax structure in the NAL unit as follows.
>
> . . .
>
> **8.2 Slice decoding process**
>
> . . .
>
> **8.2.5.2 Decoding process for gaps in frame_num**
>
> This process is invoked when frame_num is not equal to PrevRefFrameNum and is not equal to (PrevRefFrameNum + 1) % MaxFrameNum.
>
> . . .
>
> NOTE 2 – This process can only be invoked for a conforming bitstream when gaps_in_frame_num_value_allowed_flag is equal to 1. When gaps_in_frame_num_value_allowed_flag is equal to 0 and frame_num is not equal to PrevRefFrameNum and is not equal to (PrevRefFrameNum + 1) % MaxFrameNum, the decoding process should infer an unintentional loss of pictures.

ITU-T Rec. H.264 (3/2005) Advanced video coding for generic audiovisual services at pp. 96-97, 130.

367. For another example, on information and belief, Amazon performs a method of decoding an encoded video signal in a manner that is substantially the same as described above for Amazon.com video advertisements, such that it is covered by claim 5 of the ’005 Patent, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder, and as performed, for example, during internal testing of video quality and transcoding. *See supra* at paragraph 217.

368. On information and belief, Amazon performs internal testing of video quality on Amazon.com and transcoding of video on Amazon.com in a manner substantially similar to that which is performed on Amazon Prime Video. *See, e.g.*, https://www.amazon.science/blog/how-prime-video-uses-machine-learning-to-ensure-video-quality.

369. For another example, on information and belief, Amazon performs a method of decoding an encoded video signal in a manner that is substantially the same as described above for Twitch.tv video, such that it is covered by claim 5 of the ’005 Patent, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder, and as performed, for example, during internal testing of video quality and transcoding. *See supra* at paragraph 222.

> If you're a Twitch user, you probably know that Twitch is a pretty big site that streams live video, but you may not have an idea of the sheer scale and complexity of all of the different moving parts needed to hold everything together.
>
> Among other things, we have:
>
> - One of the largest live video distribution systems in the world
> - A real-time chat system
> - Web services that provide access to functionality and data
> - Data storage systems
> - Client applications on the web, and on a multitude of platforms — mobile and console in particular
> - Data science infrastructure
> - Internal tooling and systems — configuration management, deployment systems, hardware and software provisioning, testing and QA
> - Network infrastructure that keeps the bits between all of these systems and the end users flowing

https://blog.twitch.tv/en/2015/12/18/twitch-engineering-an-introduction-and-overview-a23917b71a25/.

Transcoding Options FAQ

When you start a stream, Twitch may transform your broadcast into multiple quality formats to meet the needs of devices and audiences around the world. These quality options are called transcodes. Not everyone has the bandwidth to support a 1080p60 broadcast, and network conditions can change over time.

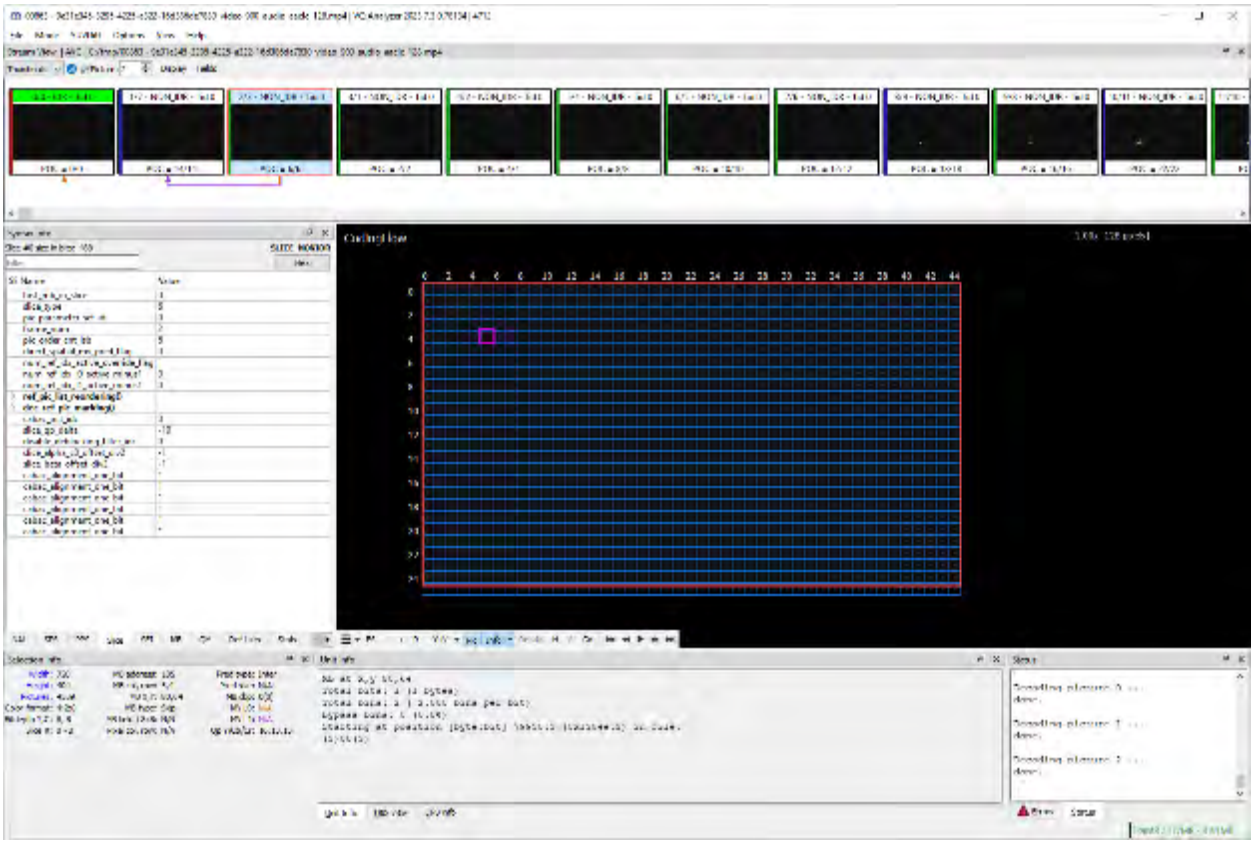
https://help.twitch.tv/s/article/transcoding-options-faq?language=en_US

J. Amazon Makes, Imports, Uses, Sells, and/or Offers for Sale Products and Services that Infringe the ’764 Patent

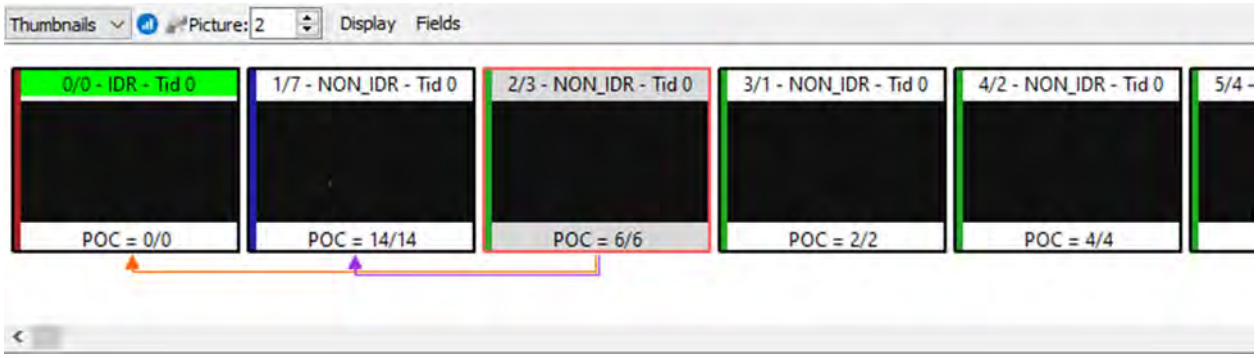
370. The Accused Products infringe one or more claims of the ’764 Patent, including, for example, claim 1.

371. As just one example of infringement, on information and belief, Amazon performs a method of encoding a video signal in a manner that is covered by claim 1 of the ’764 Patent for Amazon Prime Video content (such as trailers), as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 212.

372. When encoding Amazon Prime Video trailers, Amazon performs encoding a video signal using an encoder to form an encoded video signal, the video signal representing a sequence of pictures, as demonstrated in the screenshots below using VQ Analyzer software.



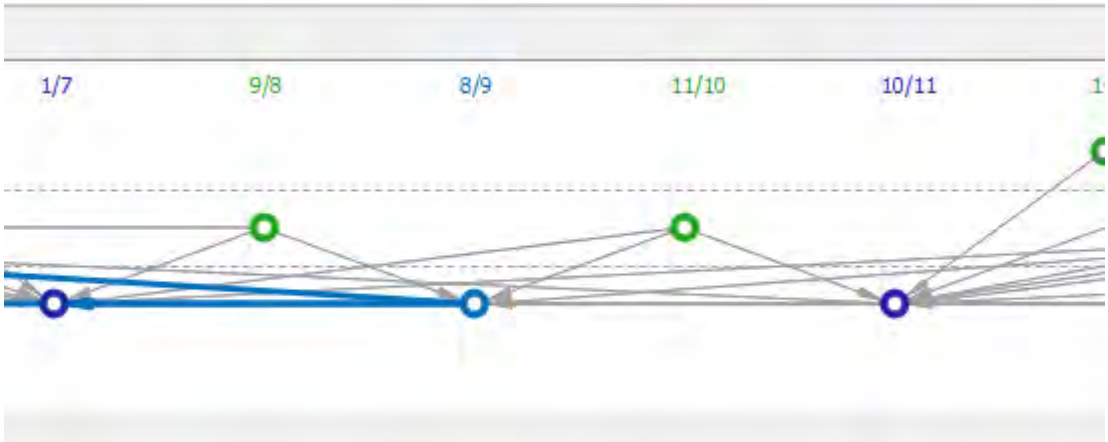
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



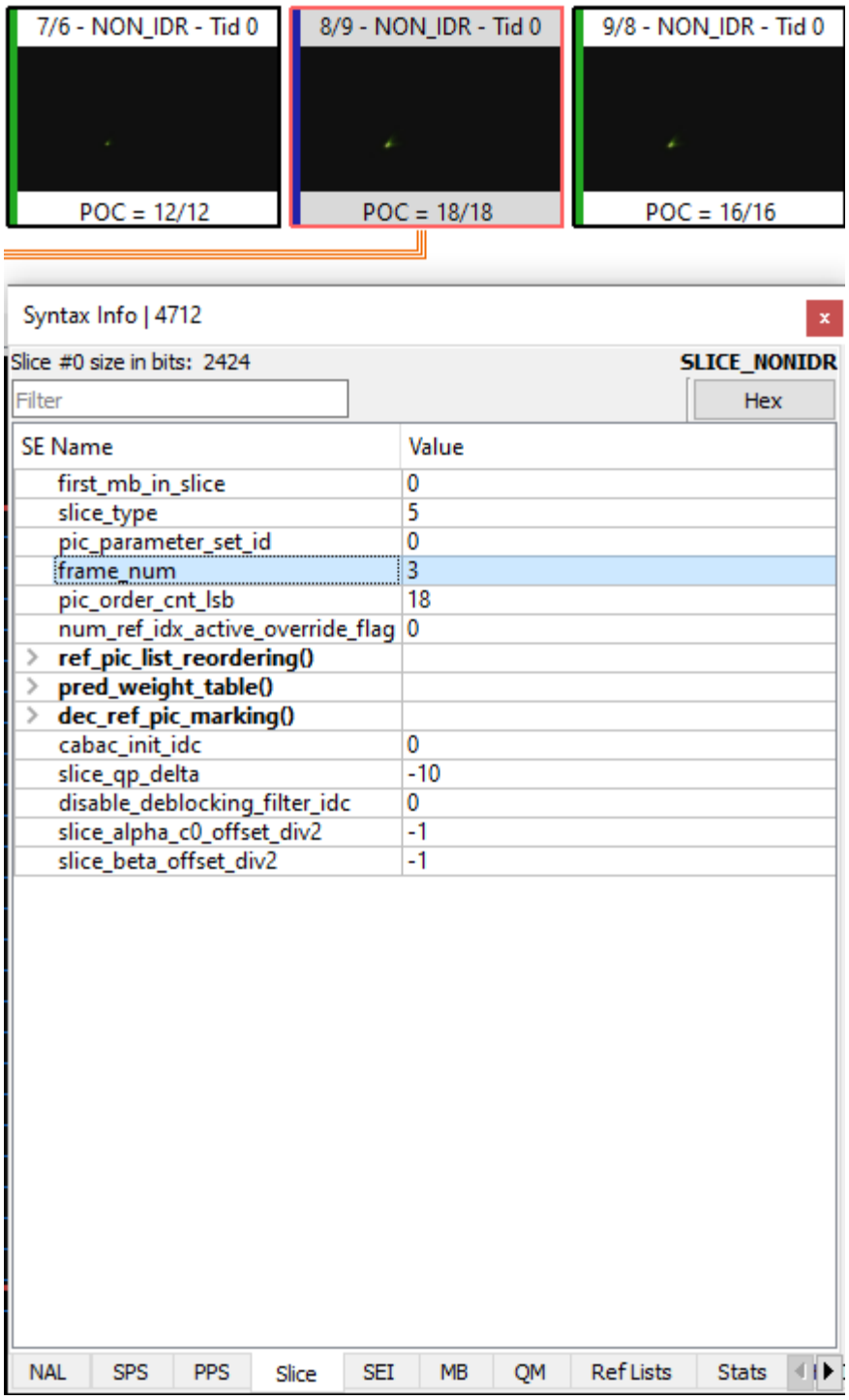
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

373. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs using an independent numbering scheme, to assign consecutive reference pictures in encoding order with respective sequence indicator values that differ with respect to each other by a predetermined amount independent of one or more of the number of non-reference pictures

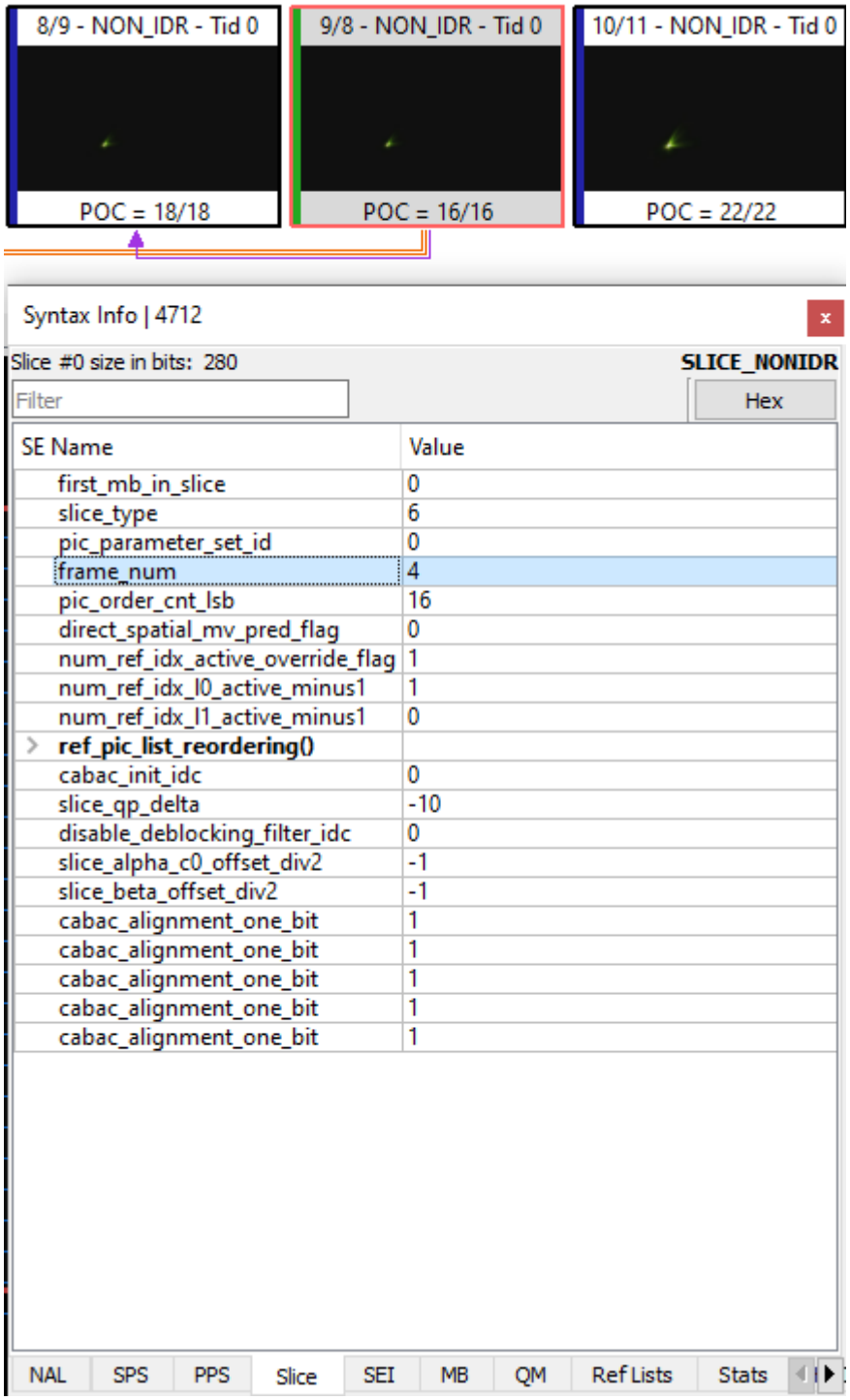
encoded between consecutive reference pictures and the number of non-coded pictures between consecutive reference pictures, as demonstrated in the screenshots below using VQ Analyzer software, where the frame number for reference pictures increases by 1 when there is one non-reference picture between reference pictures.



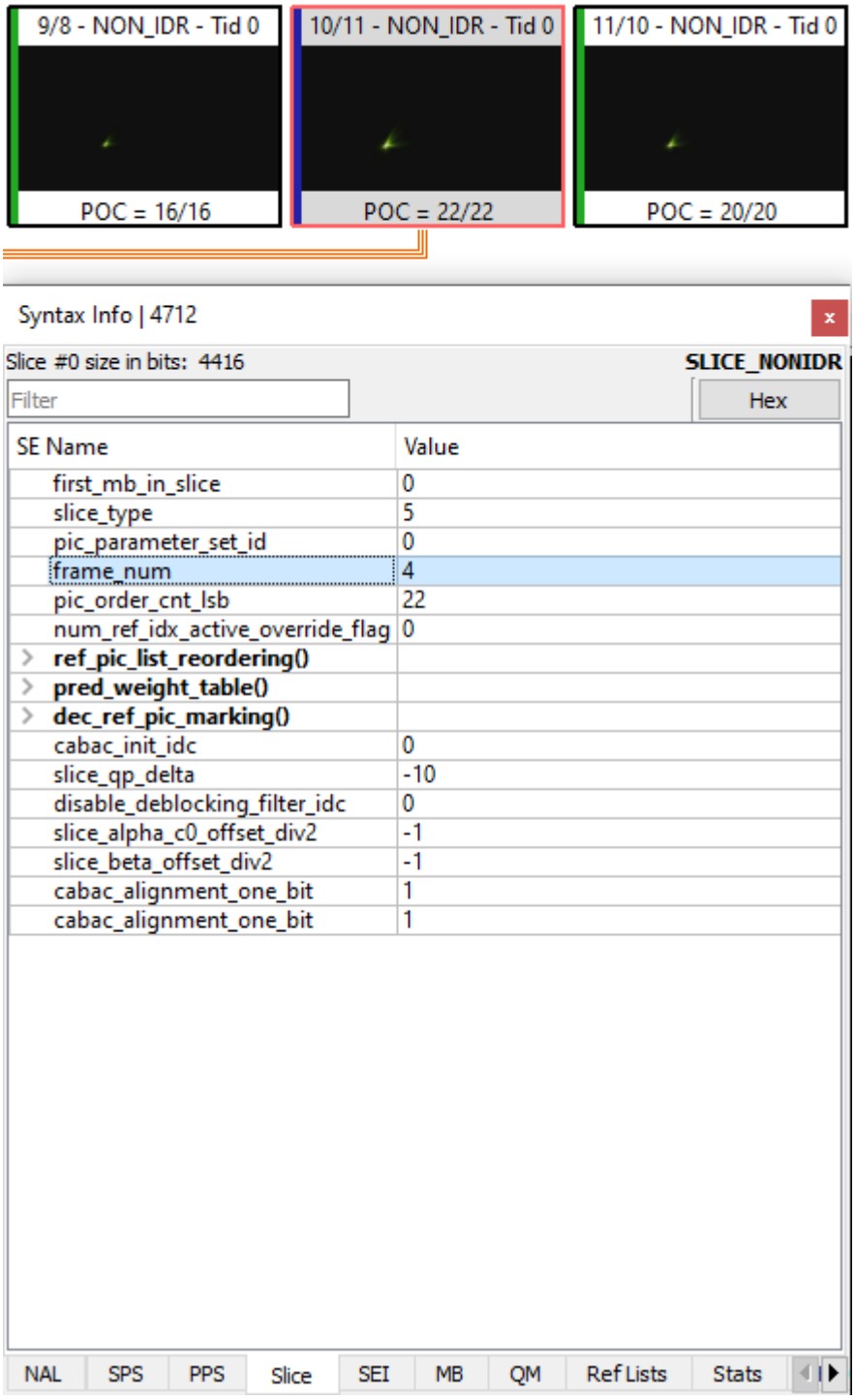
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



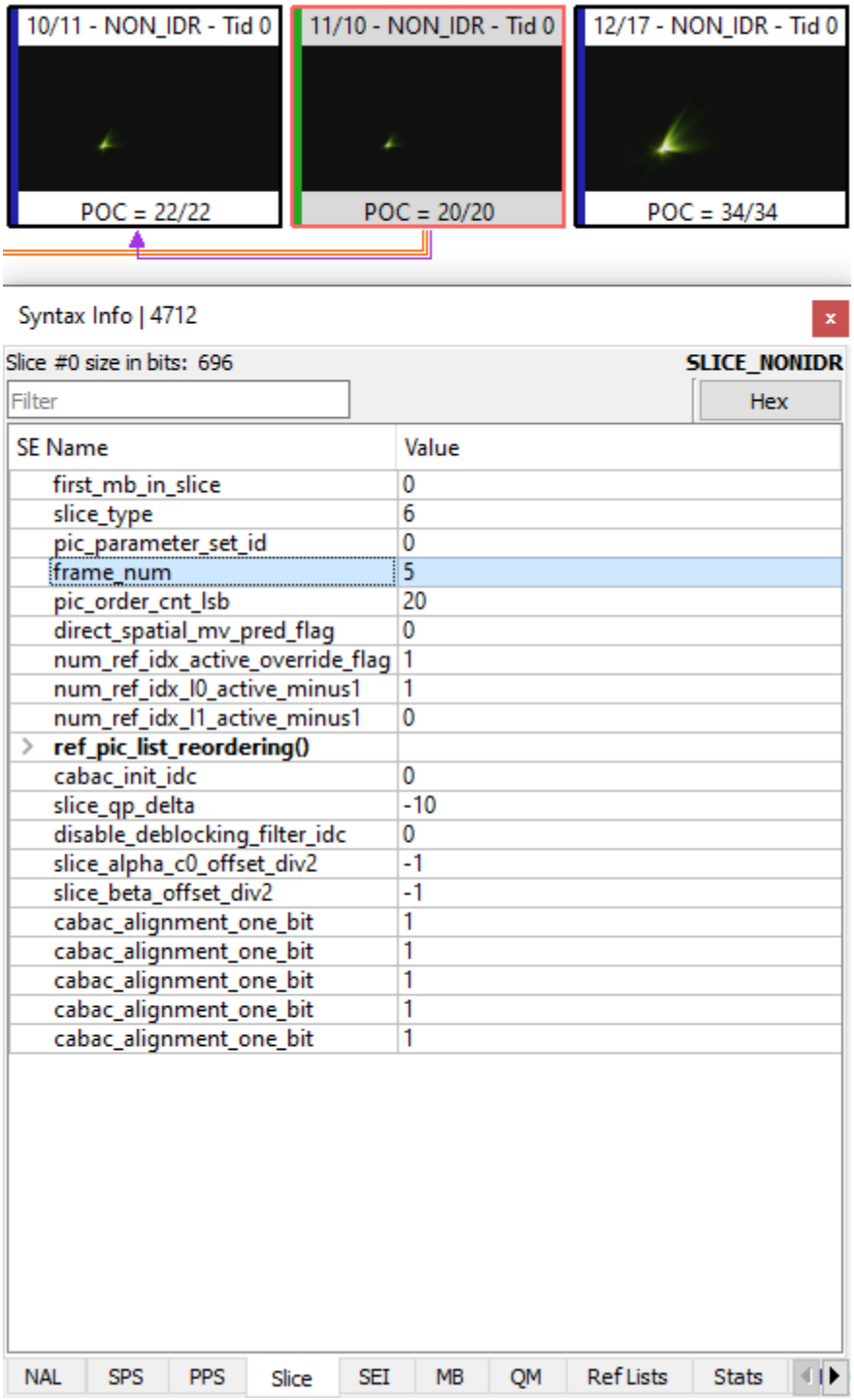
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



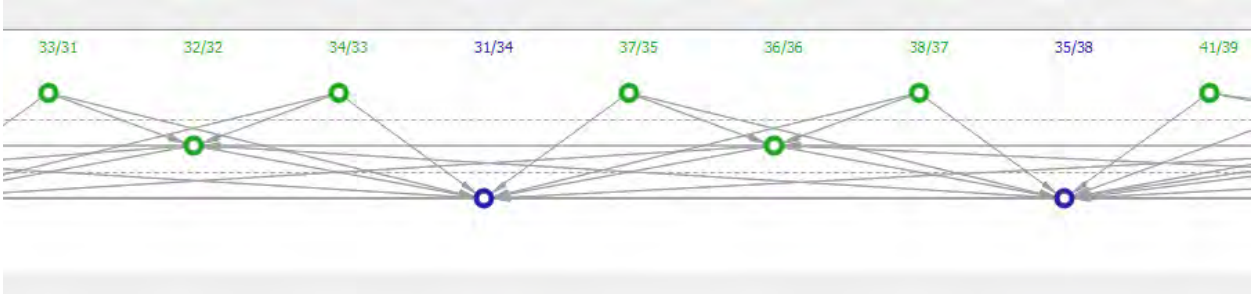
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



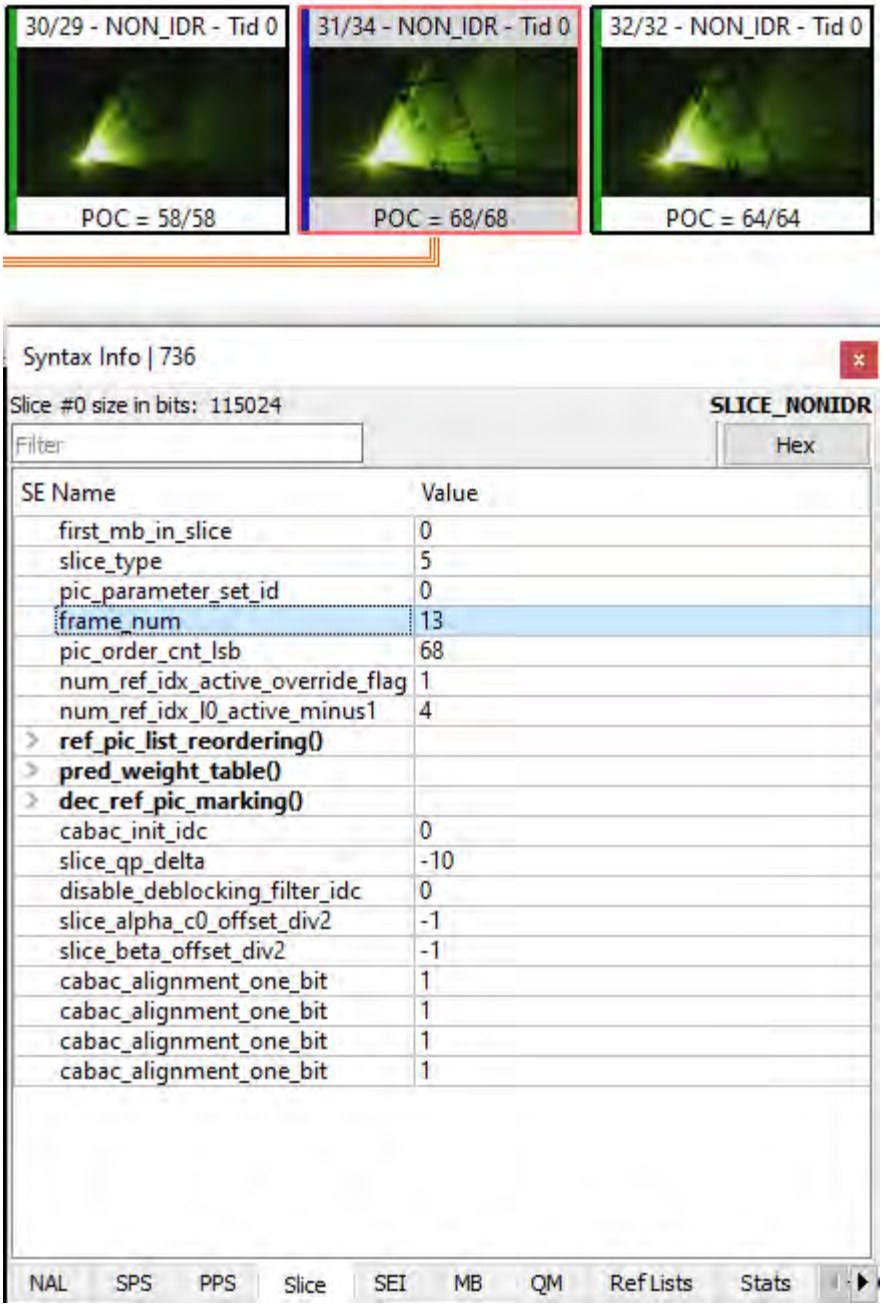
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

374. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs using an independent numbering scheme, to assign consecutive reference pictures in

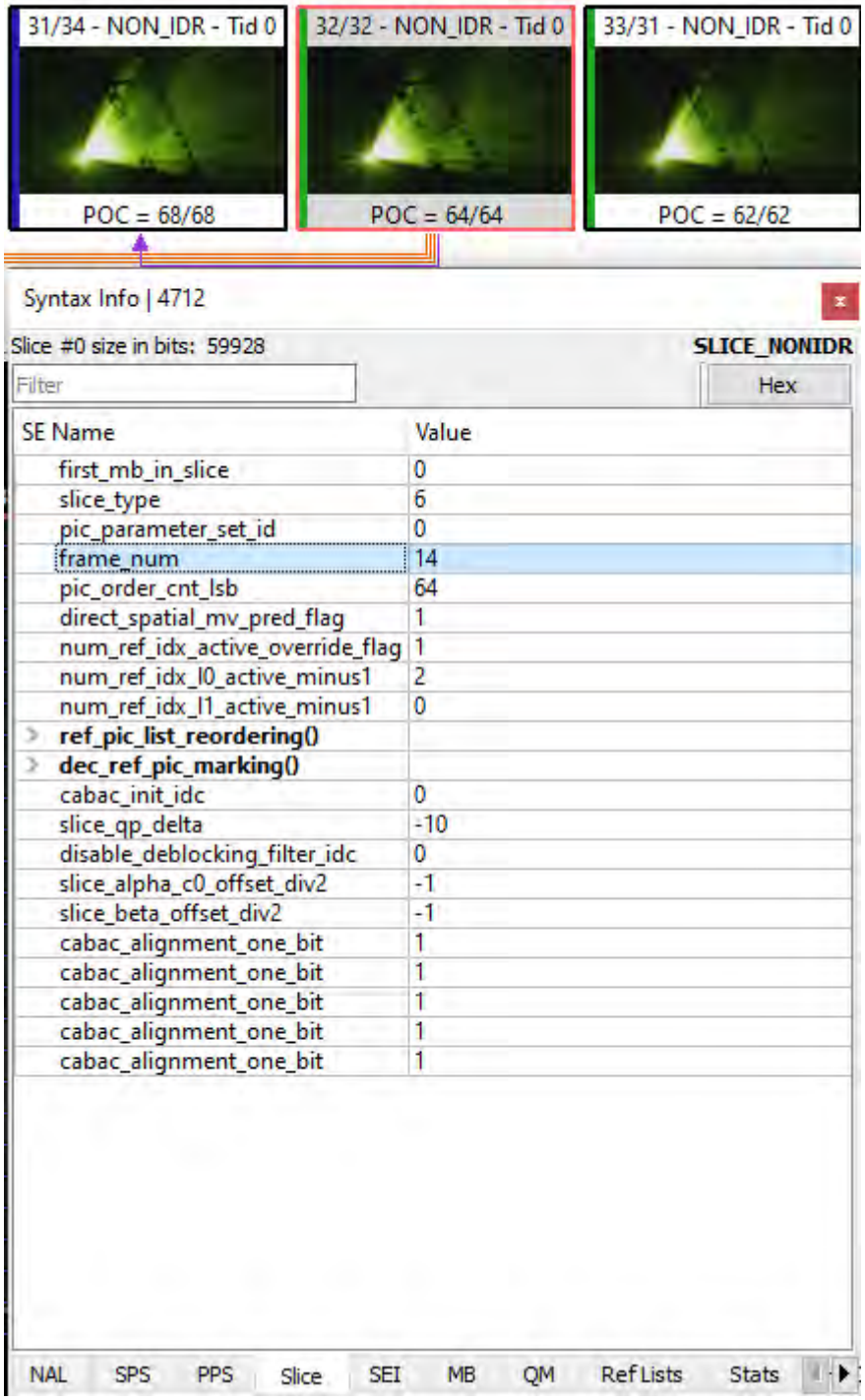
encoding order with respective sequence indicator values that differ with respect to each other by a predetermined amount independent of one or more of the number of non-reference pictures encoded between consecutive reference pictures and the number of non-coded pictures between consecutive reference pictures, as demonstrated in the screenshots below using VQ Analyzer software, where the frame number for reference pictures increases by 1 when there are two non-reference pictures between reference pictures.



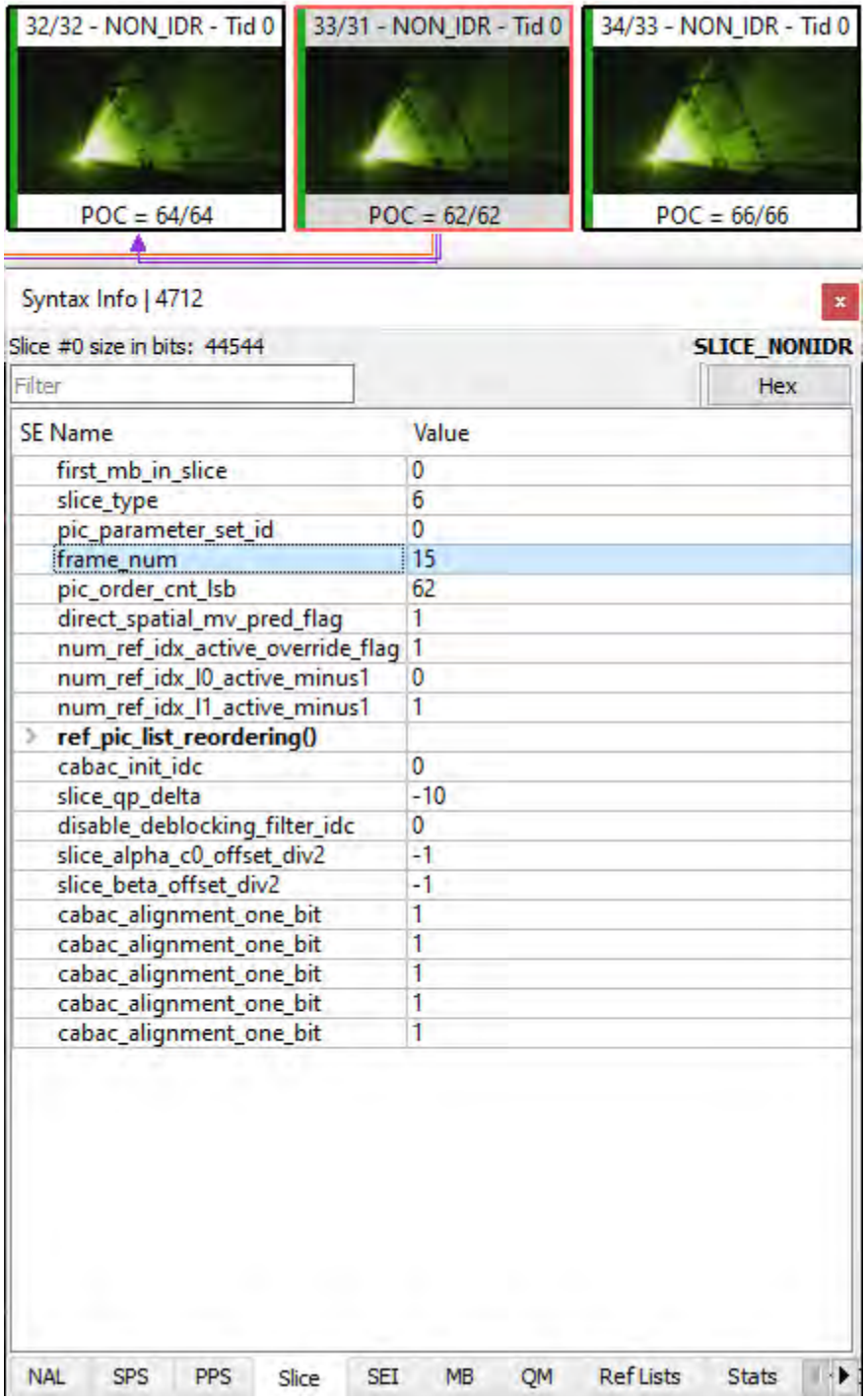
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



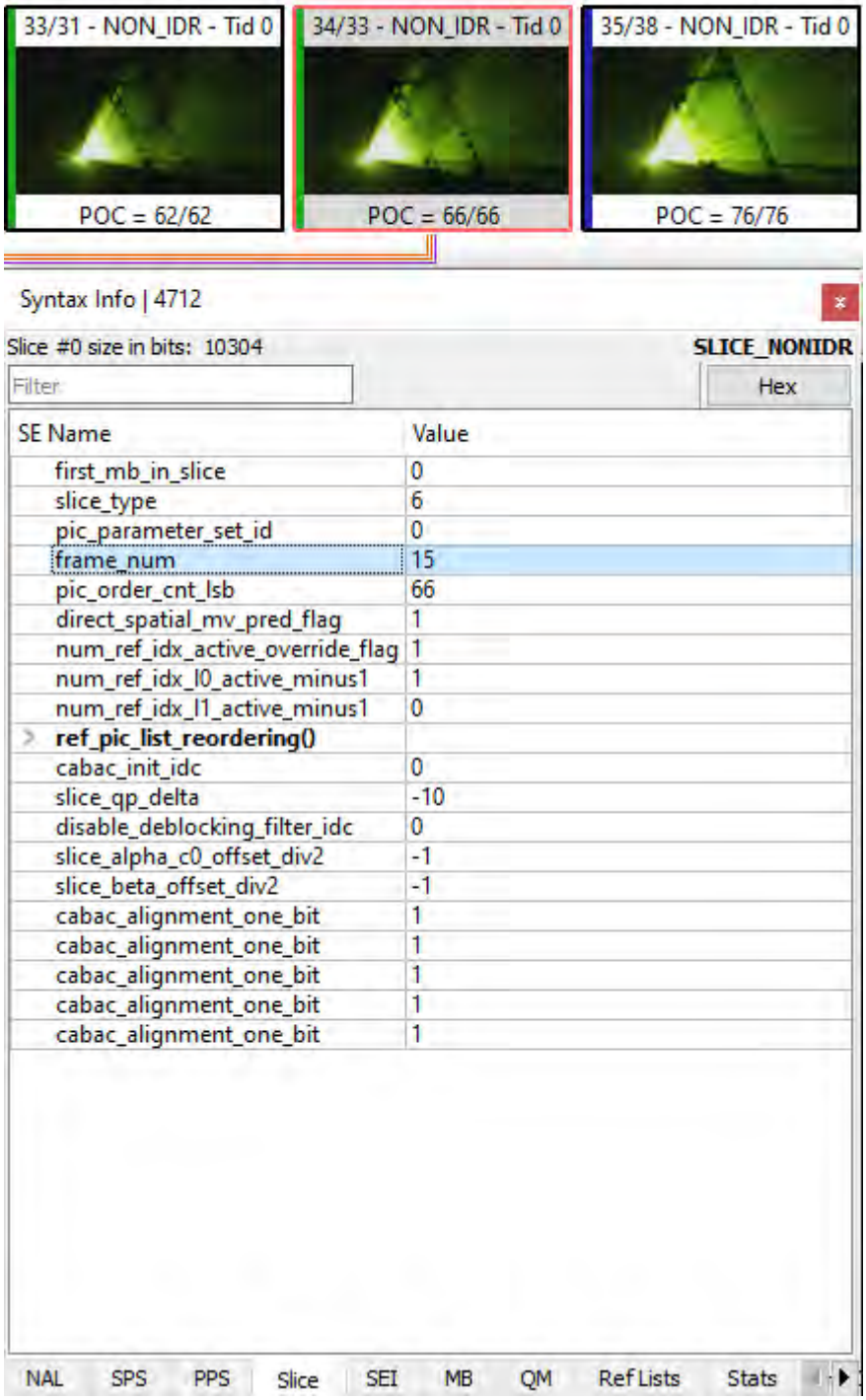
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



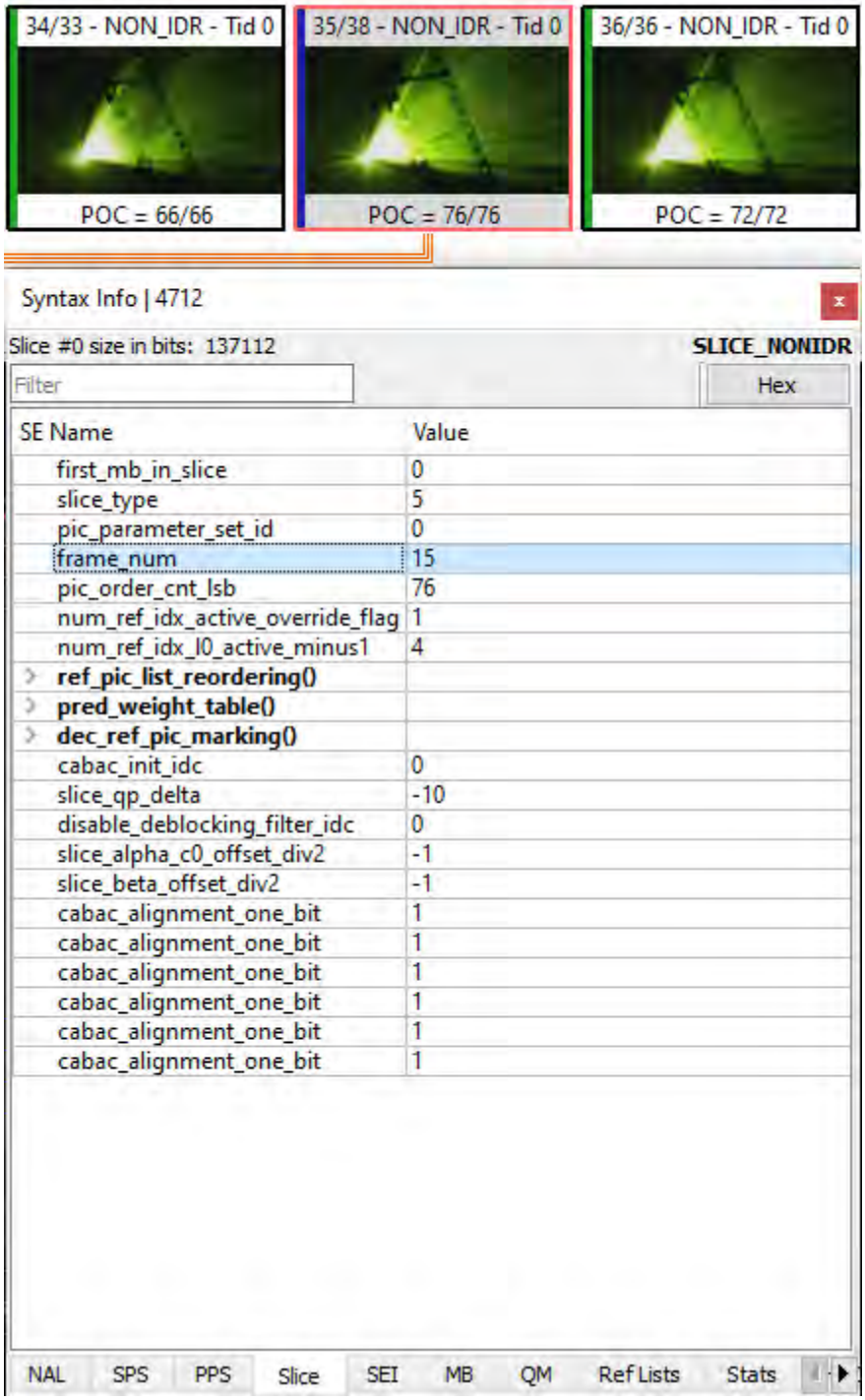
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



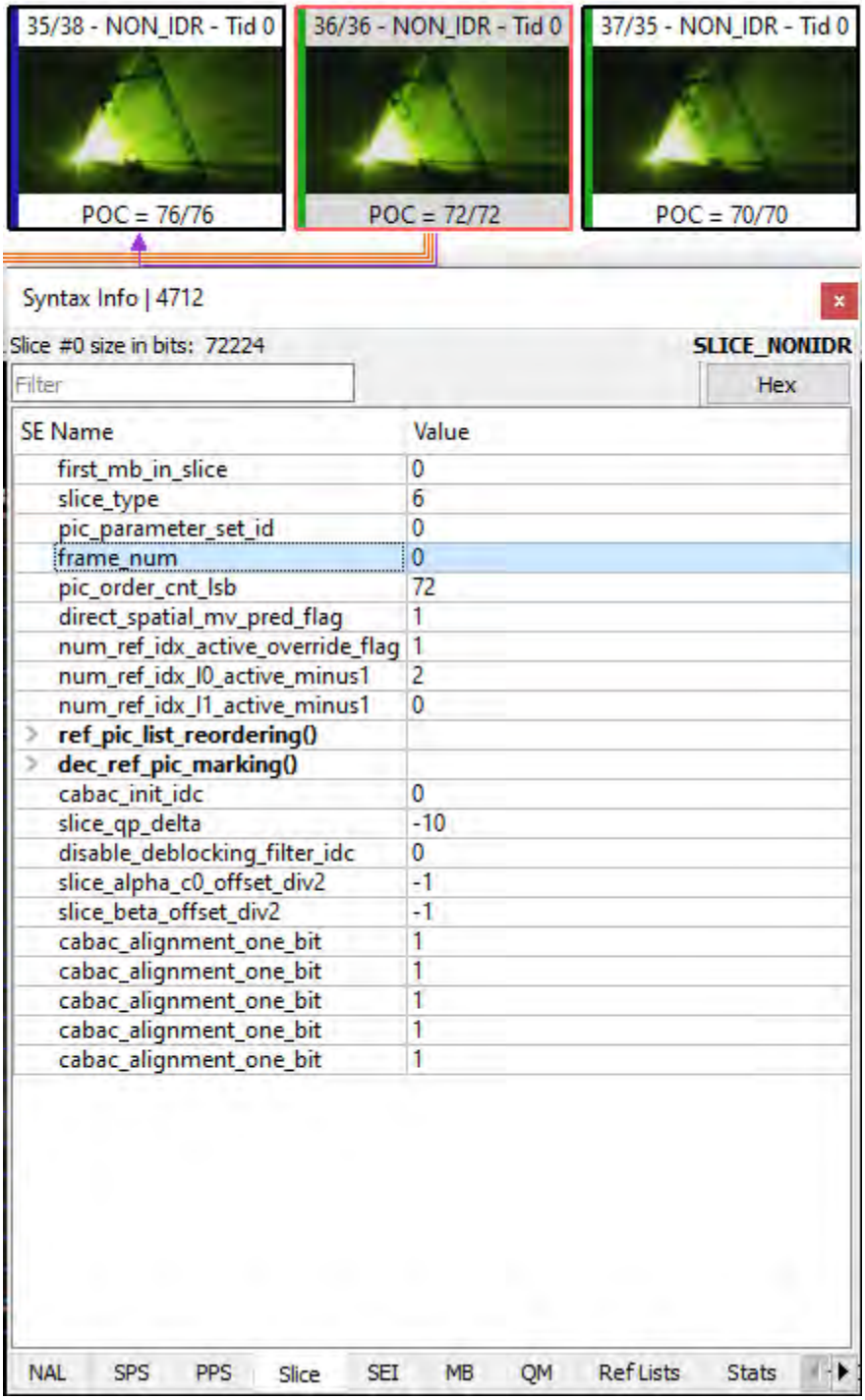
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



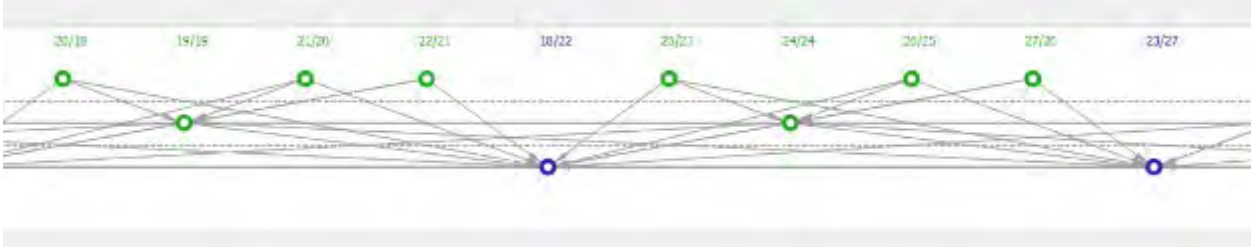
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



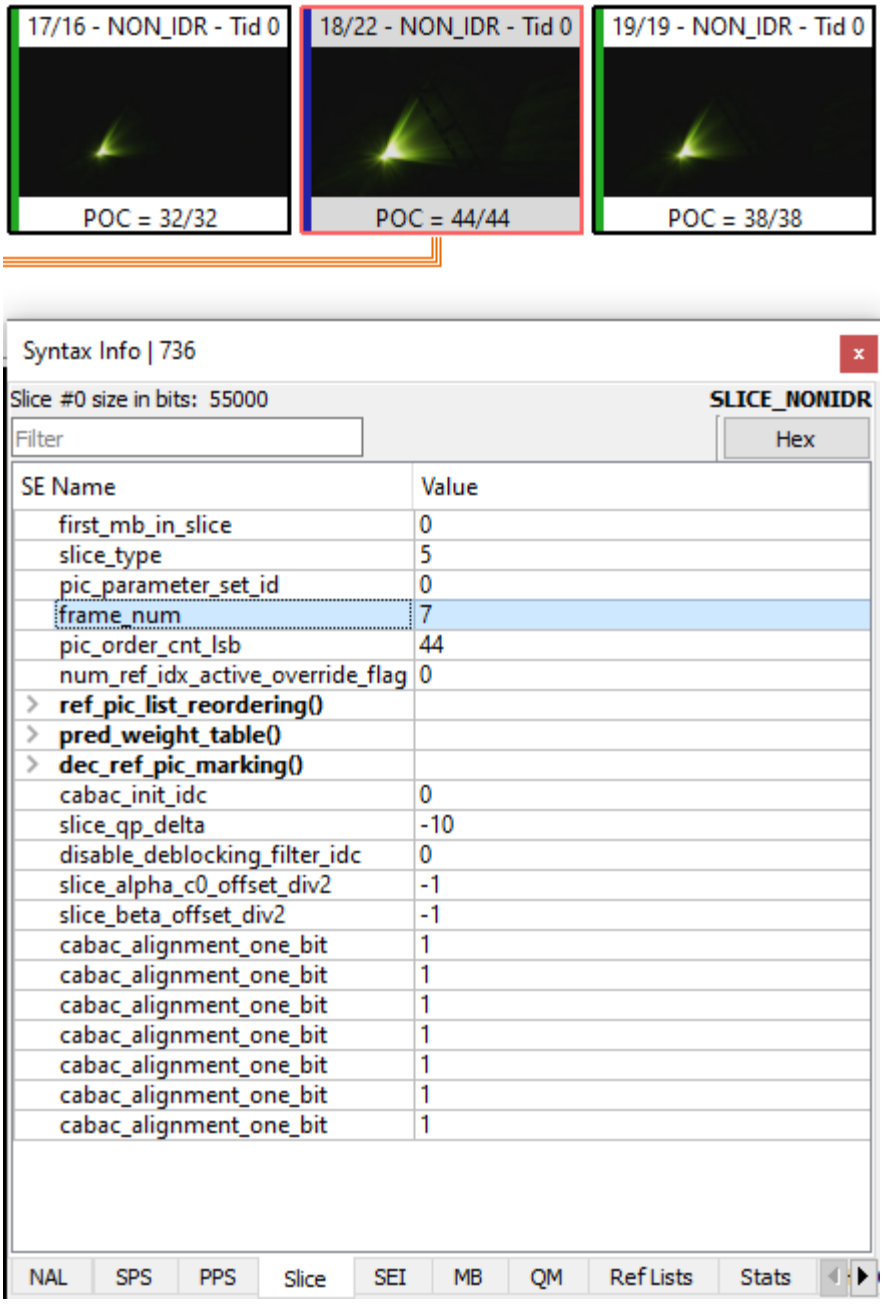
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

375. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs using an independent numbering scheme, to assign consecutive reference pictures in

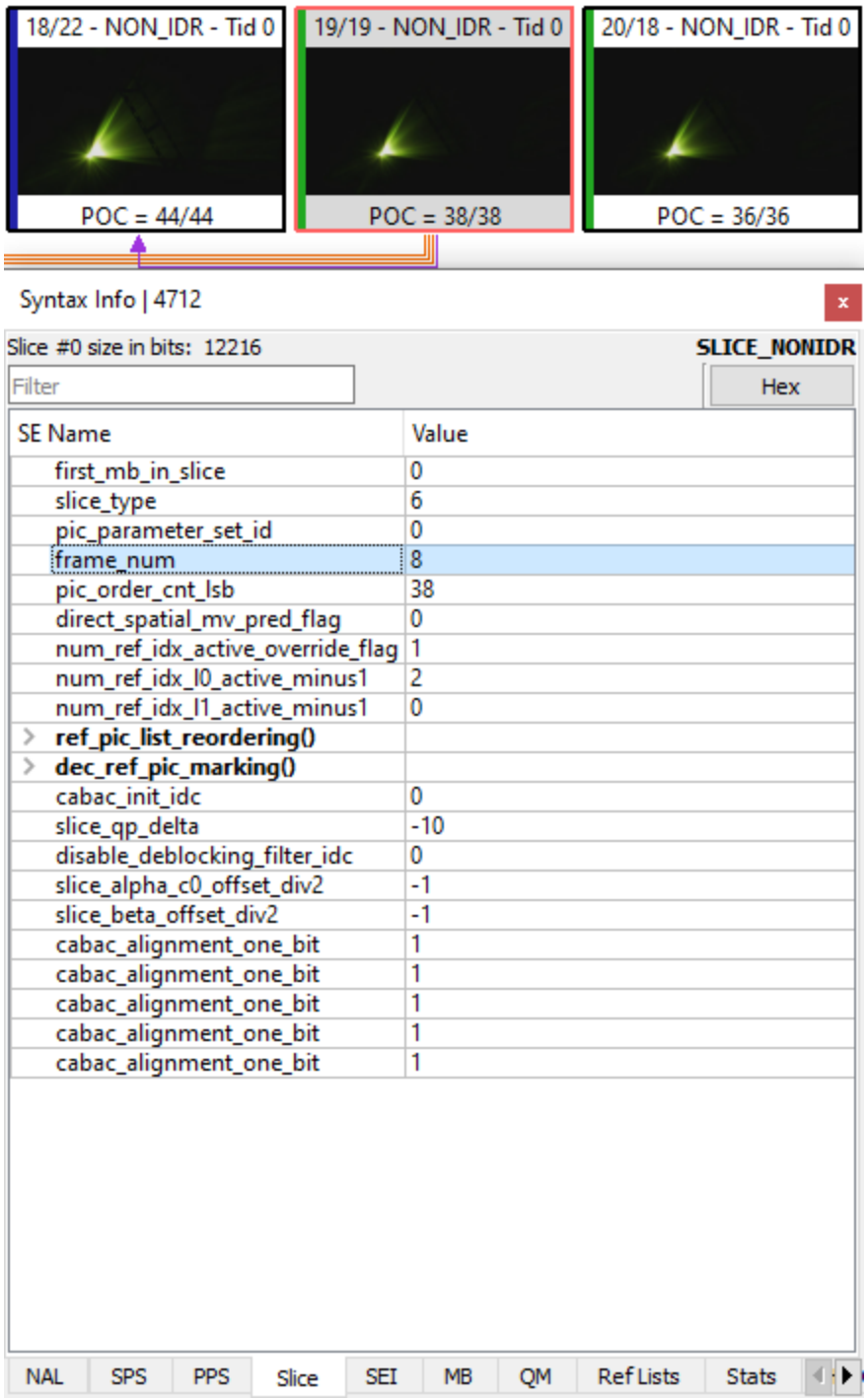
encoding order with respective sequence indicator values that differ with respect to each other by a predetermined amount independent of one or more of the number of non-reference pictures encoded between consecutive reference pictures and the number of non-coded pictures between consecutive reference pictures, as demonstrated in the screenshots below using VQ Analyzer software, where the frame number for reference pictures increases by 1 when there are three non-reference pictures between reference pictures.



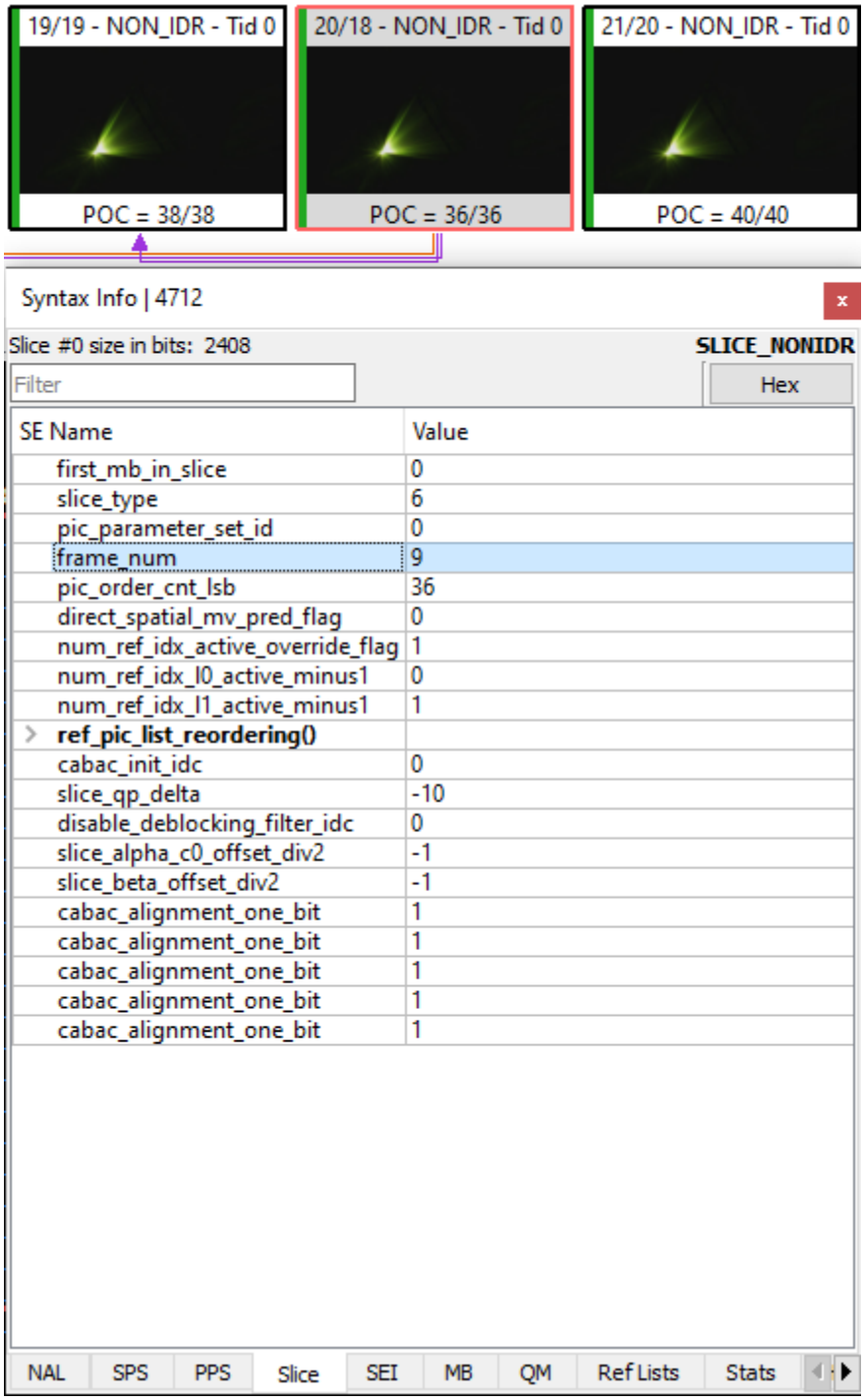
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



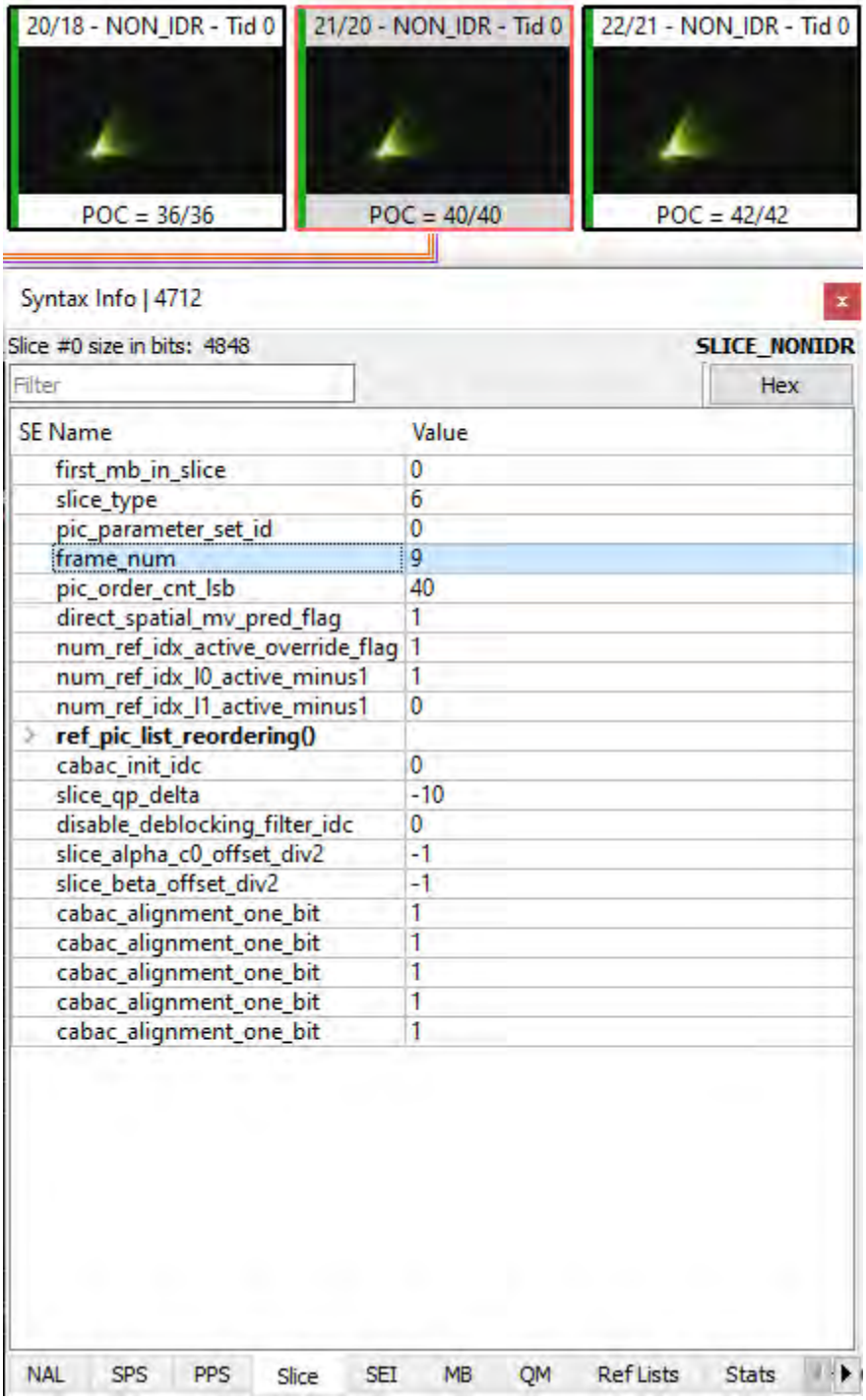
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



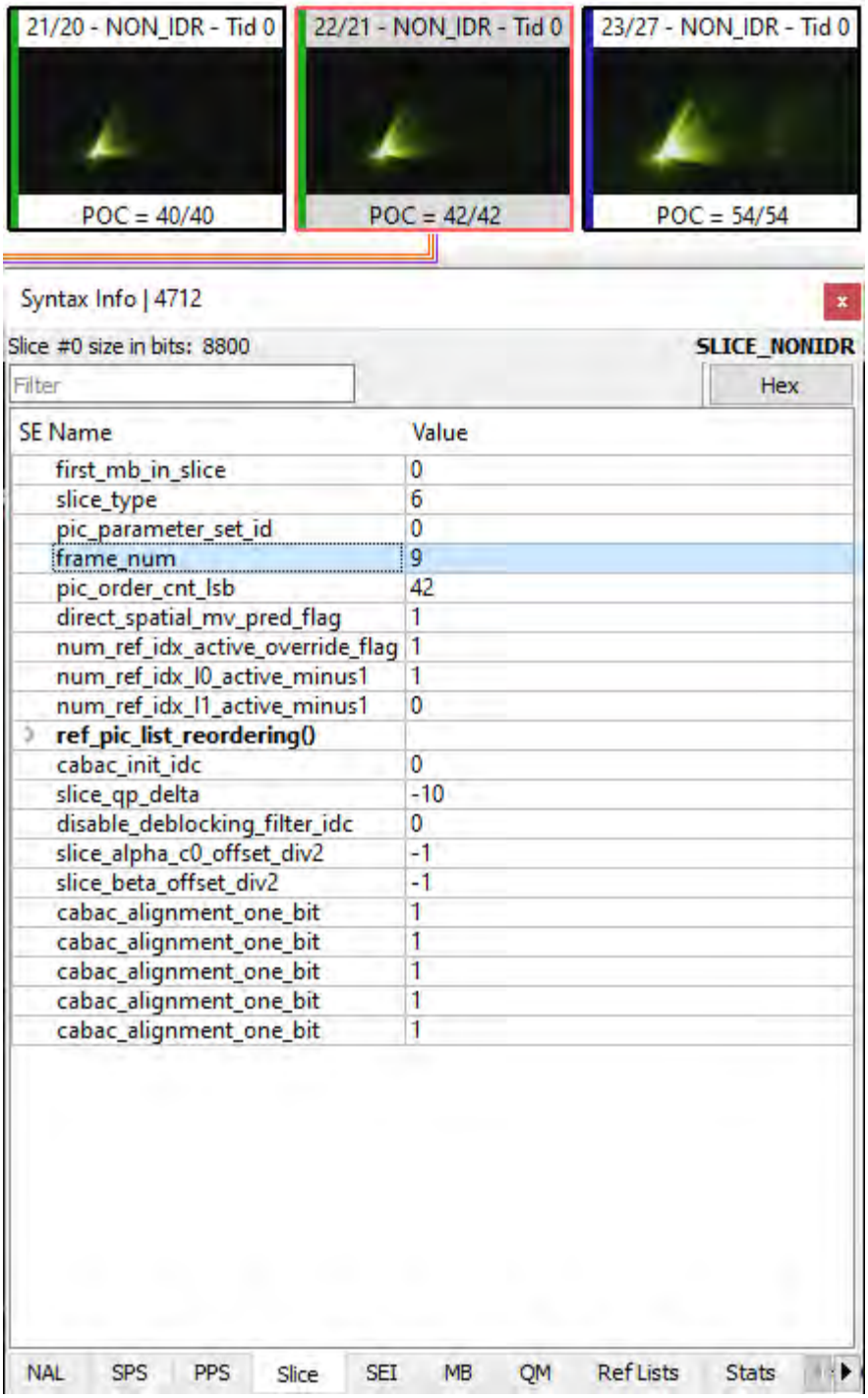
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



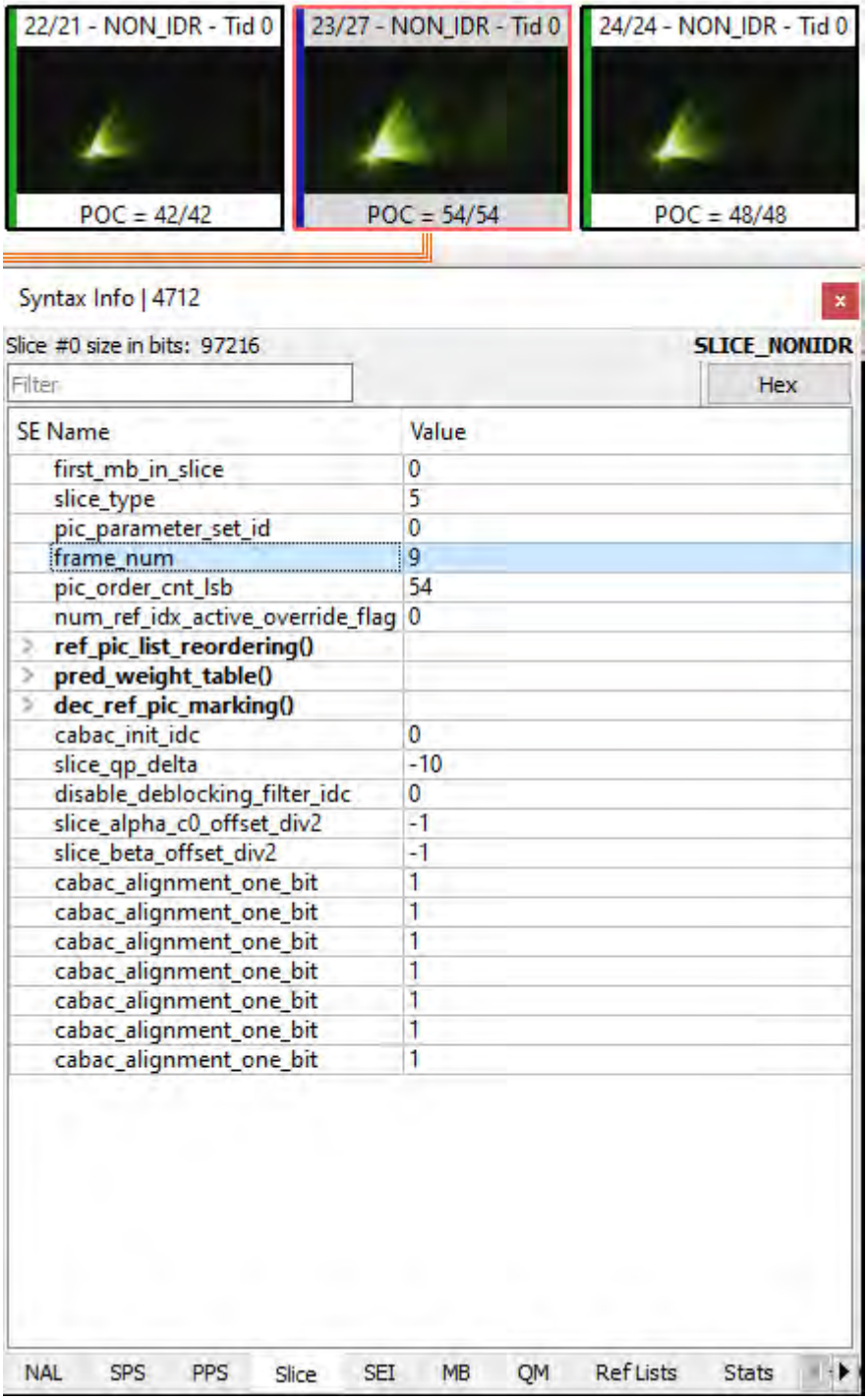
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



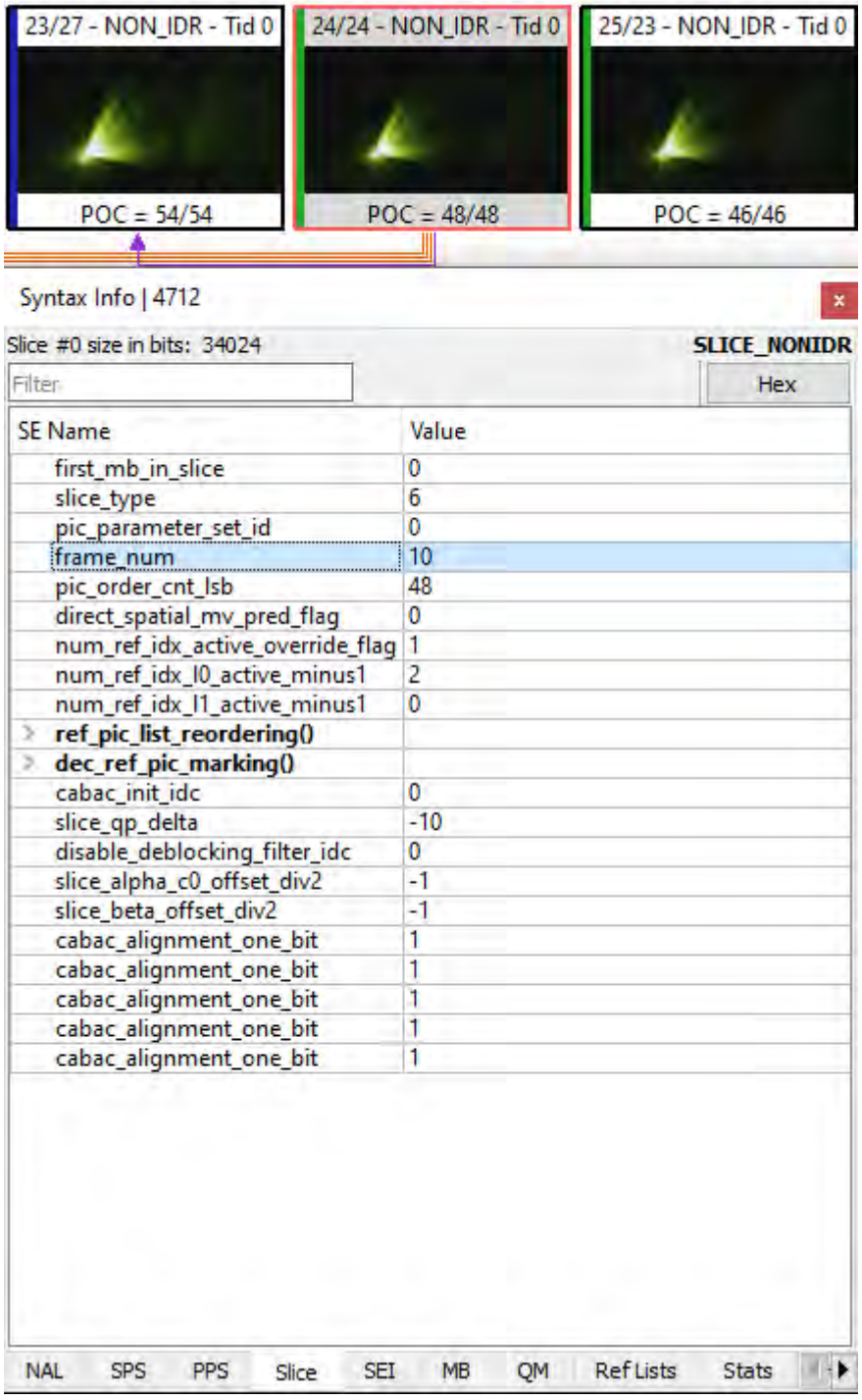
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

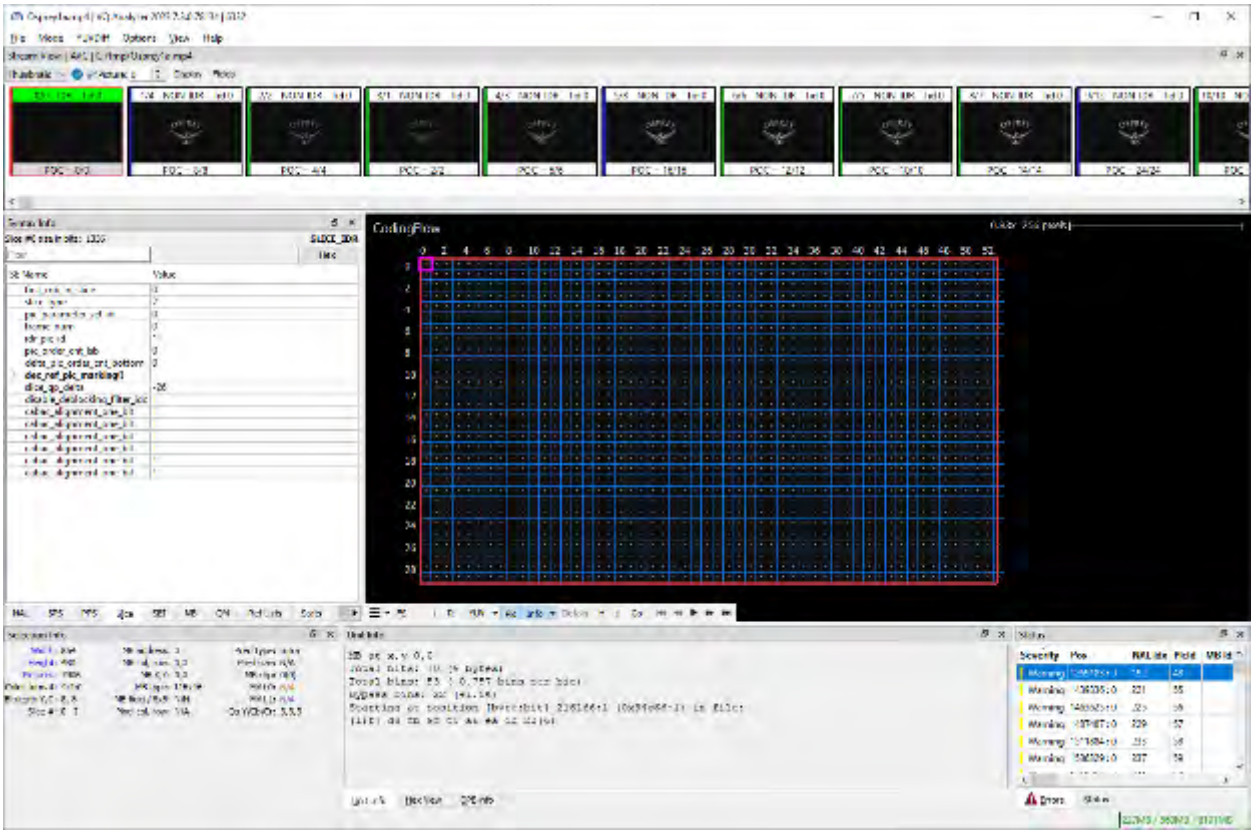


Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

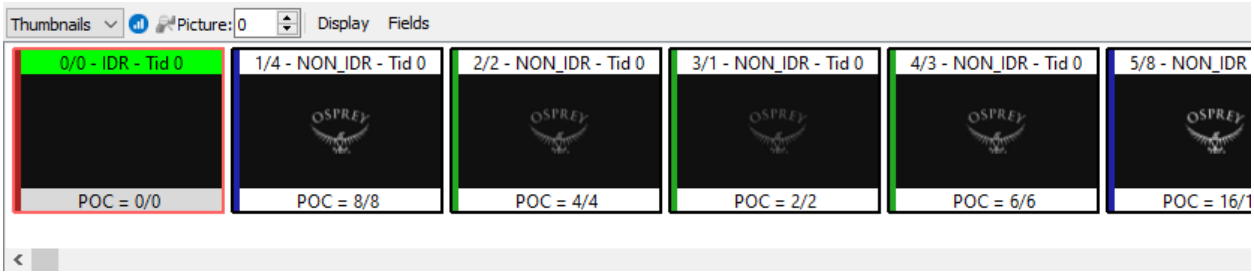
376. For another example, on information and belief, Amazon performs a method of encoding a video signal in a manner that is covered by claim 1 of the '764 Patent for Amazon.com

advertisements, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 217.

377. When encoding Amazon.com advertisements, on information and belief, Amazon performs encoding a video signal using an encoder to form an encoded video signal, the video signal representing a sequence of pictures, as demonstrated in the screenshots below using VQ Analyzer software.

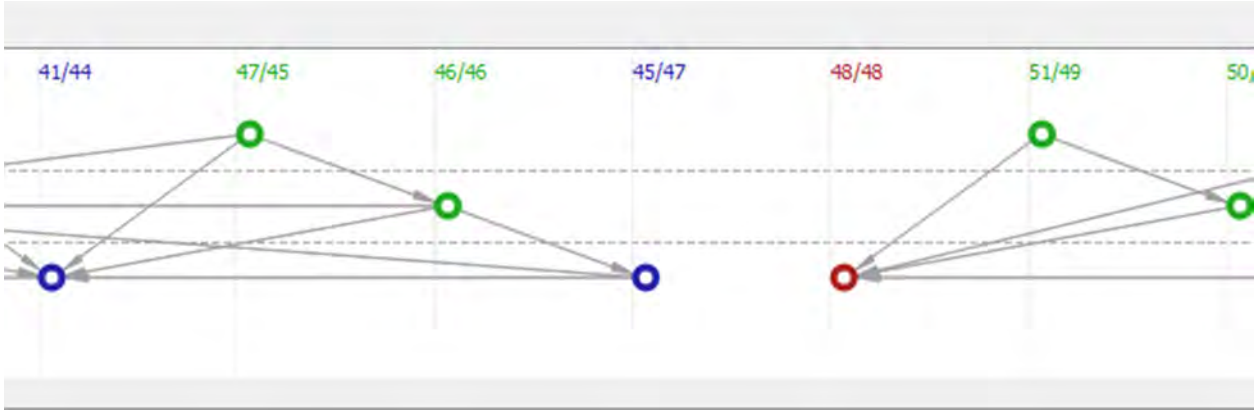


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

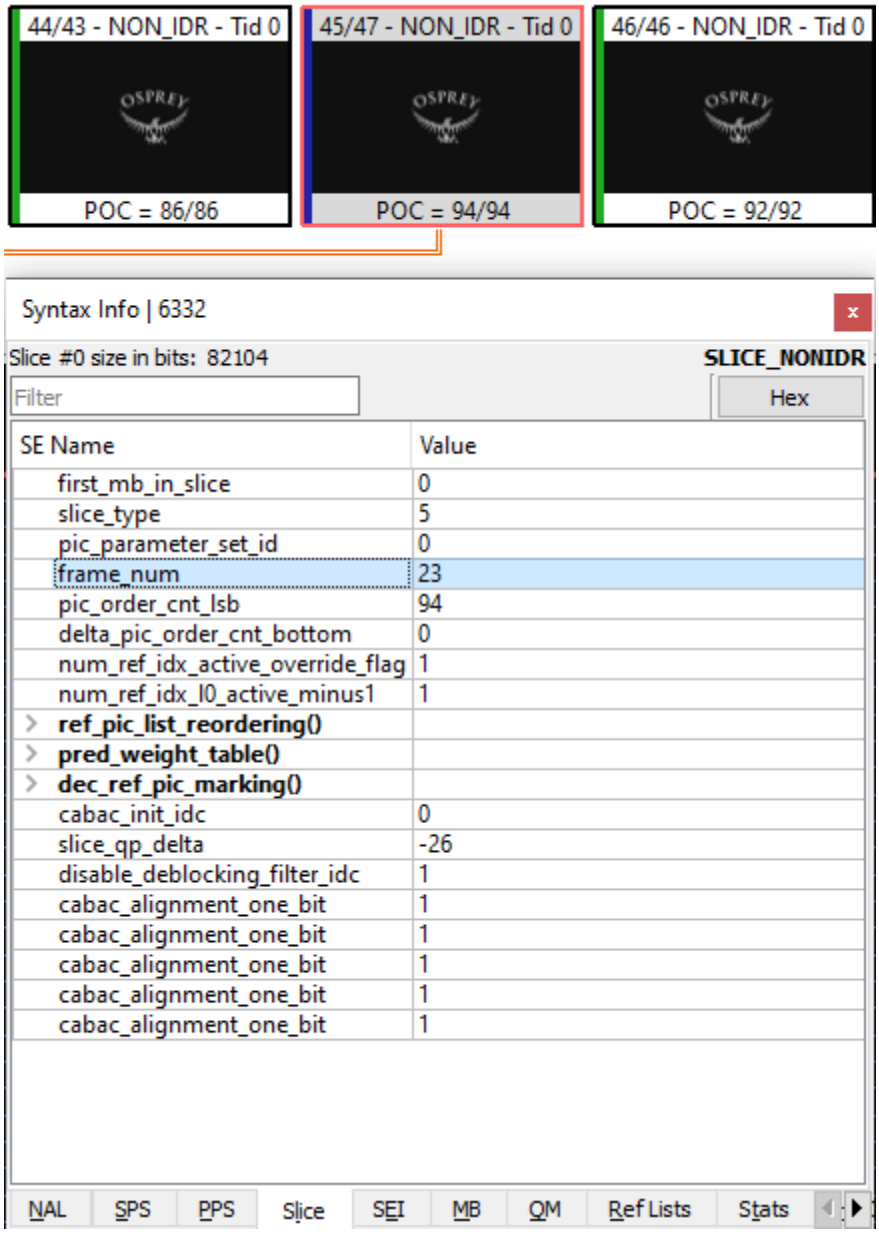


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

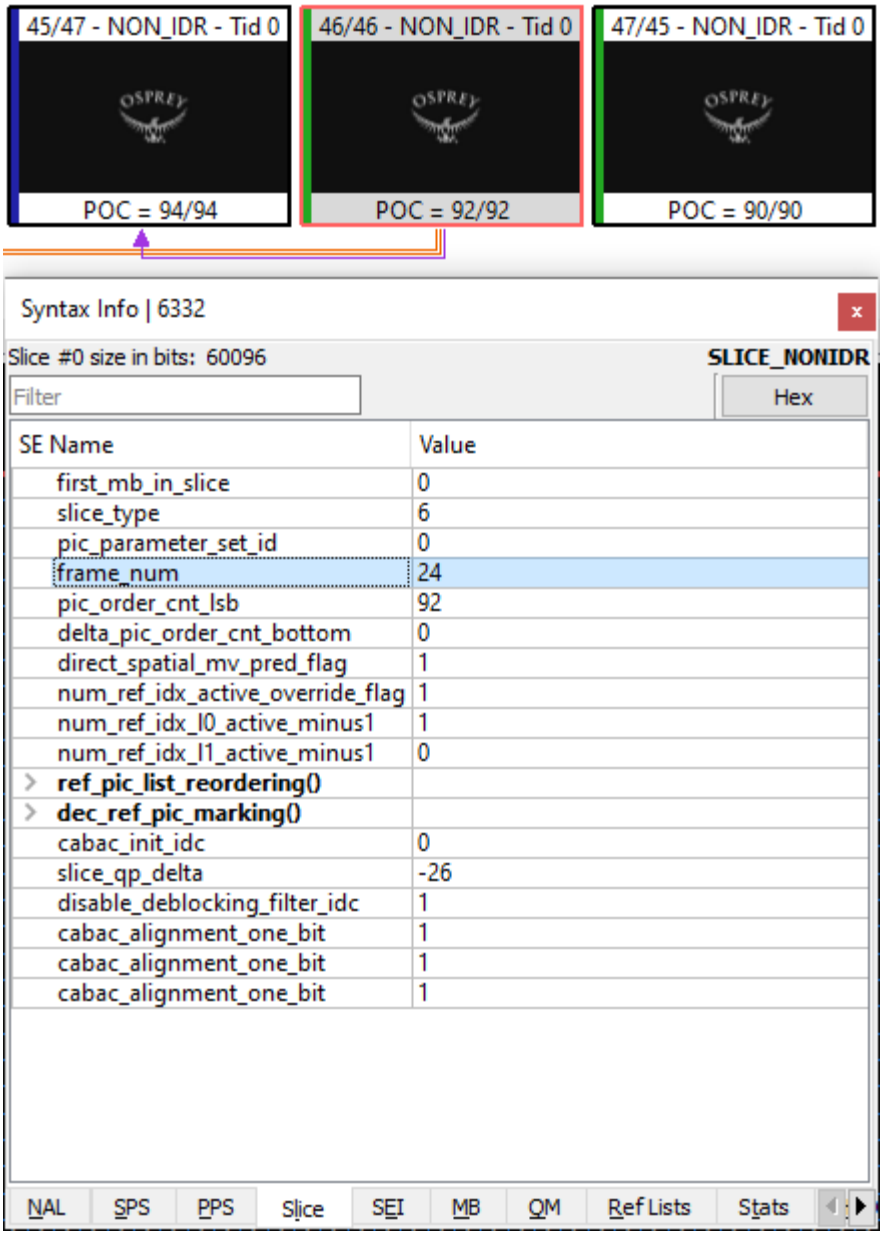
378. When encoding Amazon.com advertisements, on information and belief, Amazon performs using an independent numbering scheme, to assign consecutive reference pictures in encoding order with respective sequence indicator values that differ with respect to each other by a predetermined amount independent of one or more of the number of non-reference pictures encoded between consecutive reference pictures and the number of non-coded pictures between consecutive reference pictures, as demonstrated in the screenshots below using VQ Analyzer software, where the frame number for reference pictures increases by 1 when there is one non-reference picture between reference pictures.



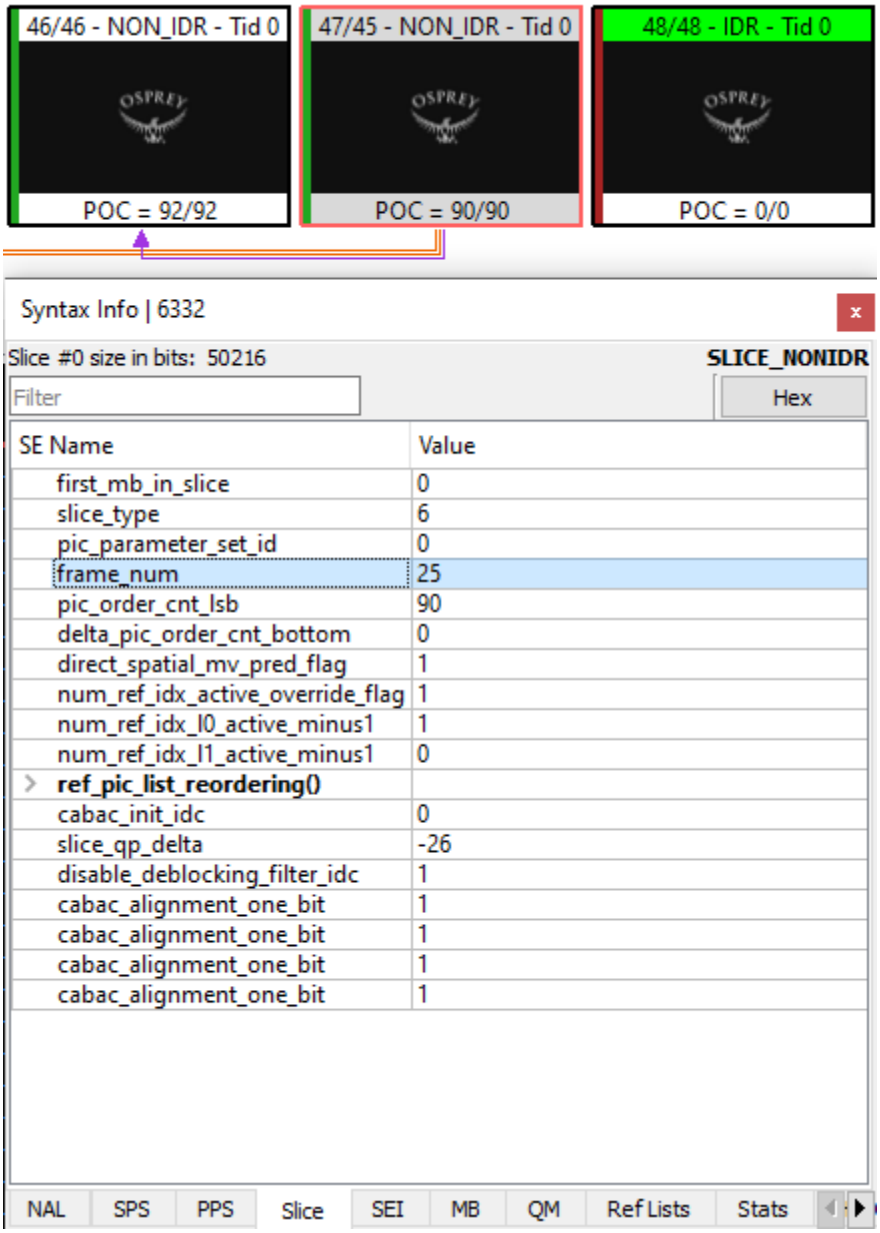
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



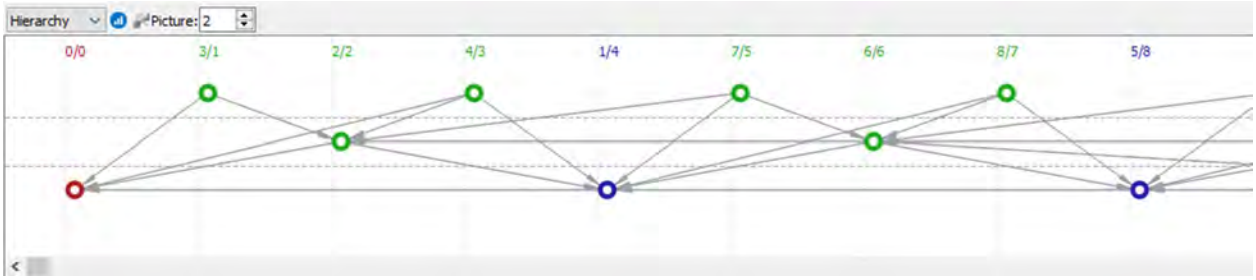
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



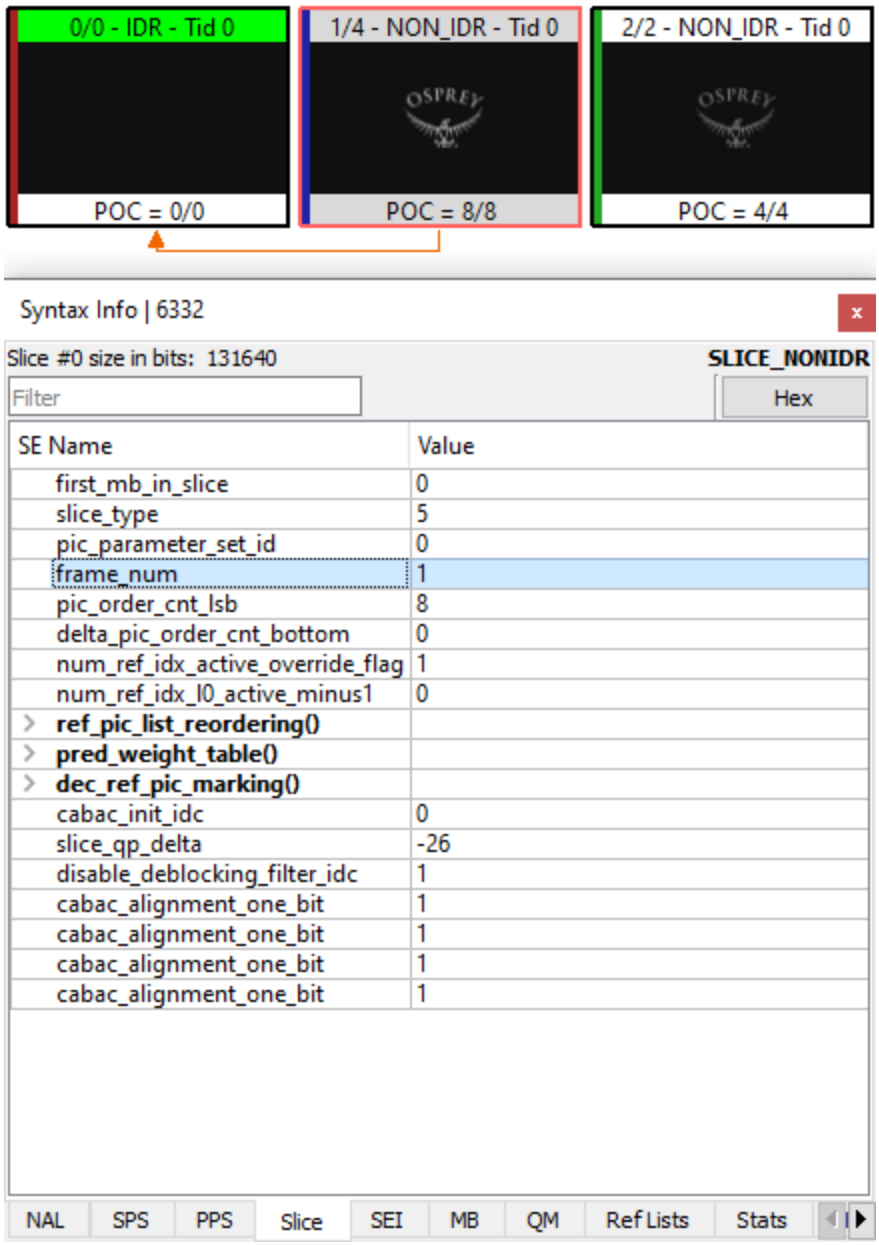
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

379. When encoding Amazon.com advertisements, on information and belief, Amazon performs using an independent numbering scheme, to assign consecutive reference pictures in encoding order with respective sequence indicator values that differ with respect to each other by a predetermined amount independent of one or more of the number of non-reference pictures encoded between consecutive reference pictures and the number of non-coded pictures between

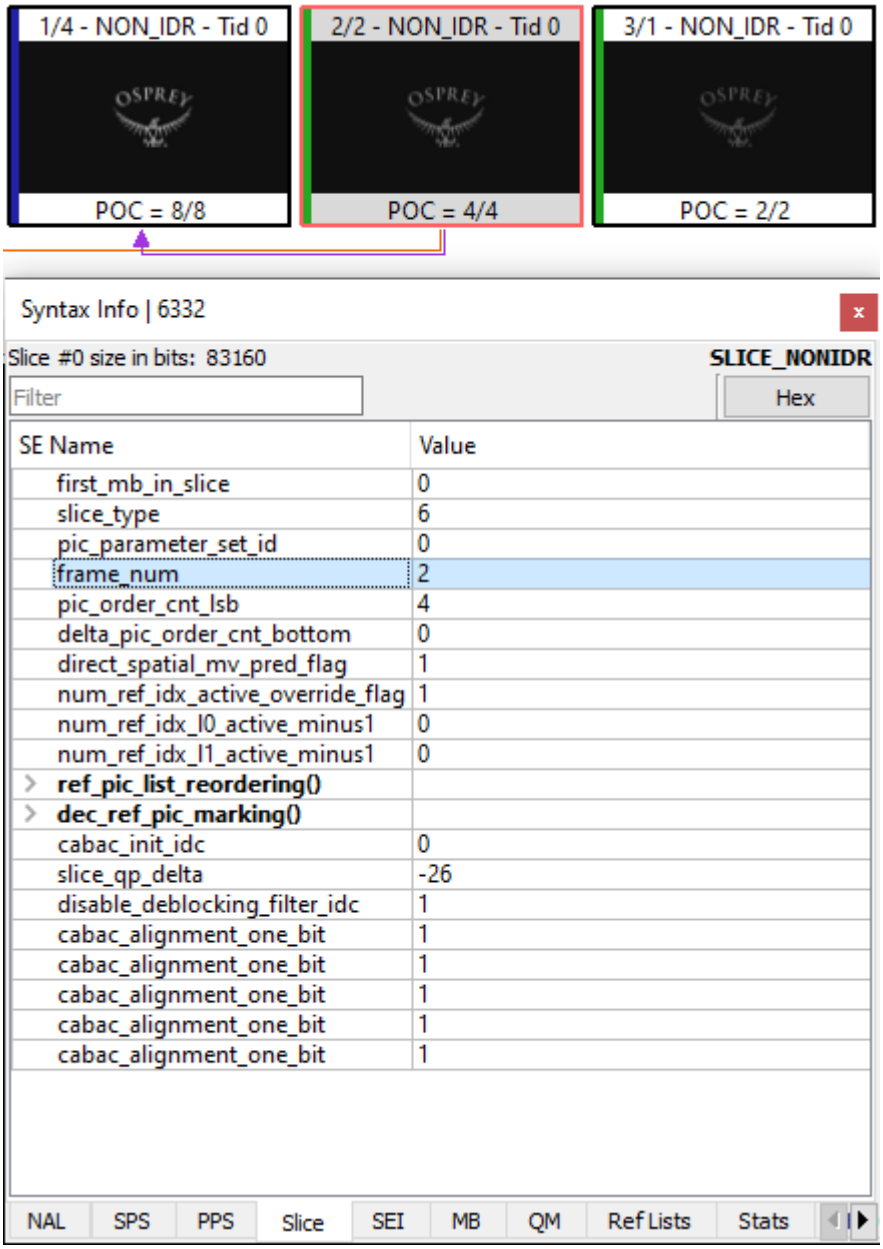
consecutive reference pictures, as demonstrated in the screenshots below using VQ Analyzer software, where the frame number for reference pictures increases by 1 when there are two non-reference pictures between reference pictures.



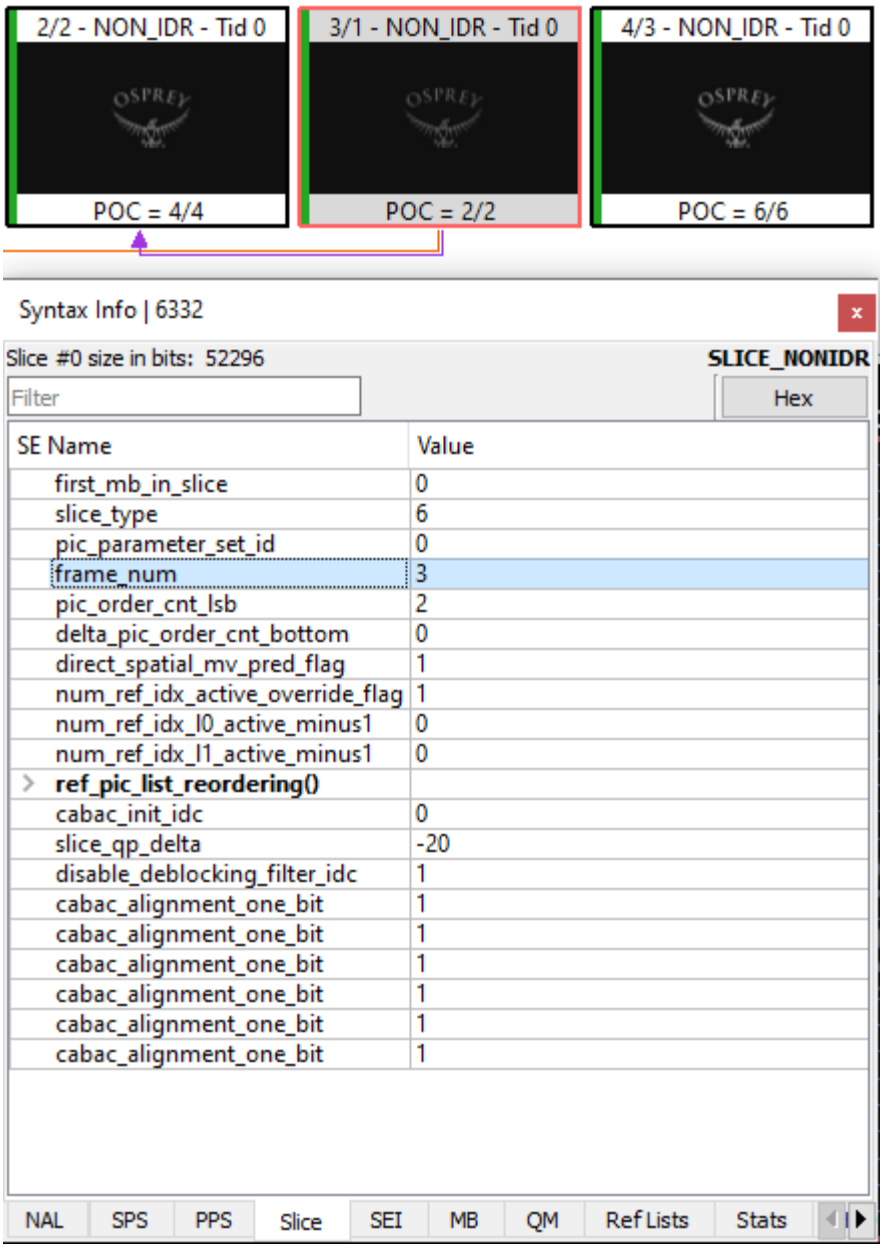
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



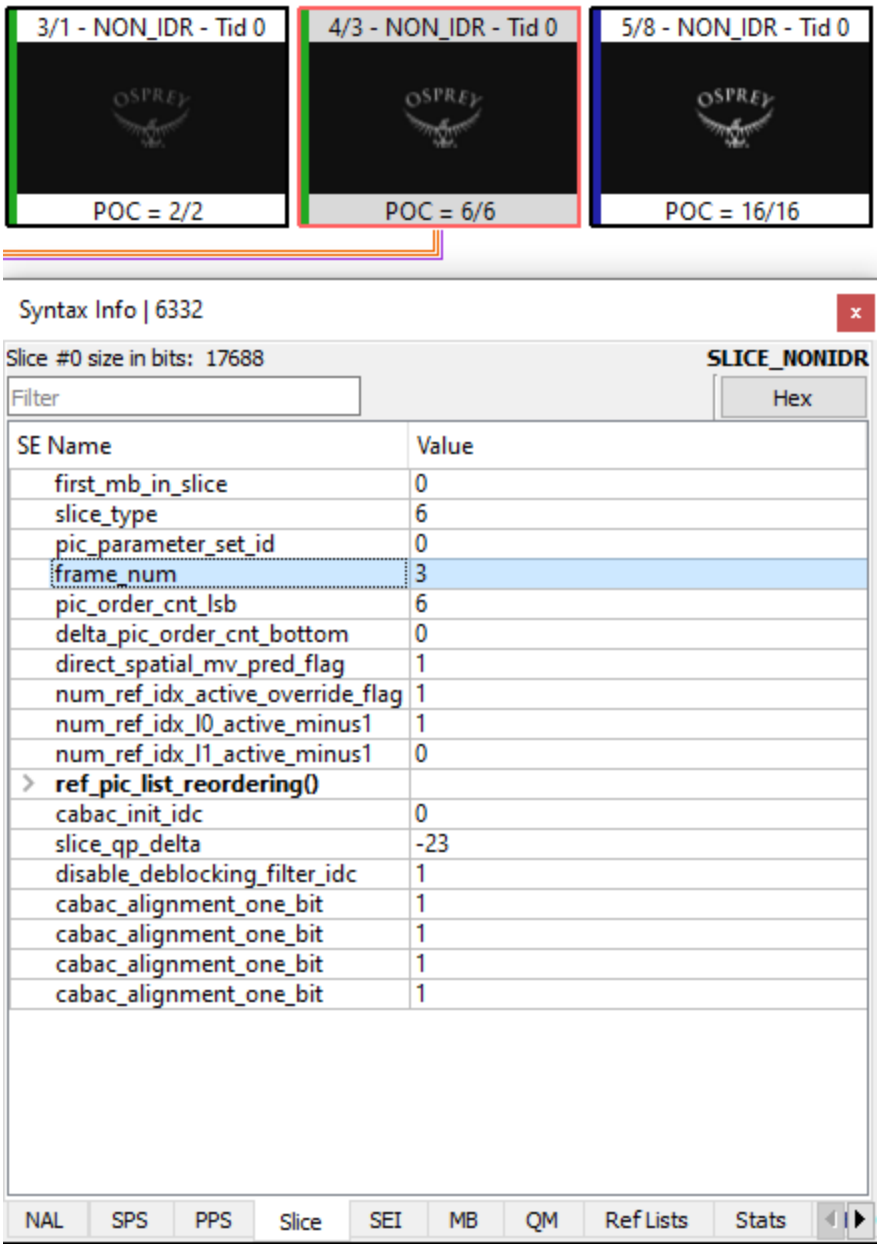
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



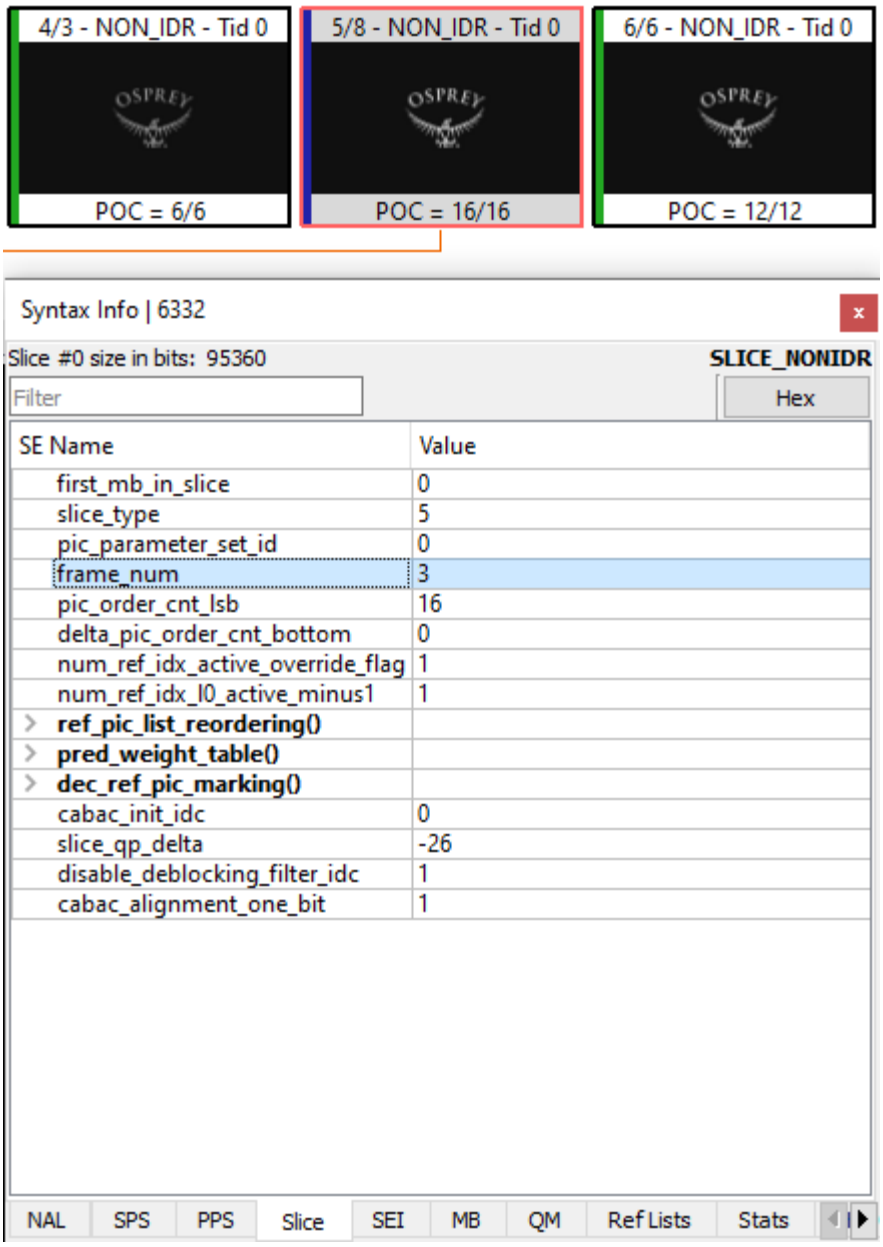
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



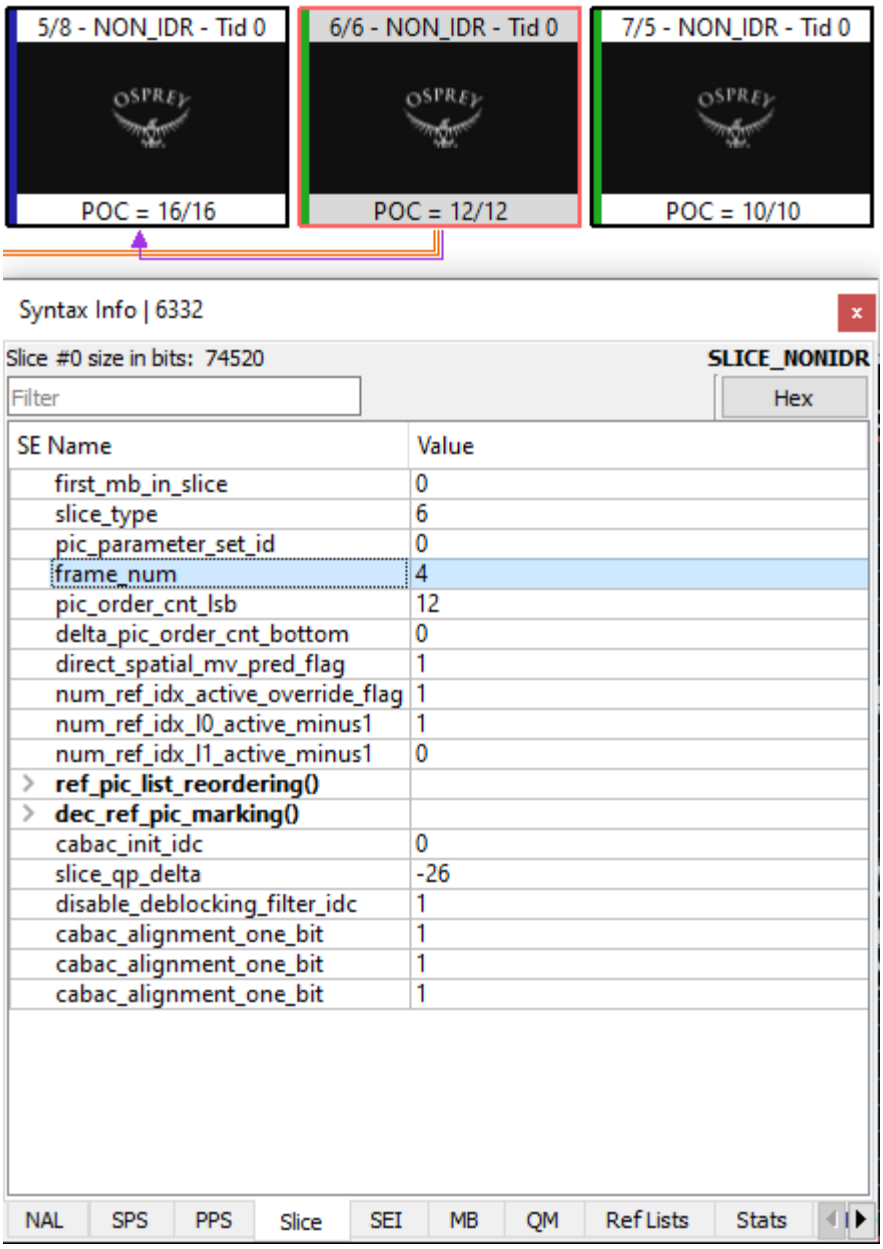
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



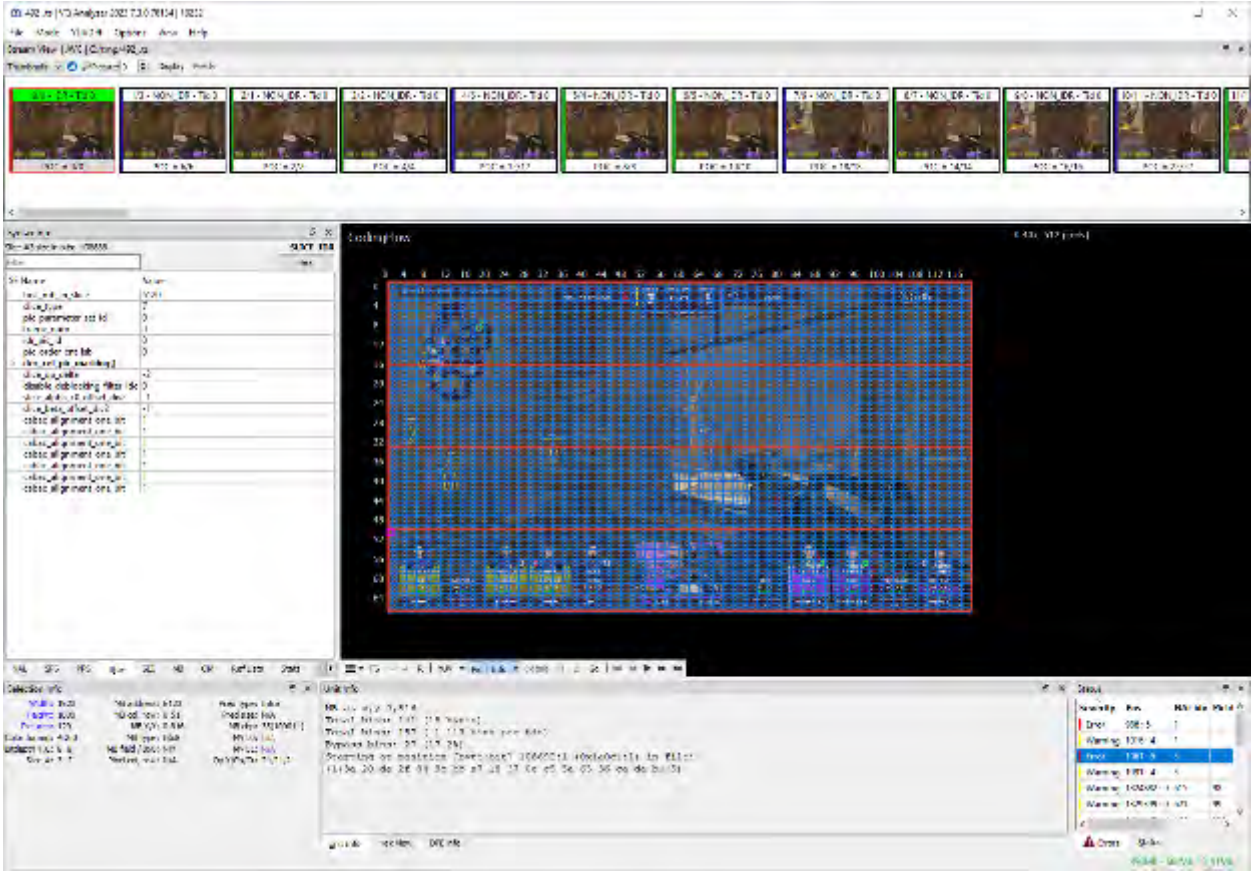
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



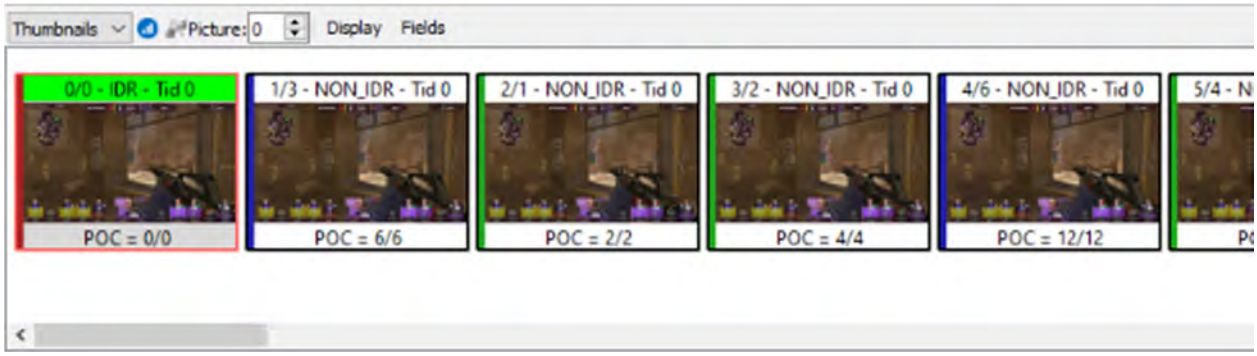
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

380. For another example, on information and belief, Amazon performs a method of encoding a video signal in a manner that is covered by claim 1 of the ’764 Patent for Twitch.tv, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 222.

381. When encoding Twitch.tv video, on information and belief, Amazon performs encoding a video signal using an encoder to form an encoded video signal, the video signal representing a sequence of pictures, as demonstrated in the screenshots below using VQ Analyzer software.

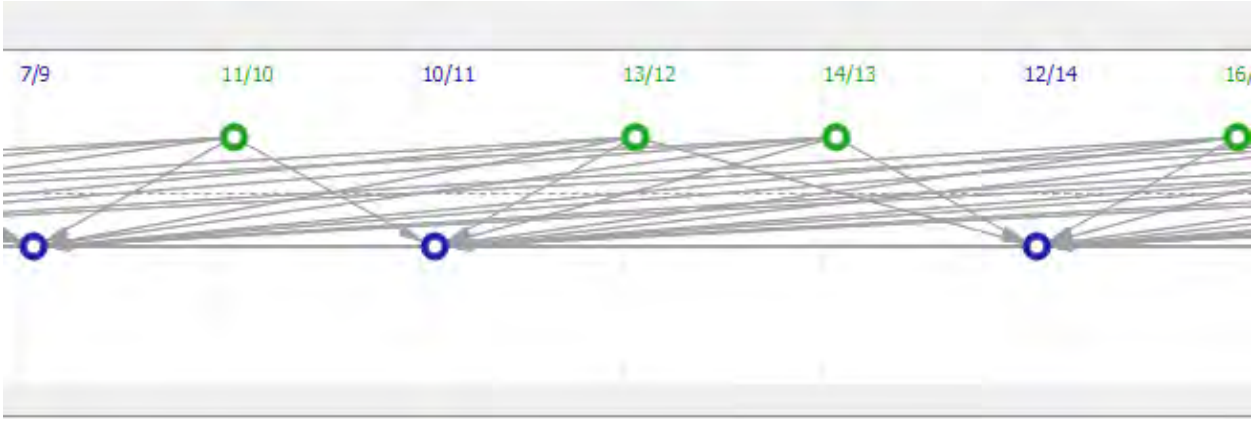


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

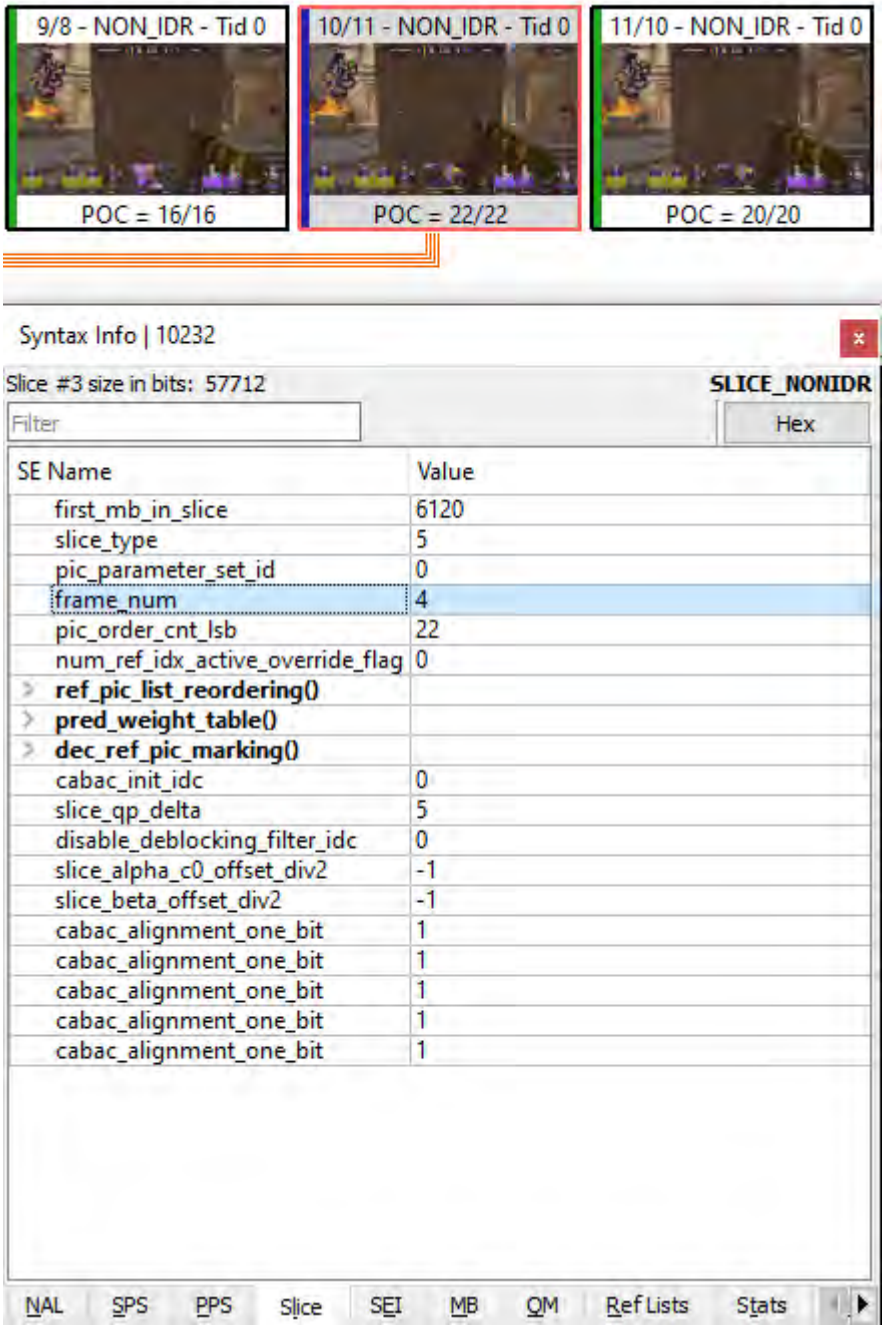


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

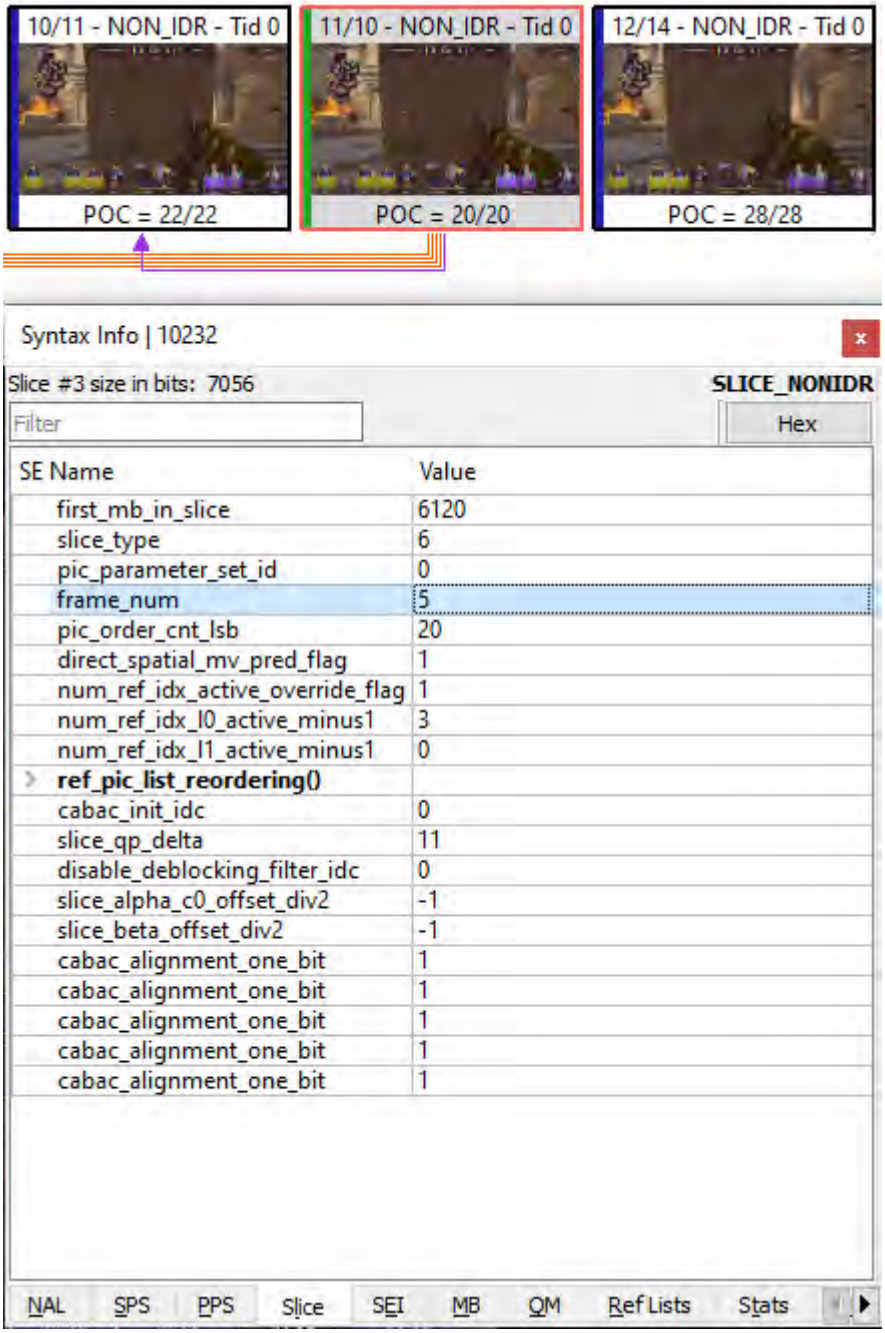
382. When encoding Twitch.tv video, on information and belief, Amazon performs using an independent numbering scheme, to assign consecutive reference pictures in encoding order with respective sequence indicator values that differ with respect to each other by a predetermined amount independent of one or more of the number of non-reference pictures encoded between consecutive reference pictures and the number of non-coded pictures between consecutive reference pictures, as demonstrated in the screenshots below using VQ Analyzer software, where the frame number for reference pictures increases by 1 when there is one non-reference picture between reference pictures.



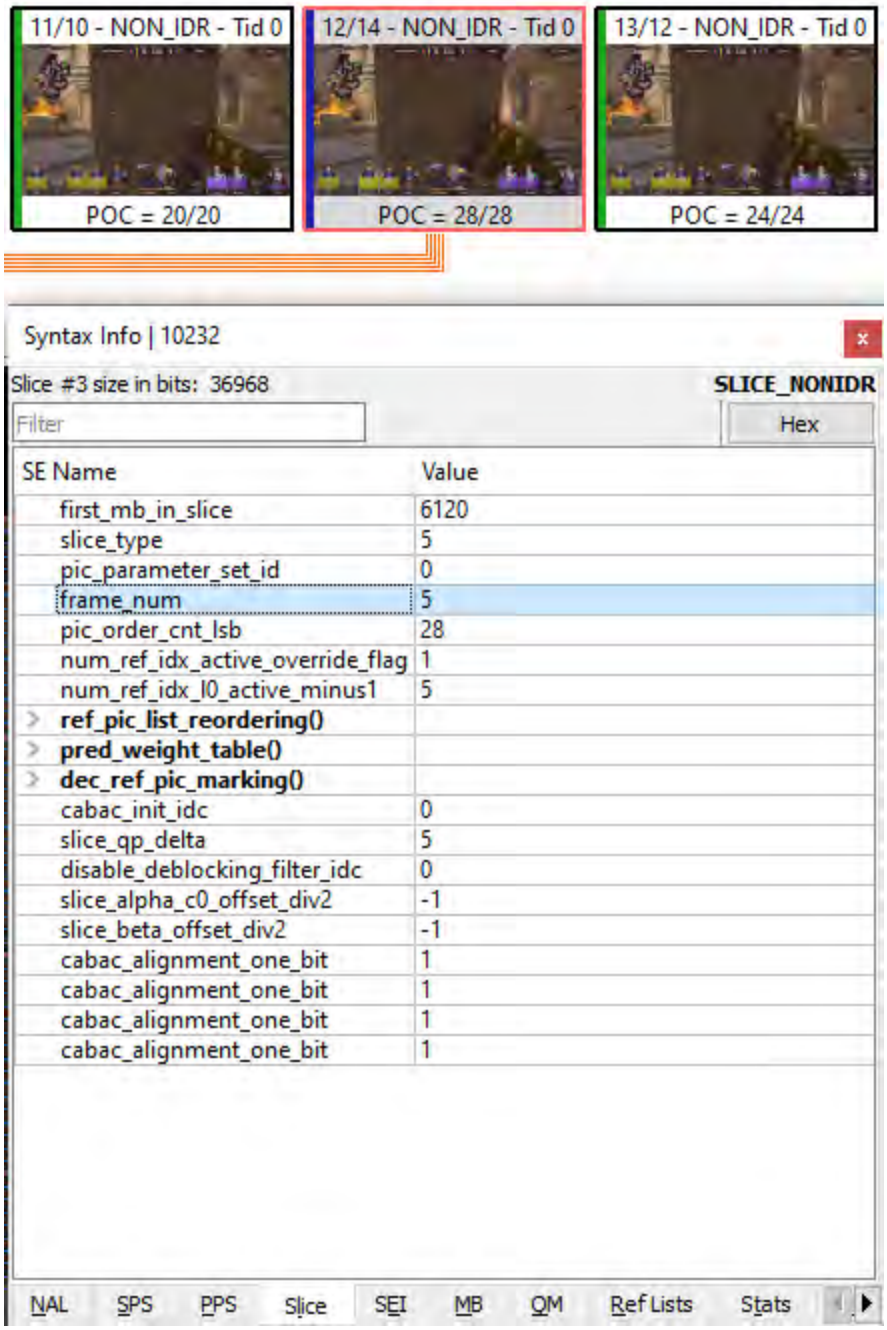
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



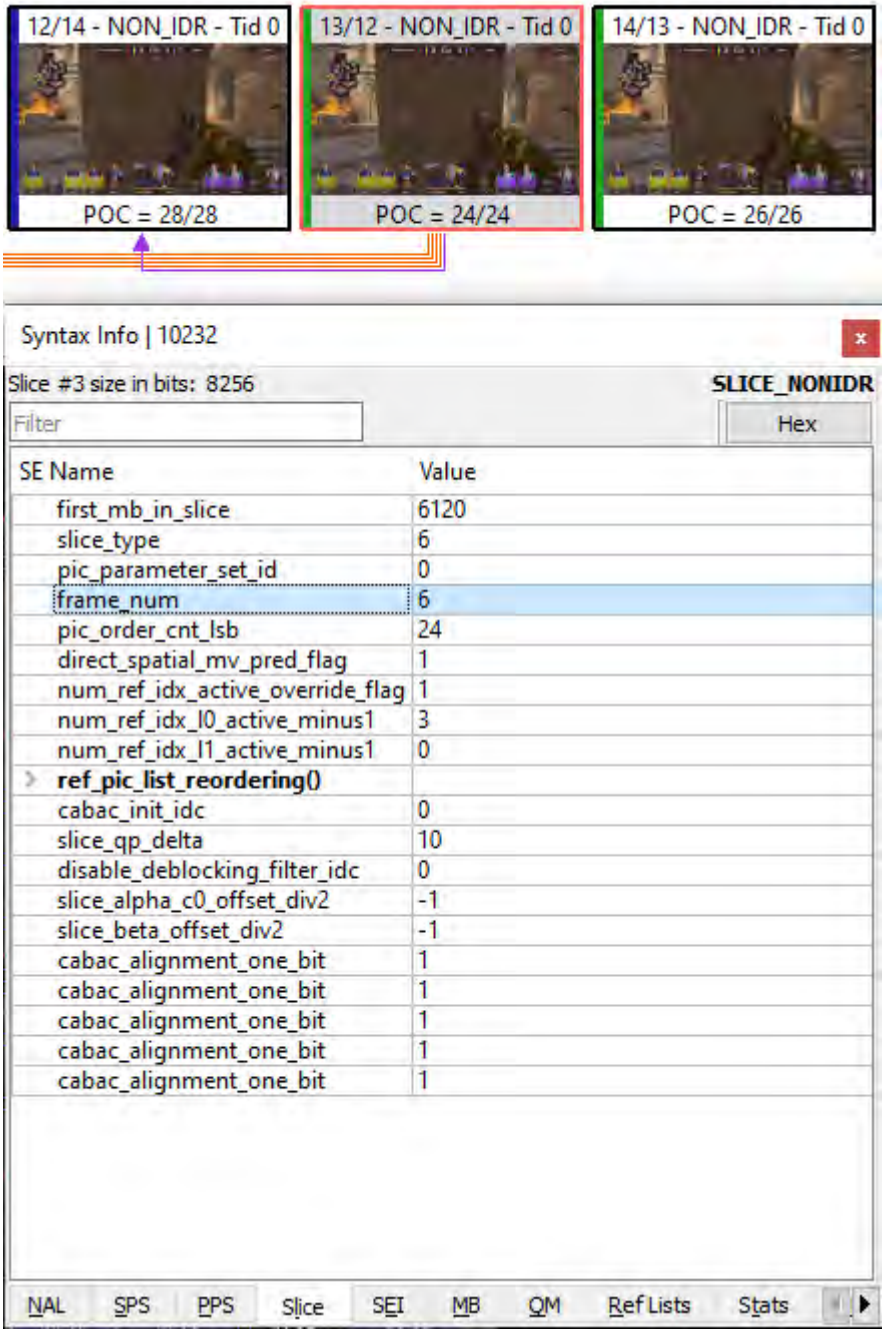
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



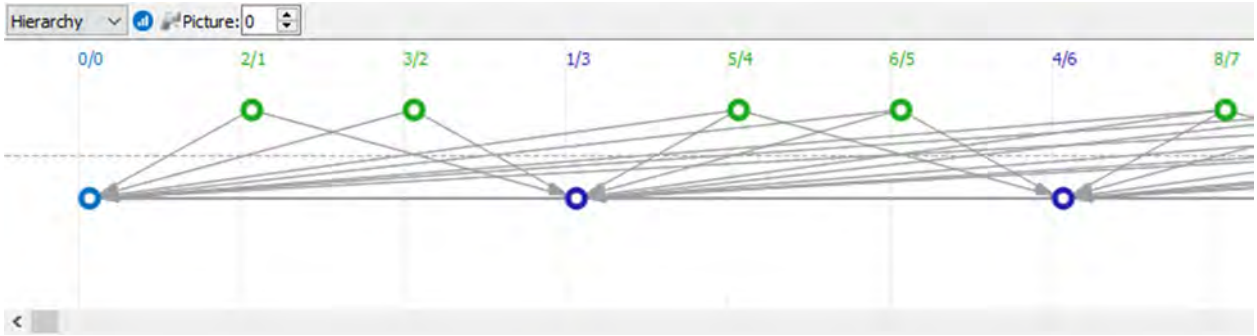
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



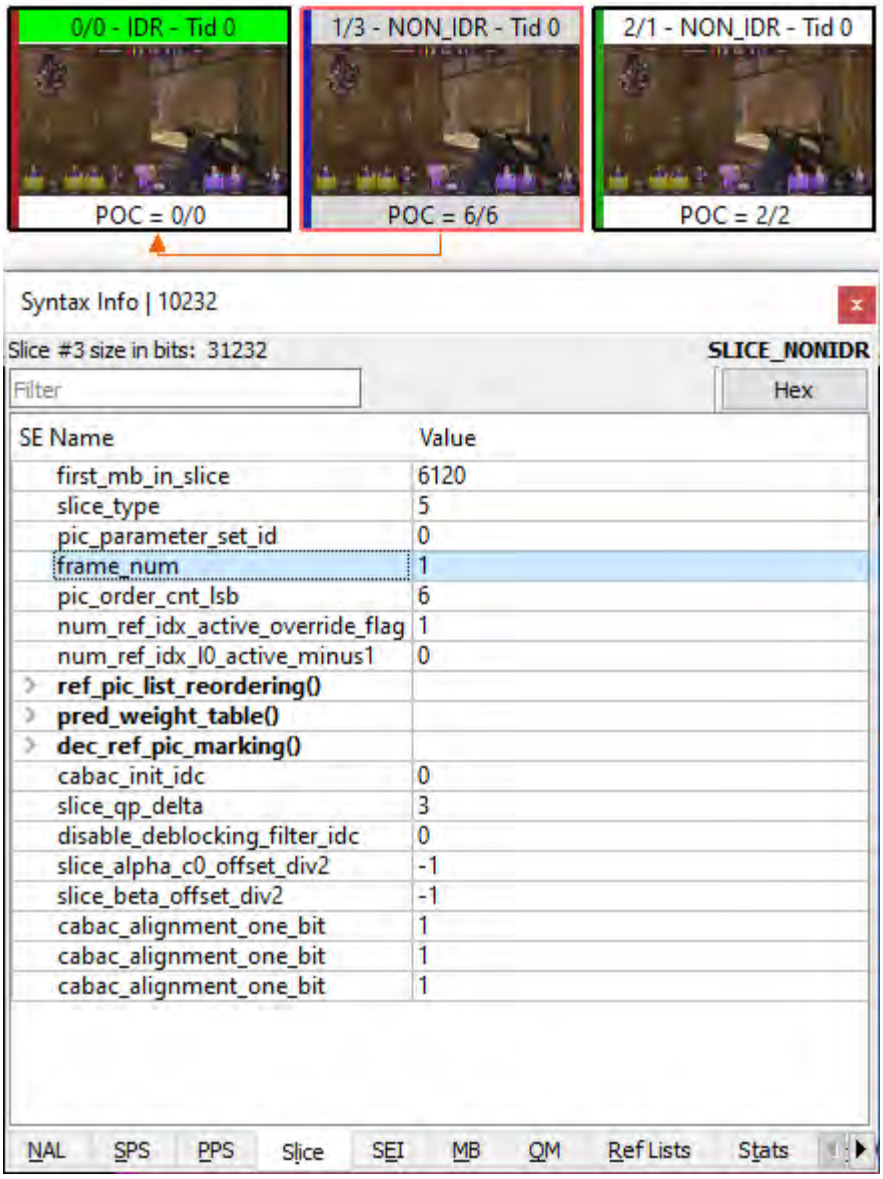
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

383. When encoding Twitch.tv video, on information and belief, Amazon performs using an independent numbering scheme, to assign consecutive reference pictures in encoding order with respective sequence indicator values that differ with respect to each other by a predetermined amount independent of one or more of the number of non-reference pictures

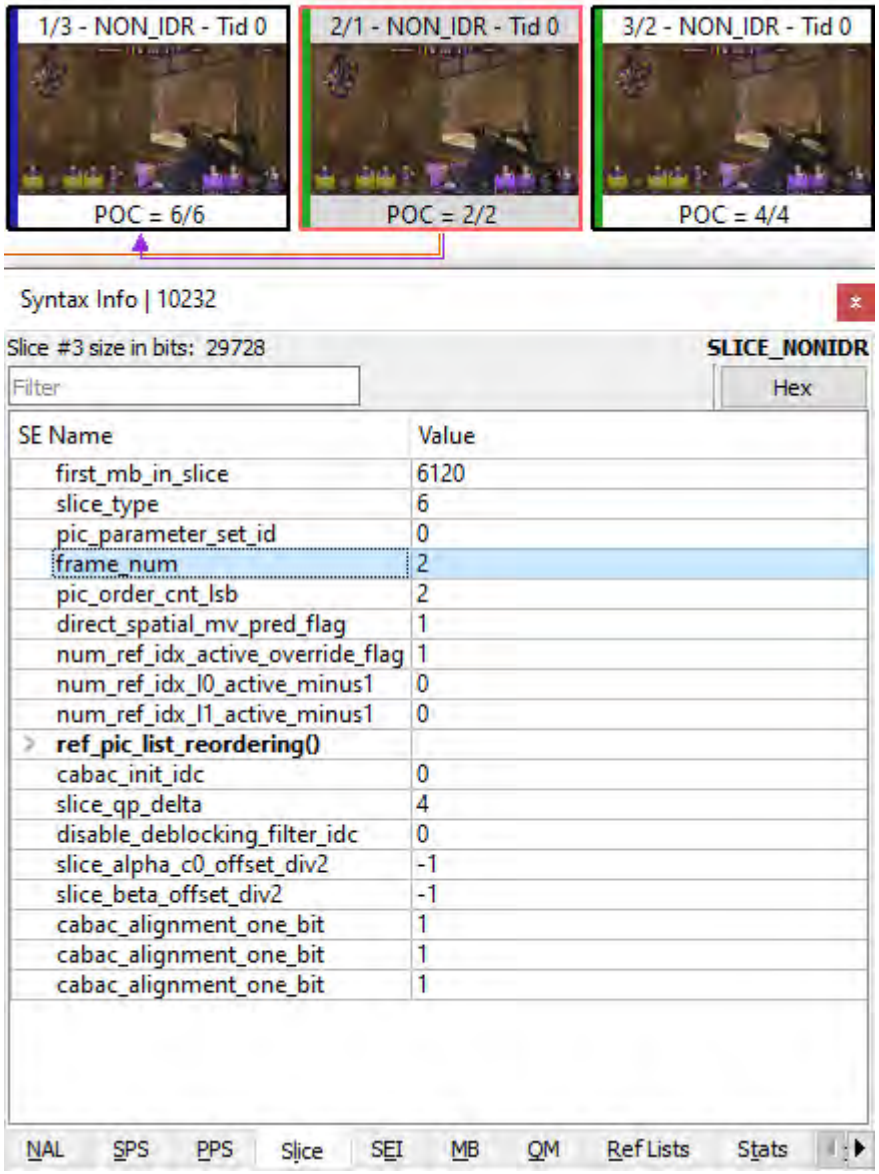
encoded between consecutive reference pictures and the number of non-coded pictures between consecutive reference pictures, as demonstrated in the screenshots below using VQ Analyzer software, where the frame number for reference pictures increases by 1 when there are two non-reference pictures between reference pictures.



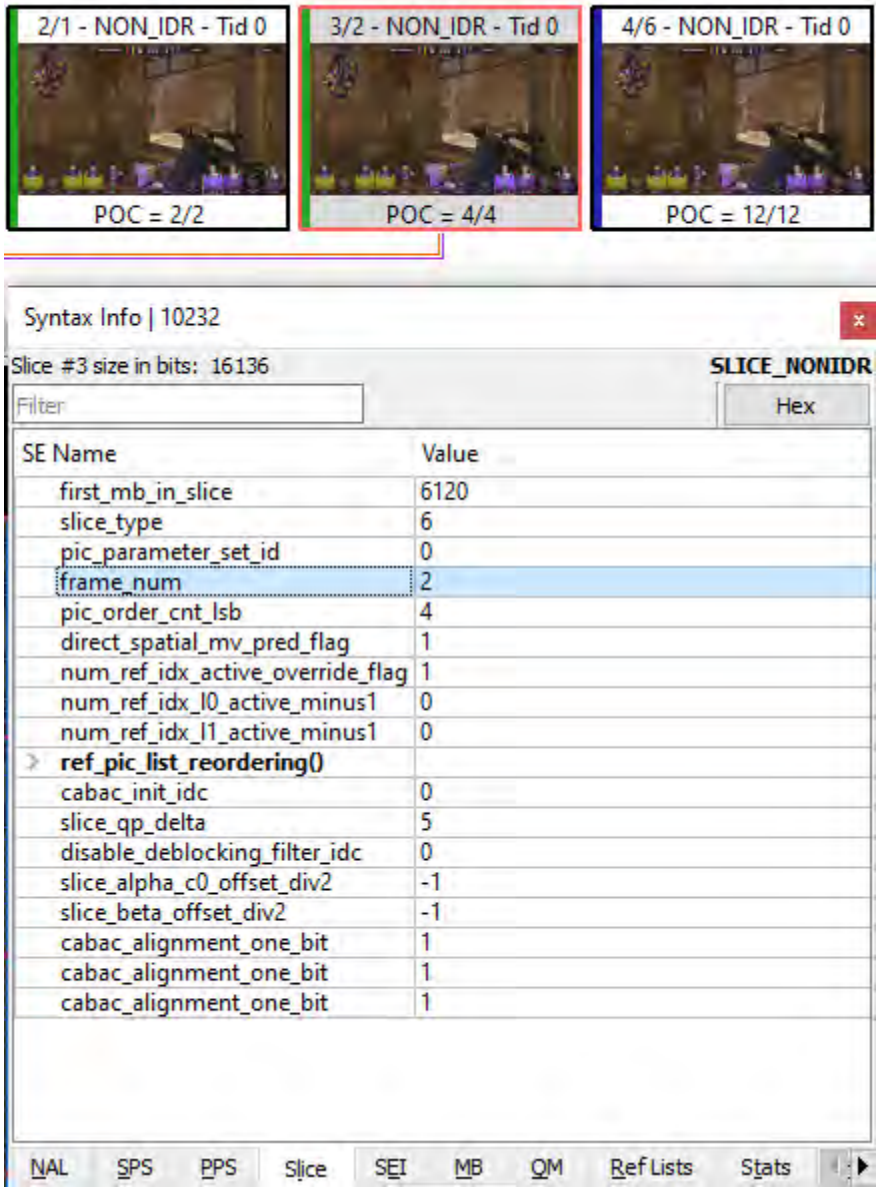
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



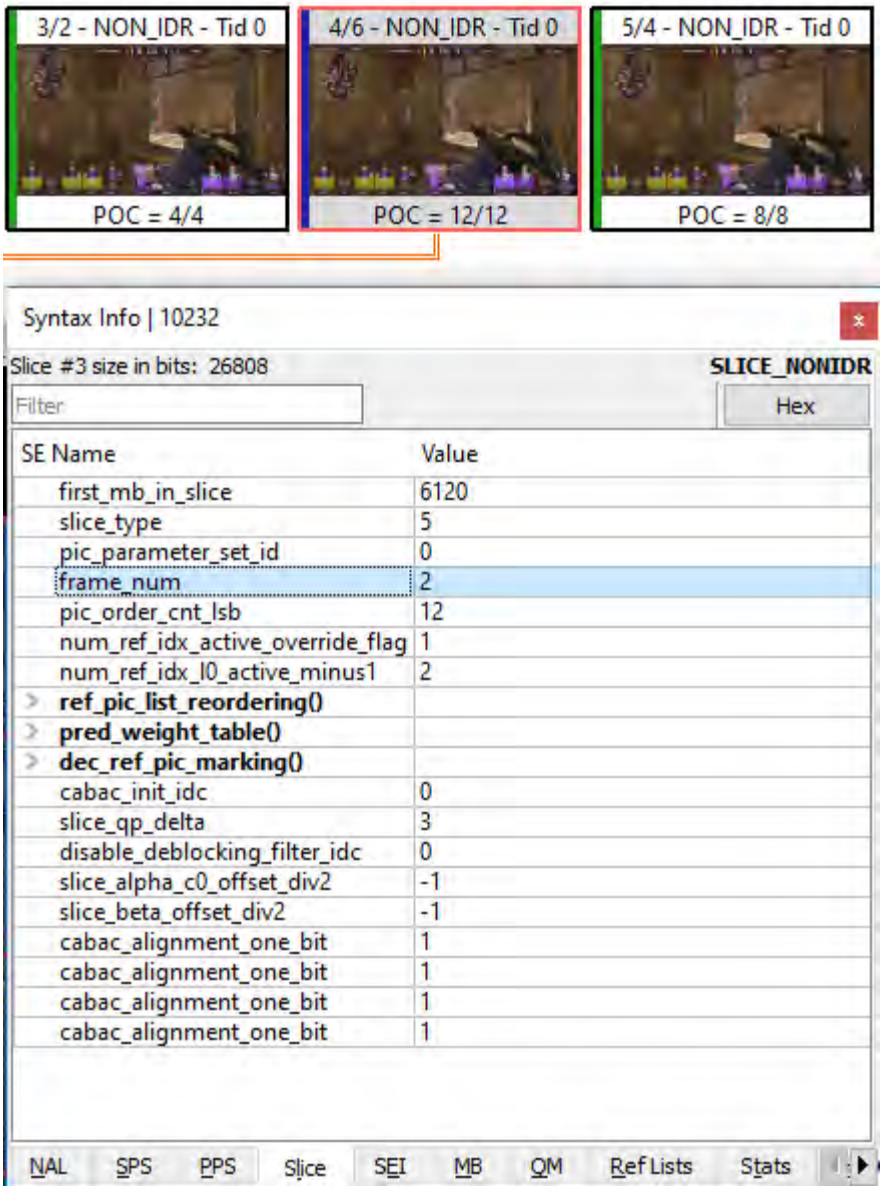
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



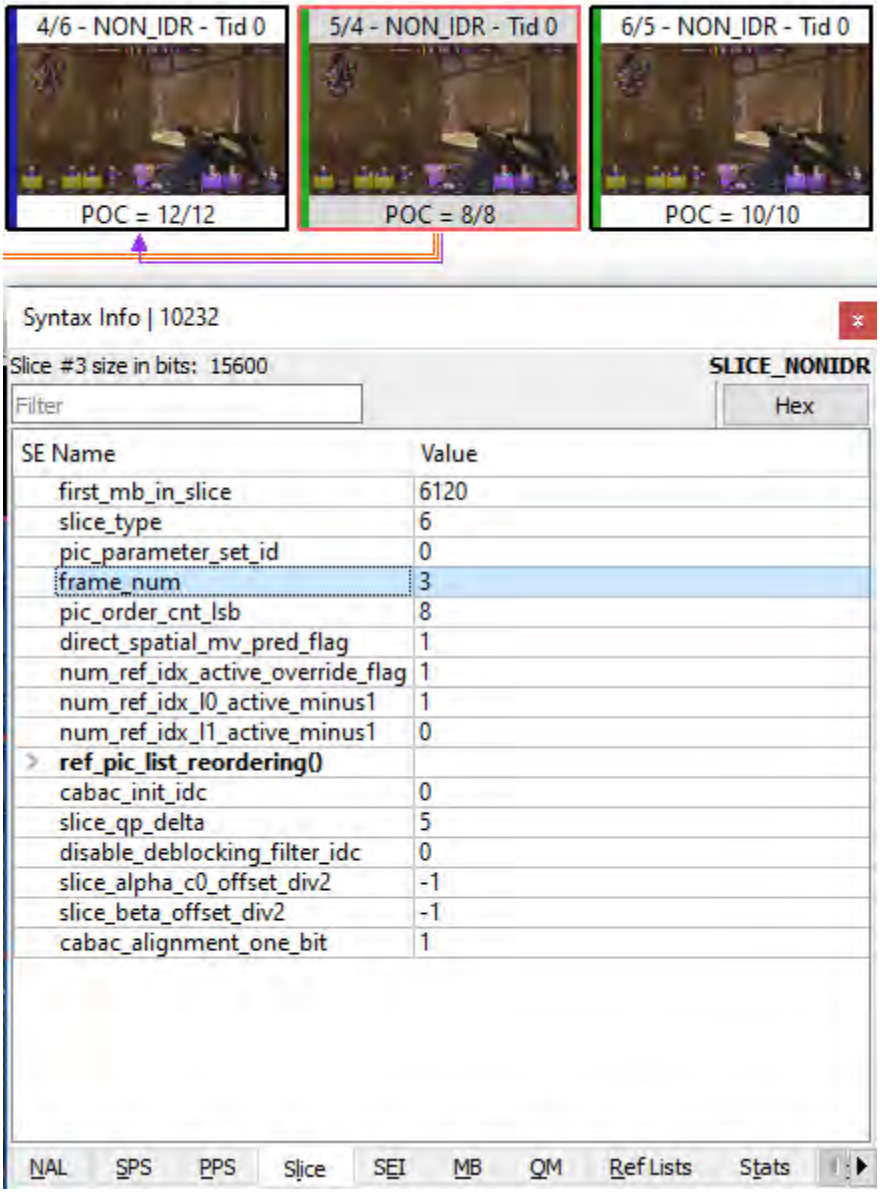
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

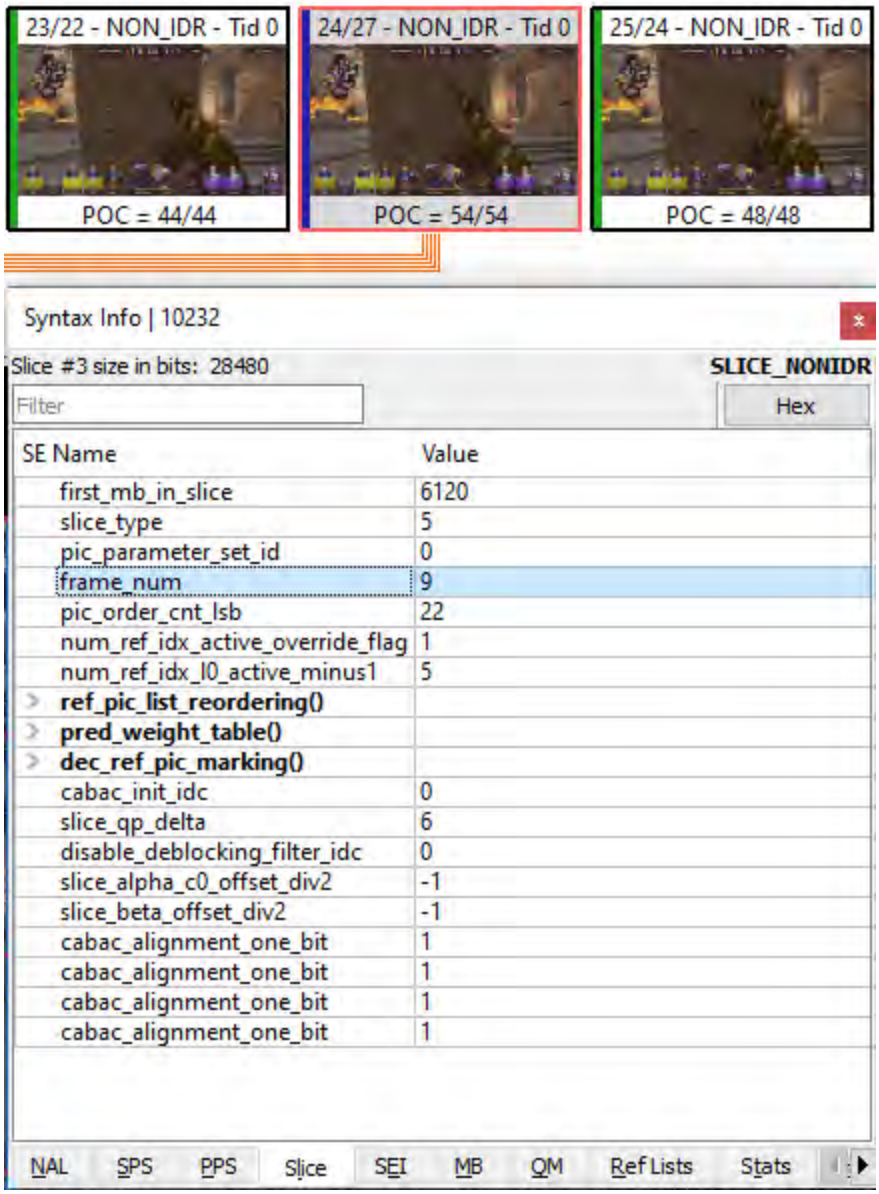


Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

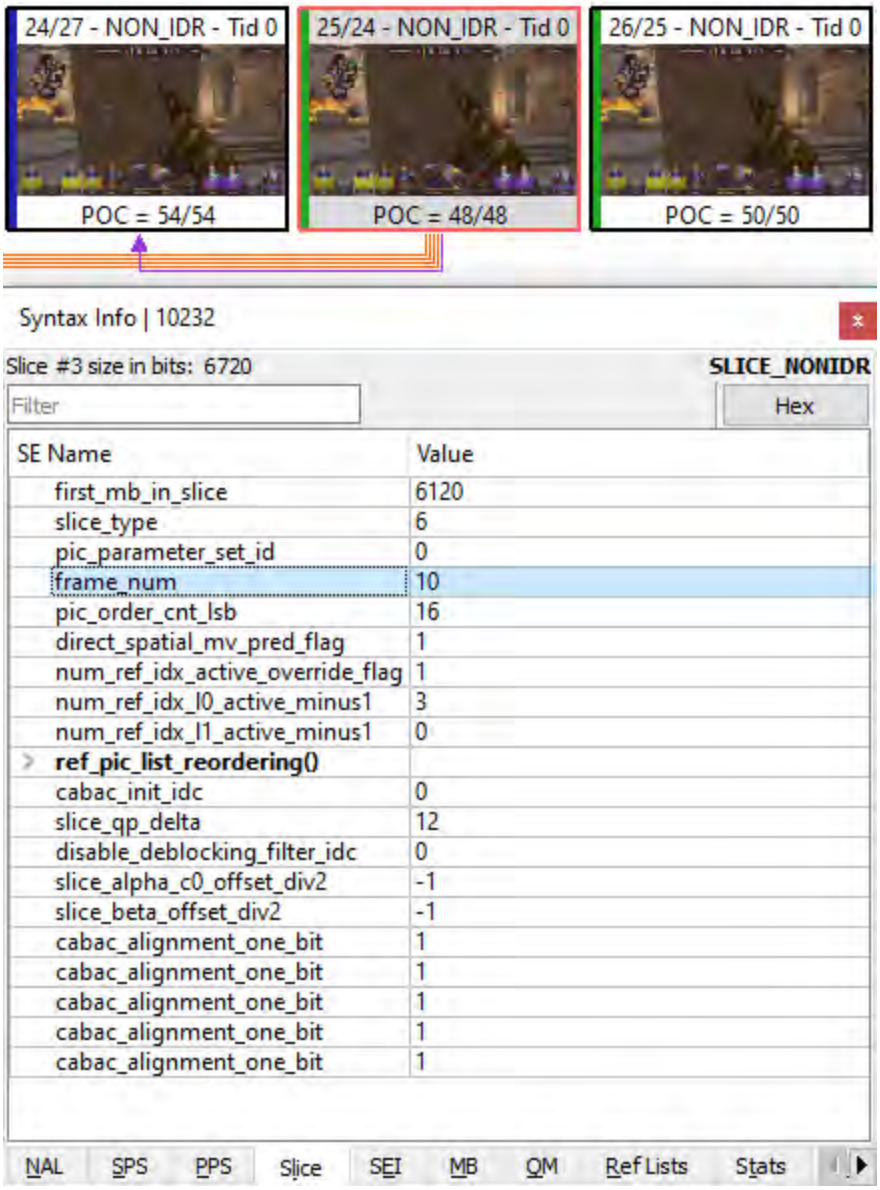
384. When encoding Twitch.tv video, on information and belief, Amazon performs using an independent numbering scheme, to assign consecutive reference pictures in encoding order with respective sequence indicator values that differ with respect to each other by a predetermined amount independent of one or more of the number of non-reference pictures encoded between consecutive reference pictures and the number of non-coded pictures between consecutive reference pictures, as demonstrated in the screenshots below using VQ Analyzer

software, where the frame number for reference pictures increases by 1 when there are three non-reference pictures between reference pictures.

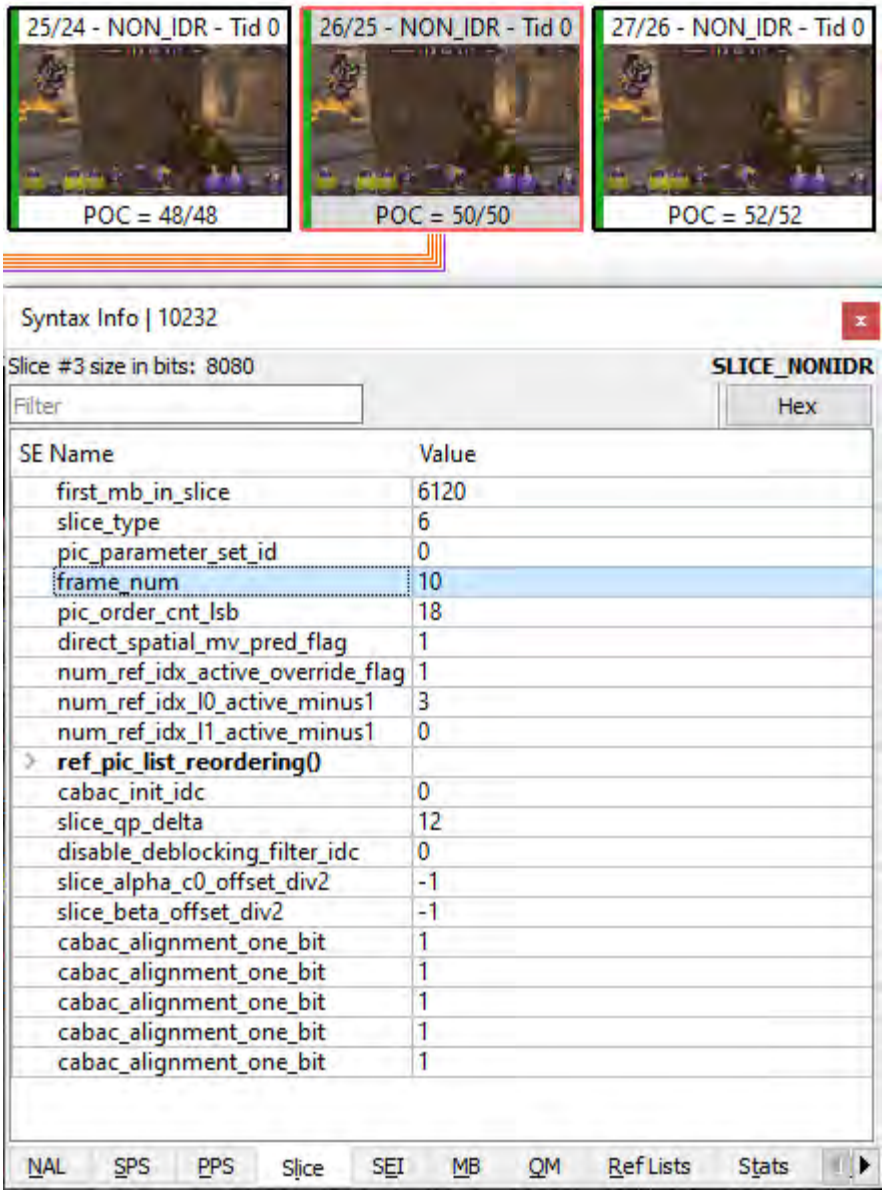
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



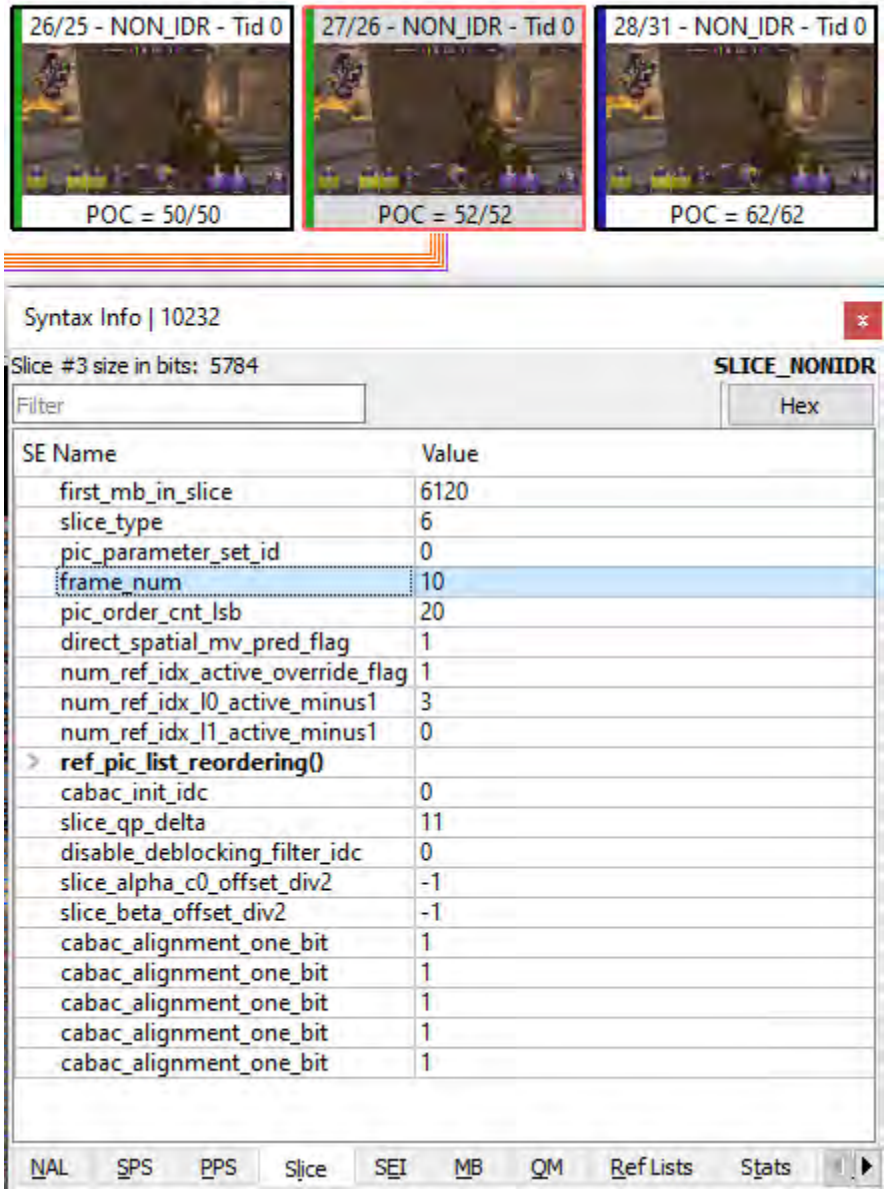
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



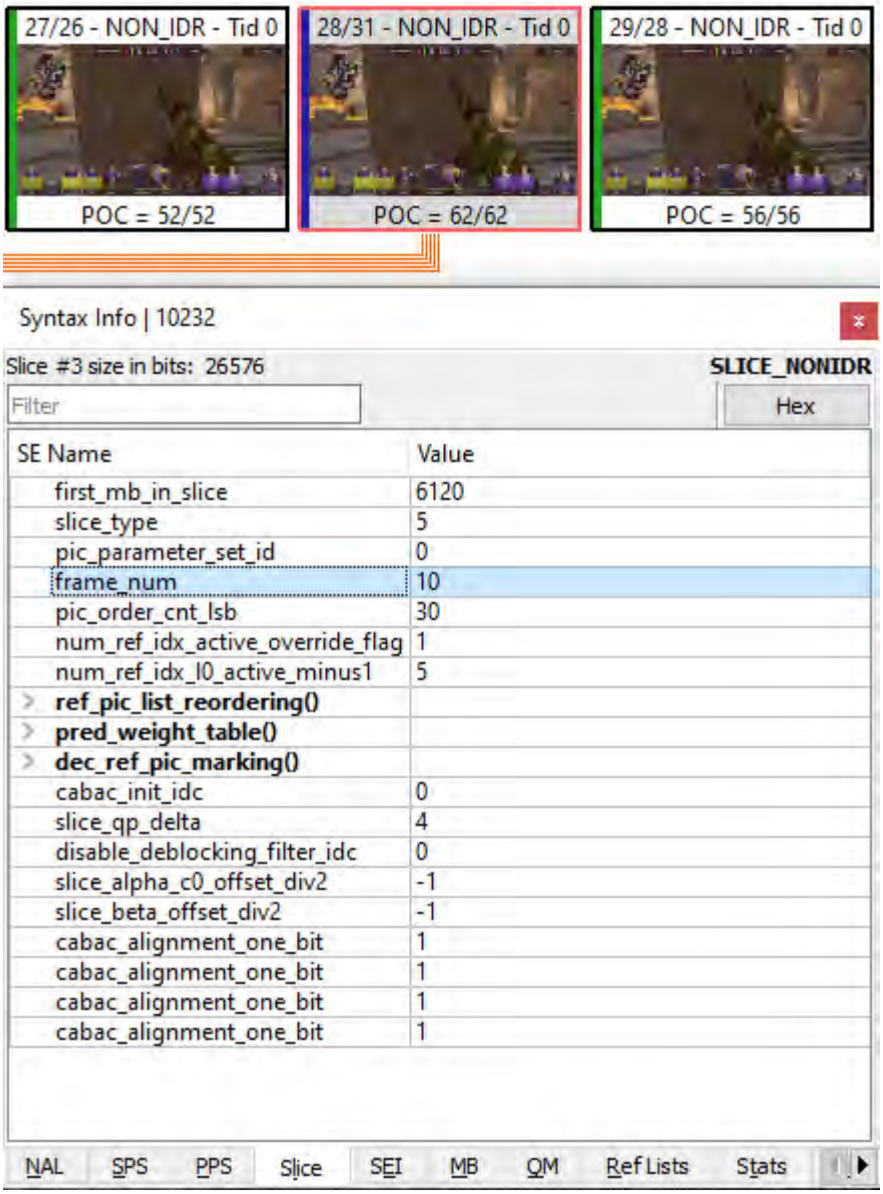
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



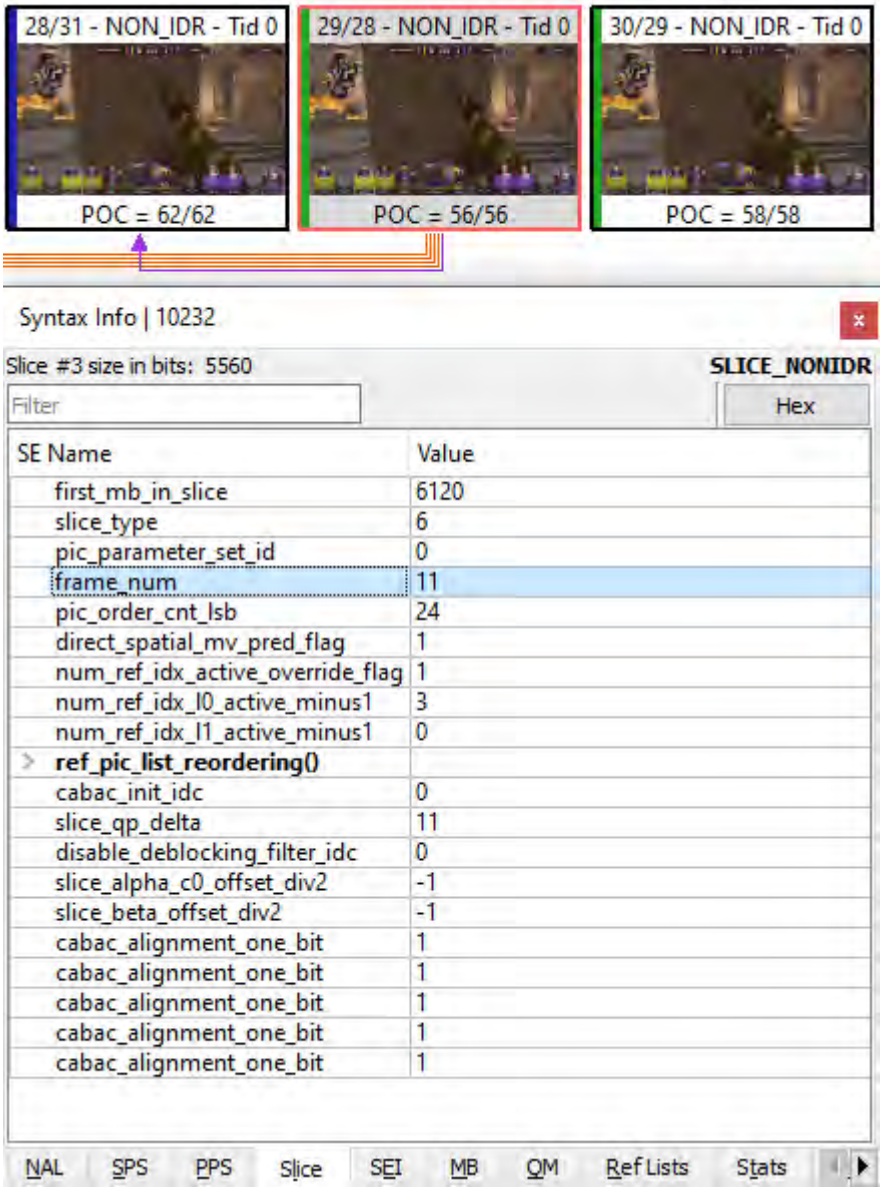
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

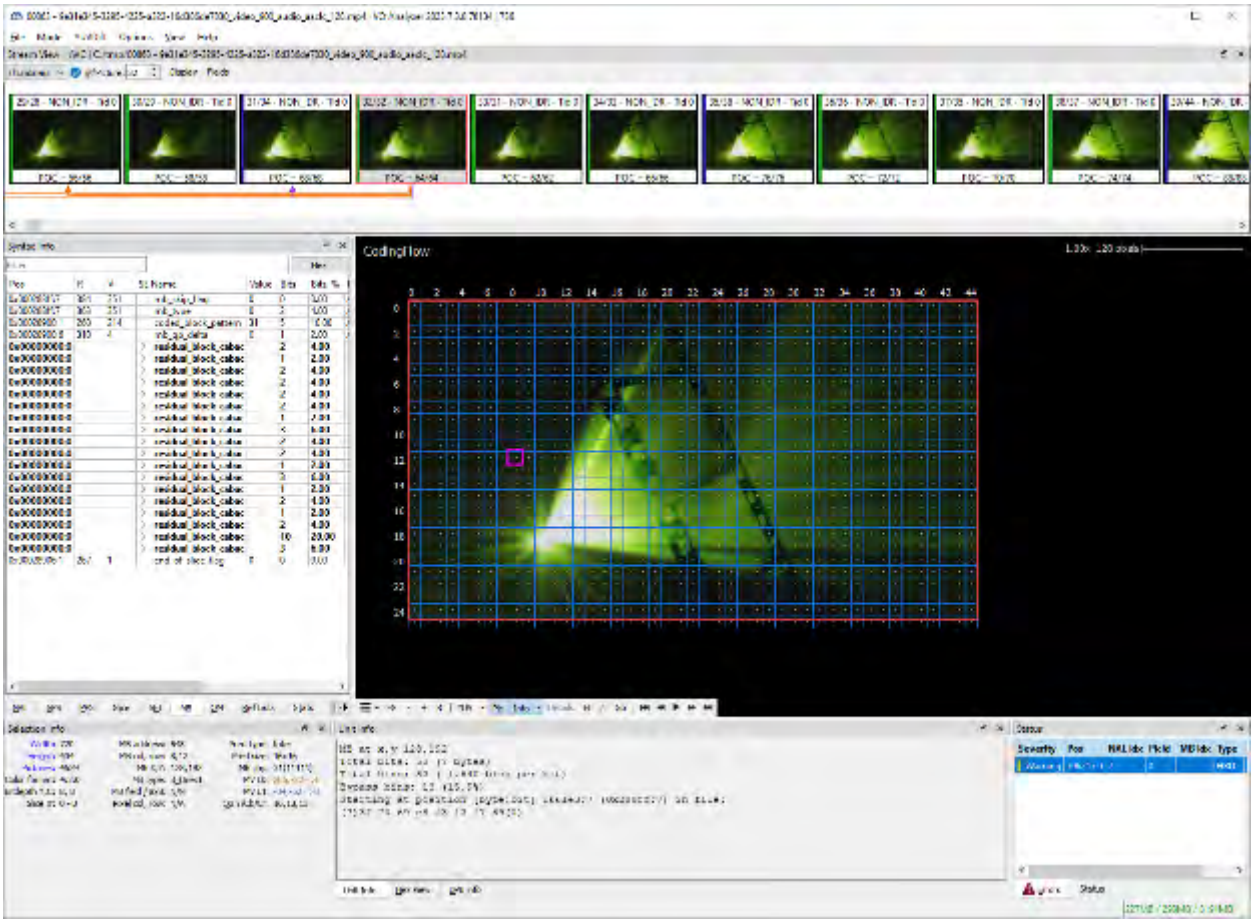
K. Amazon Makes, Imports, Uses, Sells, and/or Offers for Sale Products and Services that Infringe the ’148 Patent

385. The Accused Products infringe one or more claims of the ’148 Patent, including, for example, claim 1.

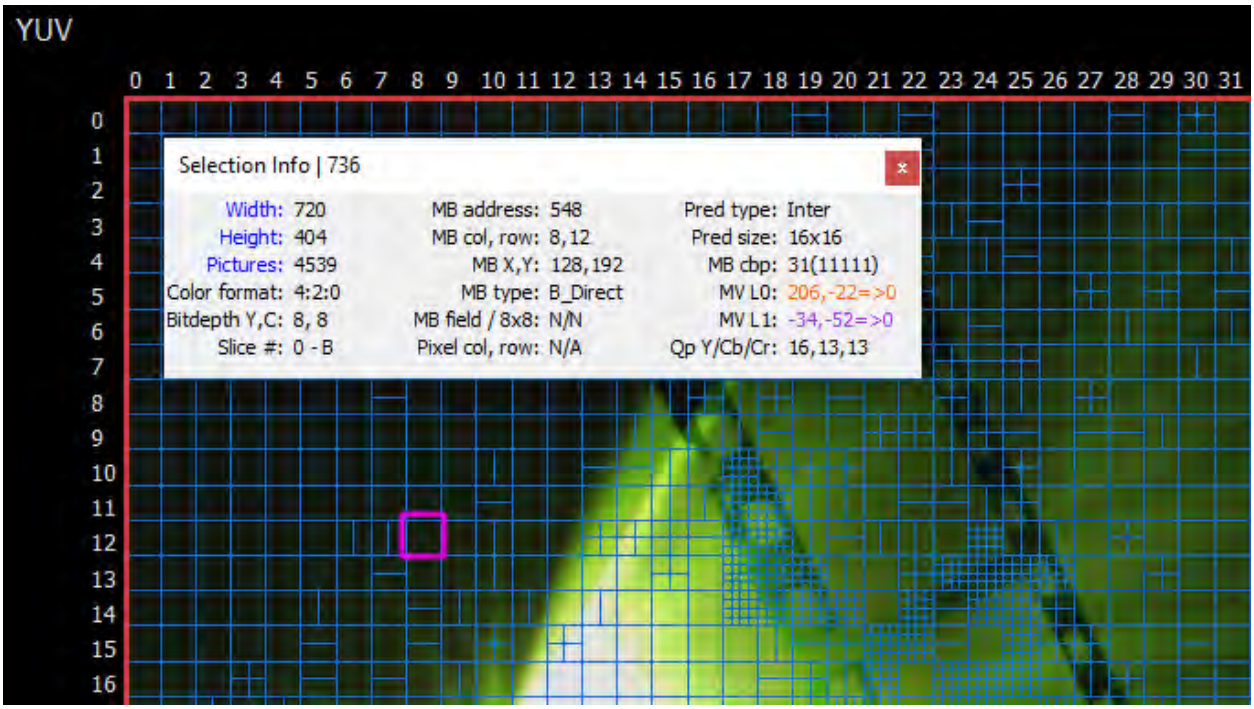
386. As just one example of infringement, on information and belief, Amazon performs a method of encoding a video sequence in a manner that is covered by claim 1 of the ’148 Patent

for Amazon Prime Video content (such as trailers), as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 212.

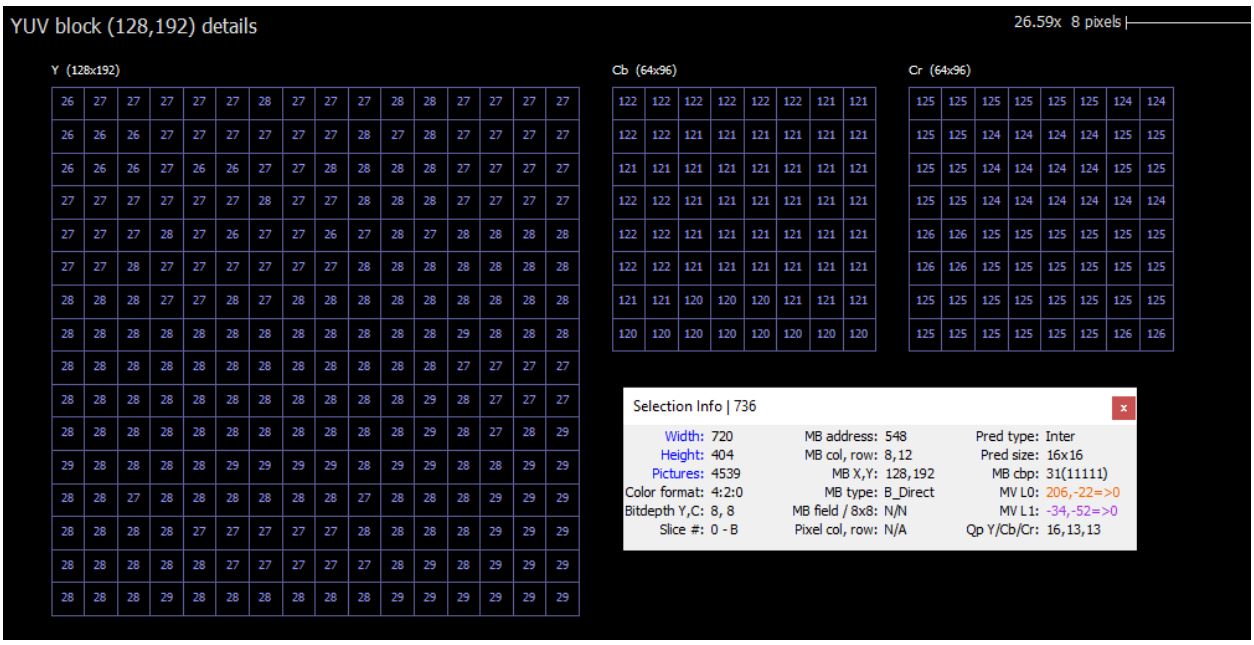
387. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs encoding a digital video sequence for use in a video coding application to produce an encoded video bit-stream representative of the digital video sequence, the digital video sequence comprising a number of frames, each frame of said sequence comprising an array of pixels divided into a plurality of blocks, each block comprising a certain number of said pixels, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



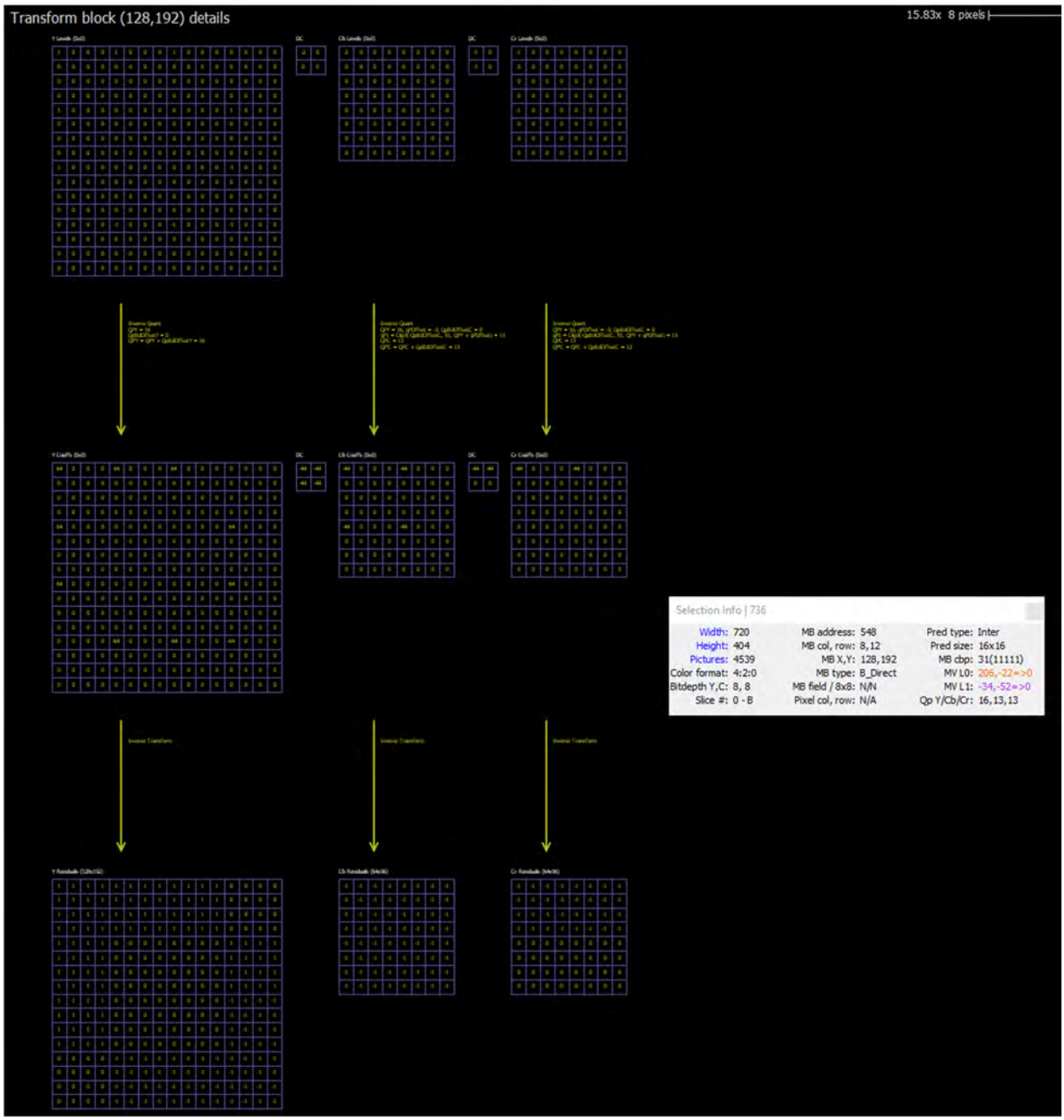
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



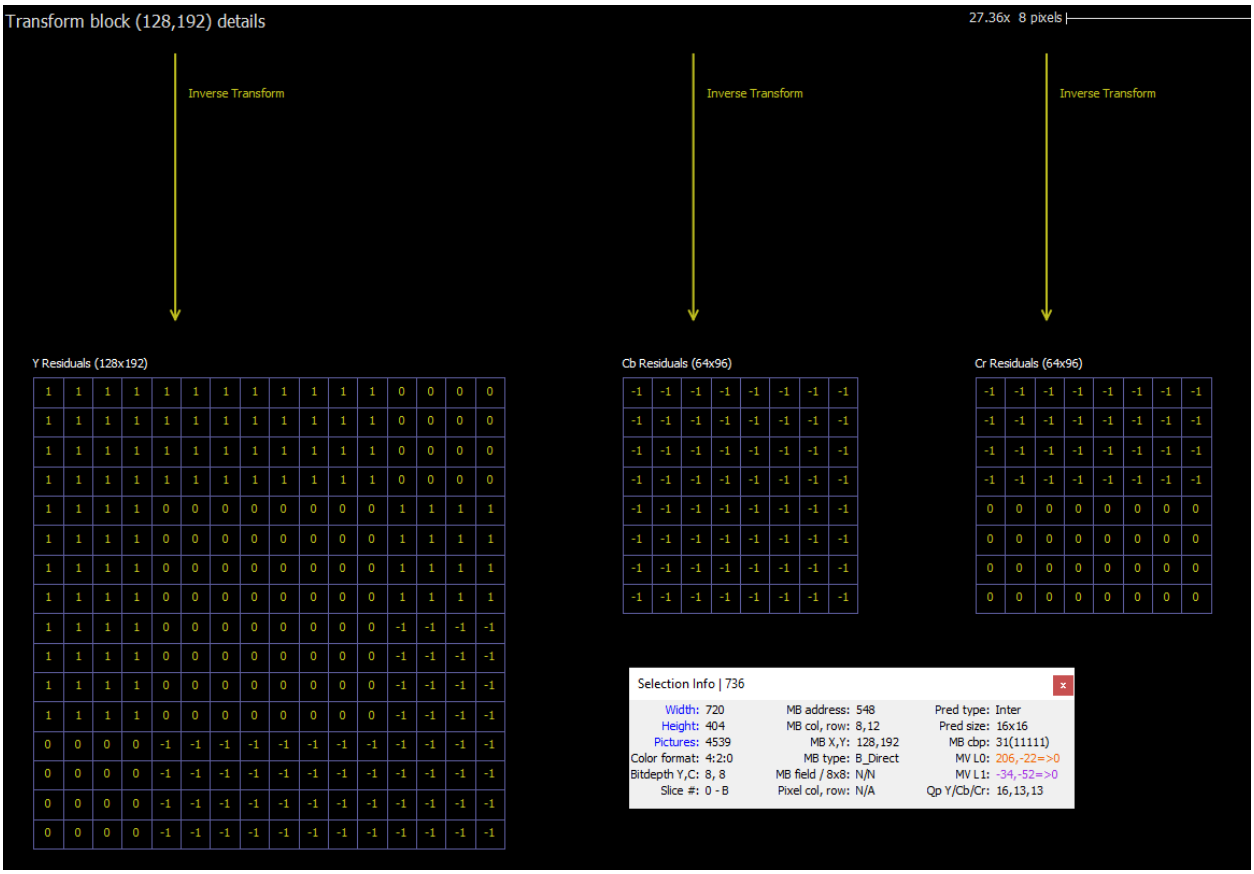
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

388. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs encoding a frame of the digital video sequence by applying motion compensated

prediction to blocks of pixels for producing corresponding blocks of prediction error values, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Transform block (128,192) details

Inverse Transform

Y Residuals (128x192)

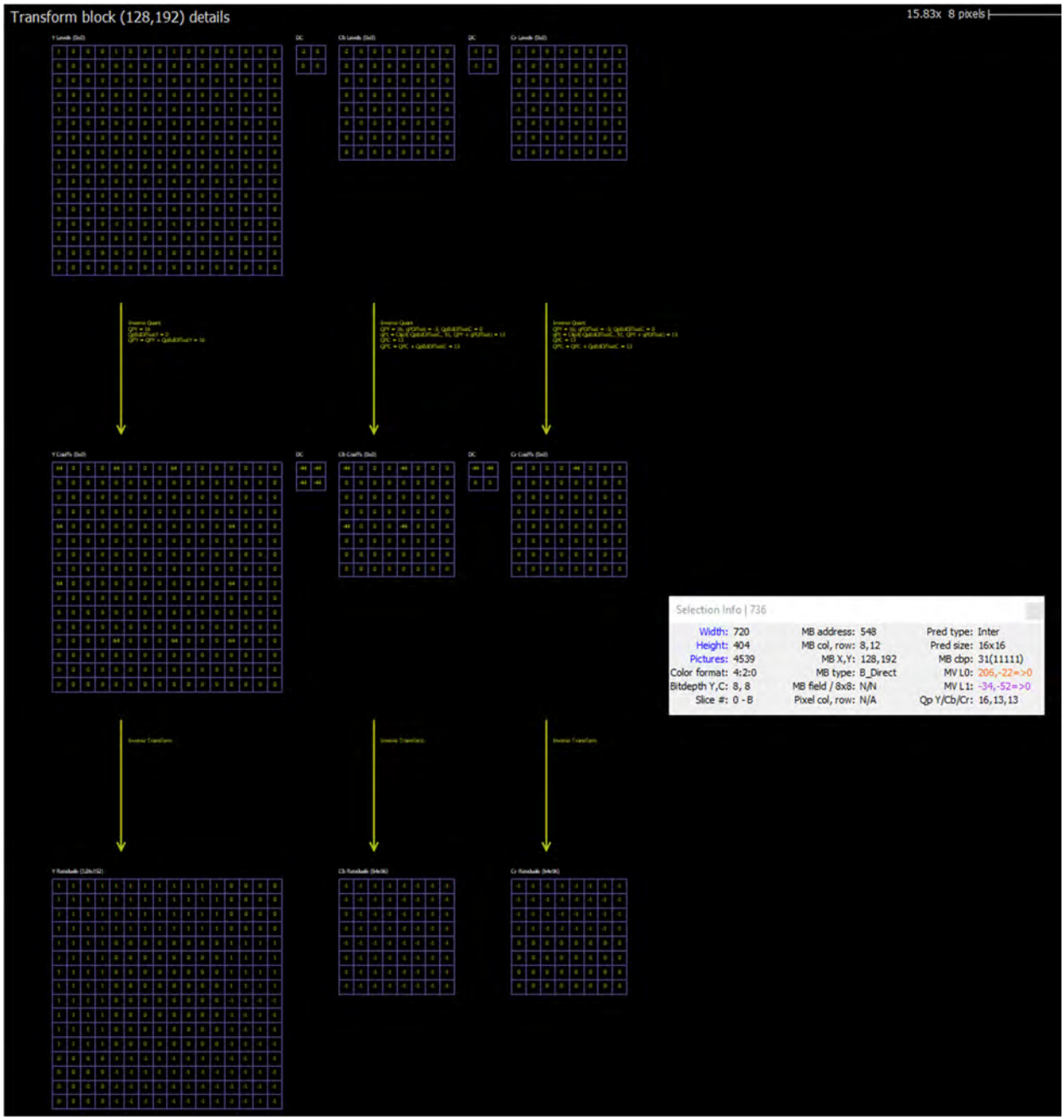
1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1
1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1
1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1
1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1
1	1	1	1	0	0	0	0	0	0	0	0	-1	-1	-1	-1
1	1	1	1	0	0	0	0	0	0	0	0	-1	-1	-1	-1
1	1	1	1	0	0	0	0	0	0	0	0	-1	-1	-1	-1
1	1	1	1	0	0	0	0	0	0	0	0	-1	-1	-1	-1
0	0	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0	0	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0	0	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0	0	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1

Selection Info | 736

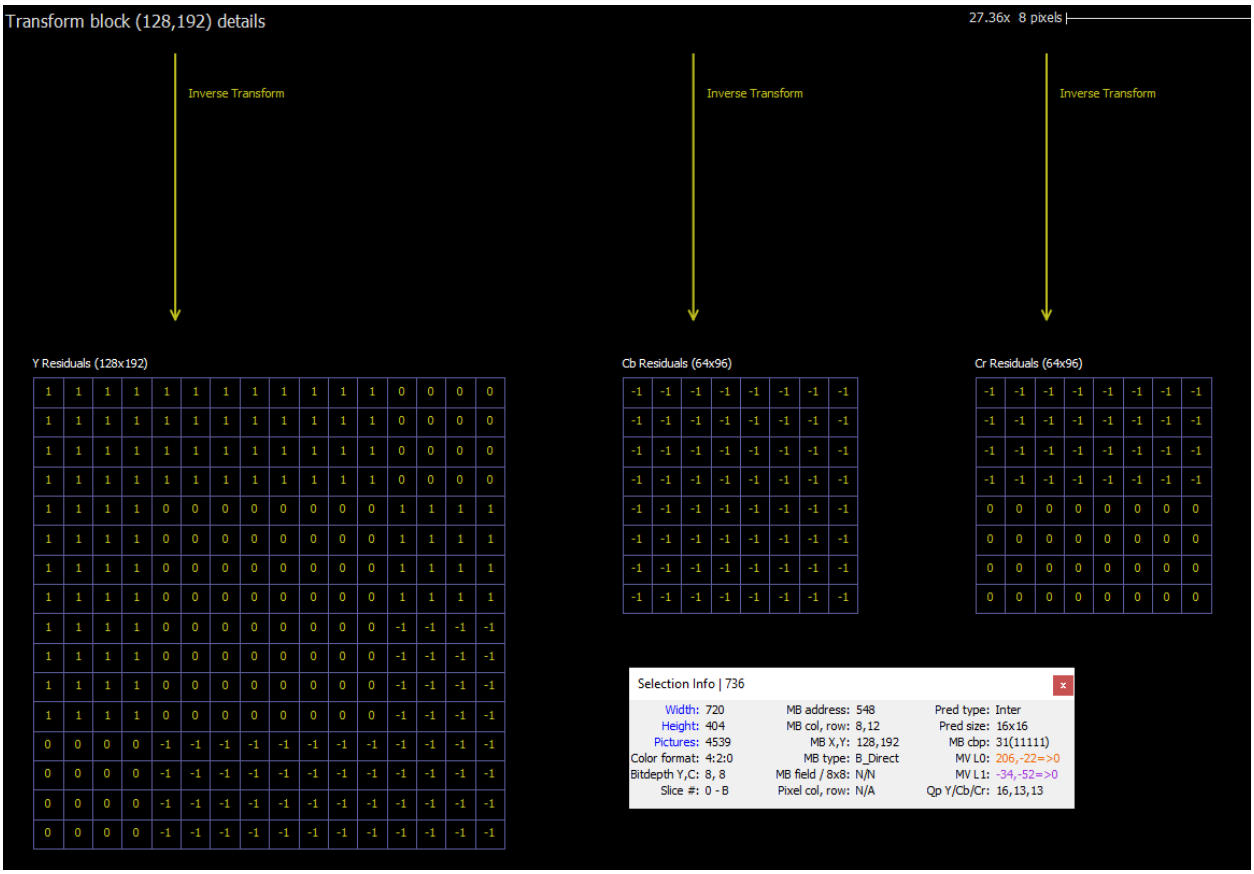
Width:	720	MB address:	548	Pred type:	Inter
Height:	404	MB col, row:	8,12	Pred size:	16x16
Pictures:	4539	MB X,Y:	128,192	MB cbp:	31(11111)
Color format:	4:2:0	MB type:	B_Direct	MV L0:	206,-22=>0
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/N	MV L1:	-34,-52=>0
Slice #:	0 - B	Pixel col, row:	N/A	Qp Y/Cb/Cr:	16,13,13

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

389. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs applying a transform coding technique to said blocks of prediction error values to produce sets of transform coefficient values representative of said blocks of prediction error values, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Transform block (128,192) details

Inverse Transform

Y Residuals (128x192)

1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1
1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1
1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1
1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1
1	1	1	1	0	0	0	0	0	0	0	0	-1	-1	-1	-1
1	1	1	1	0	0	0	0	0	0	0	0	-1	-1	-1	-1
1	1	1	1	0	0	0	0	0	0	0	0	-1	-1	-1	-1
1	1	1	1	0	0	0	0	0	0	0	0	-1	-1	-1	-1
0	0	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0	0	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0	0	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0	0	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1

Selection Info | 736

Width:	720	MB address:	548	Pred type:	Inter
Height:	404	MB col, row:	8,12	Pred size:	16x16
Pictures:	4539	MB X,Y:	128,192	MB cbp:	31(11111)
Color format:	4:2:0	MB type:	B_Direct	MV L0:	206,-22=>0
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/N	MV L1:	-34,-52=>0
Slice #:	0 - B	Pixel col, row:	N/A	Qp Y/Cb/Cr:	16,13,13

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

390. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs defining a default level of quantization for use in encoding of the digital video sequence to quantize the sets of transform coefficient values, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info

4541 units total | Hex

Pos	NAL Type	TID	Size
553	0: 7: SPS - Sequence parameter set	0	44
600	1: 8: PPS - Picture parameter set	0	5
79621	2: 5: IDR - Coded slice of an IDR picture	0	99
79724	3: 1: non-IDR - Coded slice of a non-IDR picture	0	212
79940	4: 1: non-IDR - Coded slice of a non-IDR picture	0	61
80005	5: 1: non-IDR - Coded slice of a non-IDR picture	0	19
80028	6: 1: non-IDR - Coded slice of a non-IDR picture	0	21
80053	7: 1: non-IDR - Coded slice of a non-IDR picture	0	24
80081	8: 1: non-IDR - Coded slice of a non-IDR picture	0	72
80157	9: 1: non-IDR - Coded slice of a non-IDR picture	0	53
80214	10: 1: non-IDR - Coded slice of a non-IDR picture	0	303
80521	11: 1: non-IDR - Coded slice of a non-IDR picture	0	35
80560	12: 1: non-IDR - Coded slice of a non-IDR picture	0	552
81116	13: 1: non-IDR - Coded slice of a non-IDR picture	0	87
81207	14: 1: non-IDR - Coded slice of a non-IDR picture	0	1915
83126	15: 1: non-IDR - Coded slice of a non-IDR picture	0	726
83856	16: 1: non-IDR - Coded slice of a non-IDR picture	0	198
92853	17: 1: non-IDR - Coded slice of a non-IDR picture	0	94
92951	18: 1: non-IDR - Coded slice of a non-IDR picture	0	225
93180	19: 1: non-IDR - Coded slice of a non-IDR picture	0	318
93502	20: 1: non-IDR - Coded slice of a non-IDR picture	0	6875
100381	21: 1: non-IDR - Coded slice of a non-IDR picture	0	1527
101912	22: 1: non-IDR - Coded slice of a non-IDR picture	0	301
102217	23: 1: non-IDR - Coded slice of a non-IDR picture	0	606
102827	24: 1: non-IDR - Coded slice of a non-IDR picture	0	1100
103931	25: 1: non-IDR - Coded slice of a non-IDR picture	0	12152
116087	26: 1: non-IDR - Coded slice of a non-IDR picture	0	4253
120344	27: 1: non-IDR - Coded slice of a non-IDR picture	0	1016

No filter | Extract selected units

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

4541 units total

Hex

Pos	NAL Type	TID	Size
83126	15: 1: non-IDR - Coded slice of a non-IDR picture	0	726
83856	16: 1: non-IDR - Coded slice of a non-IDR picture	0	198
92853	17: 1: non-IDR - Coded slice of a non-IDR picture	0	94
92951	18: 1: non-IDR - Coded slice of a non-IDR picture	0	225
93180	19: 1: non-IDR - Coded slice of a non-IDR picture	0	318
93502	20: 1: non-IDR - Coded slice of a non-IDR picture	0	6875
100381	21: 1: non-IDR - Coded slice of a non-IDR picture	0	1527
101912	22: 1: non-IDR - Coded slice of a non-IDR picture	0	301
102217	23: 1: non-IDR - Coded slice of a non-IDR picture	0	606
102827	24: 1: non-IDR - Coded slice of a non-IDR picture	0	1100
103931	25: 1: non-IDR - Coded slice of a non-IDR picture	0	12152
116087	26: 1: non-IDR - Coded slice of a non-IDR picture	0	4253
120344	27: 1: non-IDR - Coded slice of a non-IDR picture	0	1016
121364	28: 1: non-IDR - Coded slice of a non-IDR picture	0	2717
124085	29: 1: non-IDR - Coded slice of a non-IDR picture	0	2144
126233	30: 1: non-IDR - Coded slice of a non-IDR picture	0	9040
142869	31: 1: non-IDR - Coded slice of a non-IDR picture	0	5170
148043	32: 1: non-IDR - Coded slice of a non-IDR picture	0	680
148727	33: 1: non-IDR - Coded slice of a non-IDR picture	0	14378
163109	34: 1: non-IDR - Coded slice of a non-IDR picture	0	7491
170604	35: 1: non-IDR - Coded slice of a non-IDR picture	0	5568
176176	36: 1: non-IDR - Coded slice of a non-IDR picture	0	1288
177468	37: 1: non-IDR - Coded slice of a non-IDR picture	0	17139
194611	38: 1: non-IDR - Coded slice of a non-IDR picture	0	9028
203643	39: 1: non-IDR - Coded slice of a non-IDR picture	0	5776
209423	40: 1: non-IDR - Coded slice of a non-IDR picture	0	4836
214263	41: 1: non-IDR - Coded slice of a non-IDR picture	0	17024
231291	42: 1: non-IDR - Coded slice of a non-IDR picture	0	9415

No filter

Extract selected units

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

Filter | Hex

SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	0
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	3
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	
deblocking_filter_control_pr	
constrained_intra_pred_flag	
redundant_pic_cnt_present_f	
rbsp_stop_one_bit	
rbsp_alignment_zero_bit	
rbsp_alignment_zero_bit	
rbsp_alignment_zero_bit	
rbsp_alignment_zero_bit	
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0
rbsp_alignment_zero_bit	0

pic_init_qp_minus26 specifies the initial value minus 26 of SliceQPY for each slice. The initial value is modified at the slice layer when a non-zero value of slice_qp_delta is decoded, and is modified further when a non-zero value of mb_qp_delta is decoded at the macroblock layer. The value of **pic_init_qp_minus26** shall be in the range of −(26 + QpBdOffsetY) to +25, inclusive.

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

Slice #0 size in bits: 59928 SLICE_NONIDR

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	6
pic_parameter_set_id	0
frame_num	14
pic_order_cnt_lsb	64
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	2
num_ref_idx_l1_active_minus1	0
ref_pic_list_reordering()	
dec_ref_pic_marking()	
cabac_init_idc	0
slice_qp_delta	-10
disable_deblockin	
slice_alpha_c0_off	
slice_beta_offset_	
cabac_alignment_	
cabac_alignment_	
cabac_alignment_	
cabac_alignment_	
cabac_alignment_	

slice_qp_delta specifies the initial value of QPY to be used for all the macroblocks in the slice until modified by the value of mb_qp_delta in the macroblock layer. The initial QPY quantisation parameter for the slice is computed as

SliceQPY = 26 + pic_init_qp_minus26 + **slice_qp_delta** (7-29)

The value of **slice_qp_delta** shall be limited such that SliceQPY is in the range of −QpBdOffsetY to +51, inclusive.

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

Slice #0 size in bits: 115024 **SLICE_NONIDR**

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	5
pic_parameter_set_id	0
frame_num	13
pic_order_cnt_lsb	68
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	4
ref_pic_list_reordering()	
pred_weight_table()	
dec_ref_pic_marking()	
cabac_init_idc	0
slice_qp_delta	-10
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	-1
slice_beta_offset_div2	-1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

Slice #0 size in bits: 44544 **SLICE_NONIDR**

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	6
pic_parameter_set_id	0
frame_num	15
pic_order_cnt_lsb	62
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	0
num_ref_idx_l1_active_minus1	1
ref_pic_list_reordering()	
cabac_init_idc	0
slice_qp_delta	-10
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	-1
slice_beta_offset_div2	-1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

Filter | Hex

Pos	R	V	SE Name	Value	Bits	Bits, %
0x000288ff:7	384	351	mb_skip_flag	0	0	0.00
0x000288ff:7	363	351	mb_type	0	2	4.00
0x00028900:1	260	214	coded_block_pattern	31	5	10.00
0x00028900:6	310	4	mb_qp_delta	0	1	2.00

mb_qp_delta can change the value of QPY in the macroblock layer. The decoded value of **mb_qp_delta** shall be in the range of −(26 + QpBdOffsetY / 2) to +(25 + QpBdOffsetY / 2), inclusive. **mb_qp_delta** shall be inferred to be equal to 0 when it is not present for any macroblock (including P_Skip and B_Skip macroblock types).

The value of QPY is derived as

QPY = ((QPY,PREV + **mb_qp_delta** + 52 + 2 * QpBdOffsetY) % (52 + QpBdOffsetY)) − QpBdOffsetY (7-36)

where QPY,PREV is the luma quantisation parameter, QPY, of the previous macroblock in decoding order in the current slice. For the first macroblock in the slice QPY,PREV is initially set equal to SliceQPY derived in Equation 7-29 at the start of each slice.

The value of QP'Y is derived as

QP'Y = QPY + QpBdOffsetY (7-37)

The variable TransformBypassModeFlag is derived as follows.

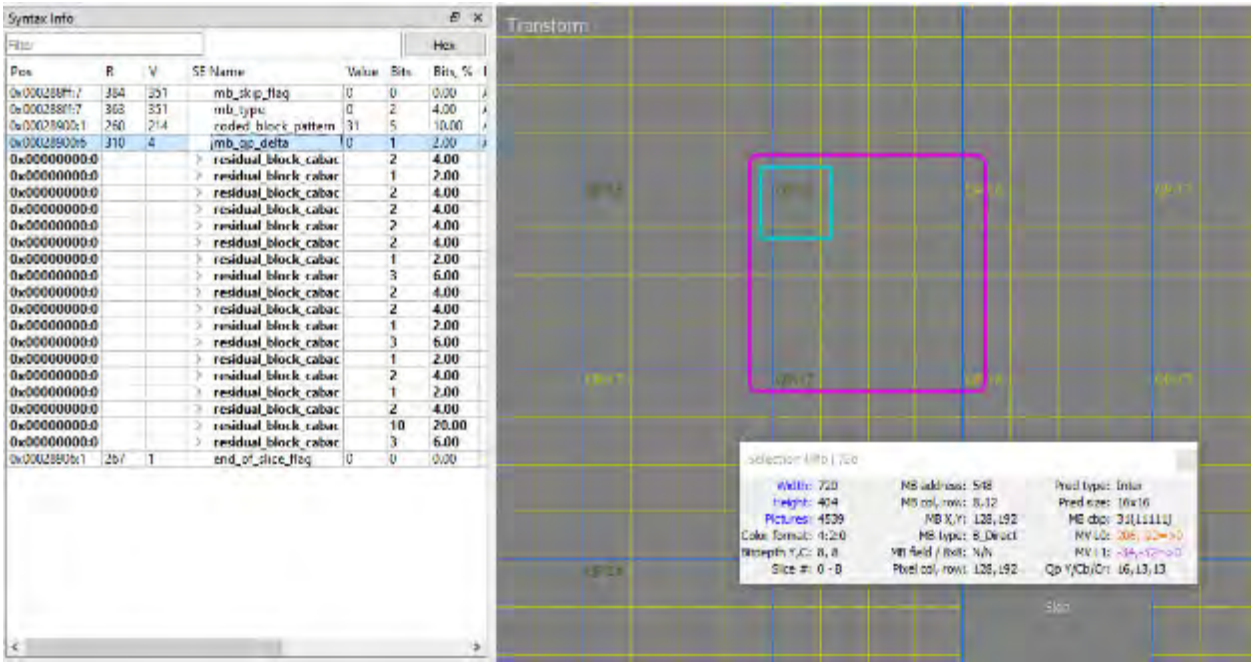
- If qpprime_y_zero_transform_bypass_flag is equal to 1 and QP'Y is equal to 0, TransformBypassModeFlag is set equal to 1.
- Otherwise (qpprime_y_zero_transform_bypass_flag is equal to 0 or QP'Y is not equal to 0),

TransformBypassModeFlag is set equal to 0.

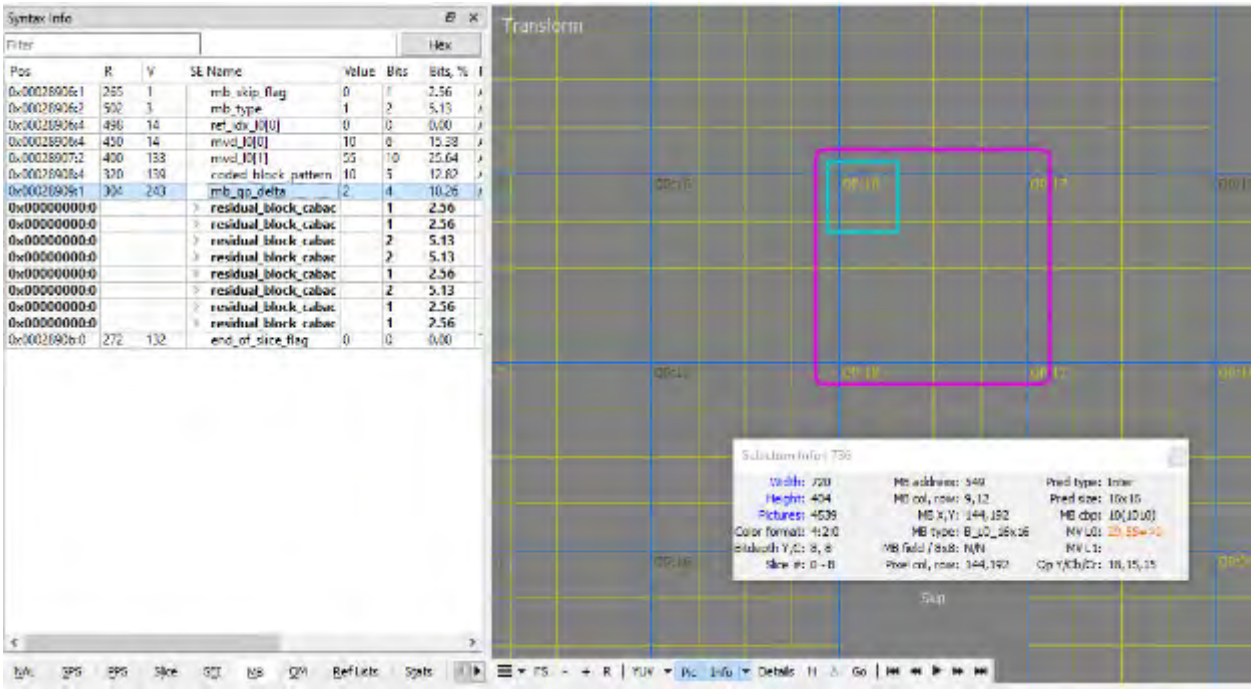
Selection Info | 736

Width:	720	MB address:	548	Pred type:	Inter
Height:	404	MB col, row:	8,12	Pred size:	16x16
Pictures:	4539	MB X,Y:	128,192	MB cbp:	31(11111)
Color format:	4:2:0	MB type:	B_Direct	MV L0:	206,-22=>0
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/N	MV L1:	-34,-52=>0
Slice #:	0 - B	Pixel col, row:	N/A	Qp Y/Cb/Cr:	16,13,13

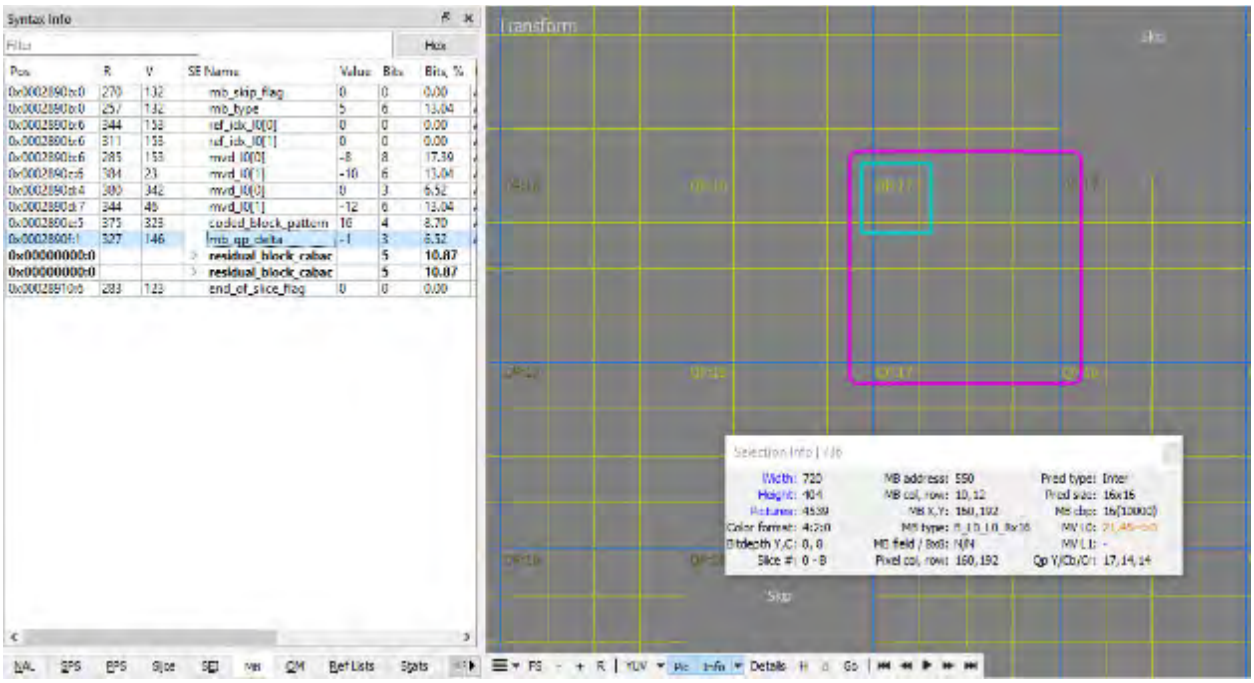
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

391. When encoding Amazon Prime Video trailers, on information and belief, Amazon performs providing an indication of the default level of quantization to a decoding process, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info

4541 units total | Hex

Pos	NAL Type	TID	Size
553	0: 7: SPS - Sequence parameter set	0	44
600	1: 8: PPS - Picture parameter set	0	5
79621	2: 5: IDR - Coded slice of an IDR picture	0	99
79724	3: 1: non-IDR - Coded slice of a non-IDR picture	0	212
79940	4: 1: non-IDR - Coded slice of a non-IDR picture	0	61
80005	5: 1: non-IDR - Coded slice of a non-IDR picture	0	19
80028	6: 1: non-IDR - Coded slice of a non-IDR picture	0	21
80053	7: 1: non-IDR - Coded slice of a non-IDR picture	0	24
80081	8: 1: non-IDR - Coded slice of a non-IDR picture	0	72
80157	9: 1: non-IDR - Coded slice of a non-IDR picture	0	53
80214	10: 1: non-IDR - Coded slice of a non-IDR picture	0	303
80521	11: 1: non-IDR - Coded slice of a non-IDR picture	0	35
80560	12: 1: non-IDR - Coded slice of a non-IDR picture	0	552
81116	13: 1: non-IDR - Coded slice of a non-IDR picture	0	87
81207	14: 1: non-IDR - Coded slice of a non-IDR picture	0	1915
83126	15: 1: non-IDR - Coded slice of a non-IDR picture	0	726
83856	16: 1: non-IDR - Coded slice of a non-IDR picture	0	198
92853	17: 1: non-IDR - Coded slice of a non-IDR picture	0	94
92951	18: 1: non-IDR - Coded slice of a non-IDR picture	0	225
93180	19: 1: non-IDR - Coded slice of a non-IDR picture	0	318
93502	20: 1: non-IDR - Coded slice of a non-IDR picture	0	6875
100381	21: 1: non-IDR - Coded slice of a non-IDR picture	0	1527
101912	22: 1: non-IDR - Coded slice of a non-IDR picture	0	301
102217	23: 1: non-IDR - Coded slice of a non-IDR picture	0	606
102827	24: 1: non-IDR - Coded slice of a non-IDR picture	0	1100
103931	25: 1: non-IDR - Coded slice of a non-IDR picture	0	12152
116087	26: 1: non-IDR - Coded slice of a non-IDR picture	0	4253
120344	27: 1: non-IDR - Coded slice of a non-IDR picture	0	1016

No filter | Extract selected units

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

4541 units total

Hex

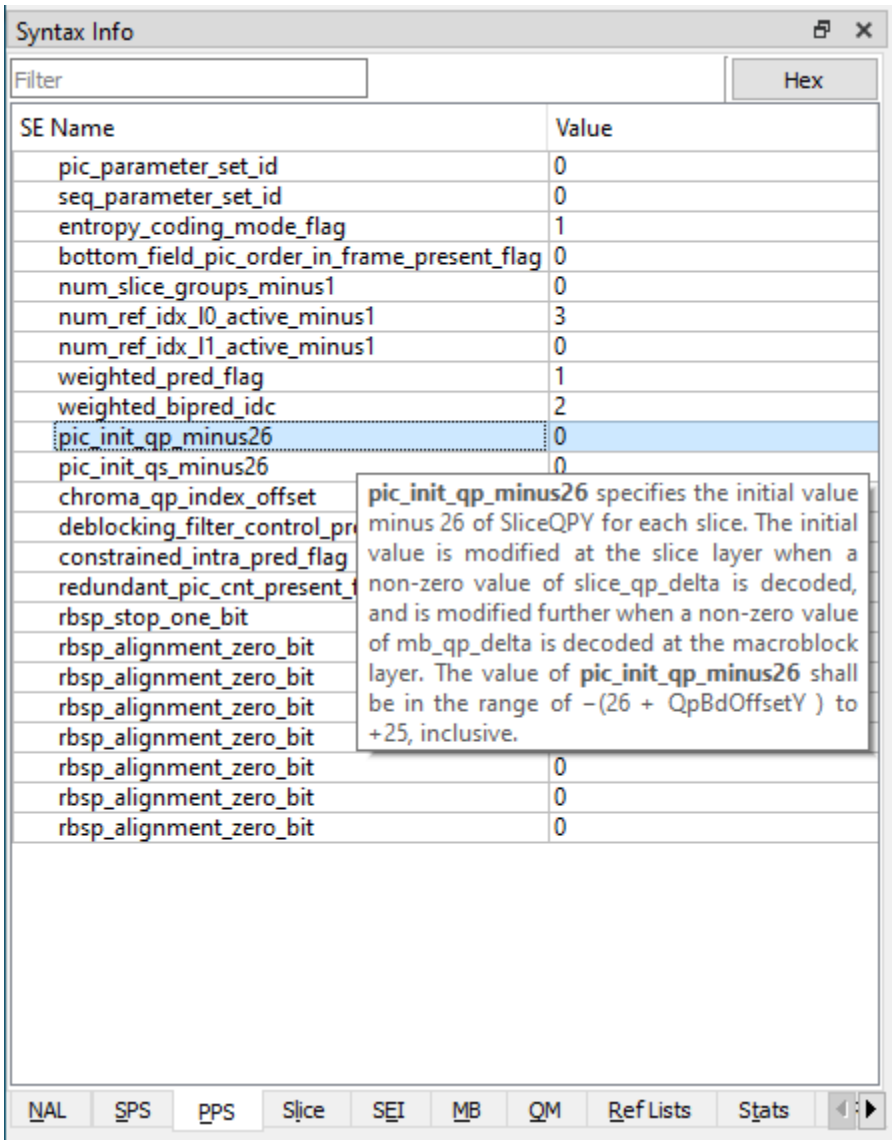
Pos	NAL Type	TID	Size
83126	15: 1: non-IDR - Coded slice of a non-IDR picture	0	726
83856	16: 1: non-IDR - Coded slice of a non-IDR picture	0	198
92853	17: 1: non-IDR - Coded slice of a non-IDR picture	0	94
92951	18: 1: non-IDR - Coded slice of a non-IDR picture	0	225
93180	19: 1: non-IDR - Coded slice of a non-IDR picture	0	318
93502	20: 1: non-IDR - Coded slice of a non-IDR picture	0	6875
100381	21: 1: non-IDR - Coded slice of a non-IDR picture	0	1527
101912	22: 1: non-IDR - Coded slice of a non-IDR picture	0	301
102217	23: 1: non-IDR - Coded slice of a non-IDR picture	0	606
102827	24: 1: non-IDR - Coded slice of a non-IDR picture	0	1100
103931	25: 1: non-IDR - Coded slice of a non-IDR picture	0	12152
116087	26: 1: non-IDR - Coded slice of a non-IDR picture	0	4253
120344	27: 1: non-IDR - Coded slice of a non-IDR picture	0	1016
121364	28: 1: non-IDR - Coded slice of a non-IDR picture	0	2717
124085	29: 1: non-IDR - Coded slice of a non-IDR picture	0	2144
126233	30: 1: non-IDR - Coded slice of a non-IDR picture	0	9040
142869	31: 1: non-IDR - Coded slice of a non-IDR picture	0	5170
148043	32: 1: non-IDR - Coded slice of a non-IDR picture	0	680
148727	33: 1: non-IDR - Coded slice of a non-IDR picture	0	14378
163109	34: 1: non-IDR - Coded slice of a non-IDR picture	0	7491
170604	35: 1: non-IDR - Coded slice of a non-IDR picture	0	5568
176176	36: 1: non-IDR - Coded slice of a non-IDR picture	0	1288
177468	37: 1: non-IDR - Coded slice of a non-IDR picture	0	17139
194611	38: 1: non-IDR - Coded slice of a non-IDR picture	0	9028
203643	39: 1: non-IDR - Coded slice of a non-IDR picture	0	5776
209423	40: 1: non-IDR - Coded slice of a non-IDR picture	0	4836
214263	41: 1: non-IDR - Coded slice of a non-IDR picture	0	17024
231291	42: 1: non-IDR - Coded slice of a non-IDR picture	0	9415

No filter

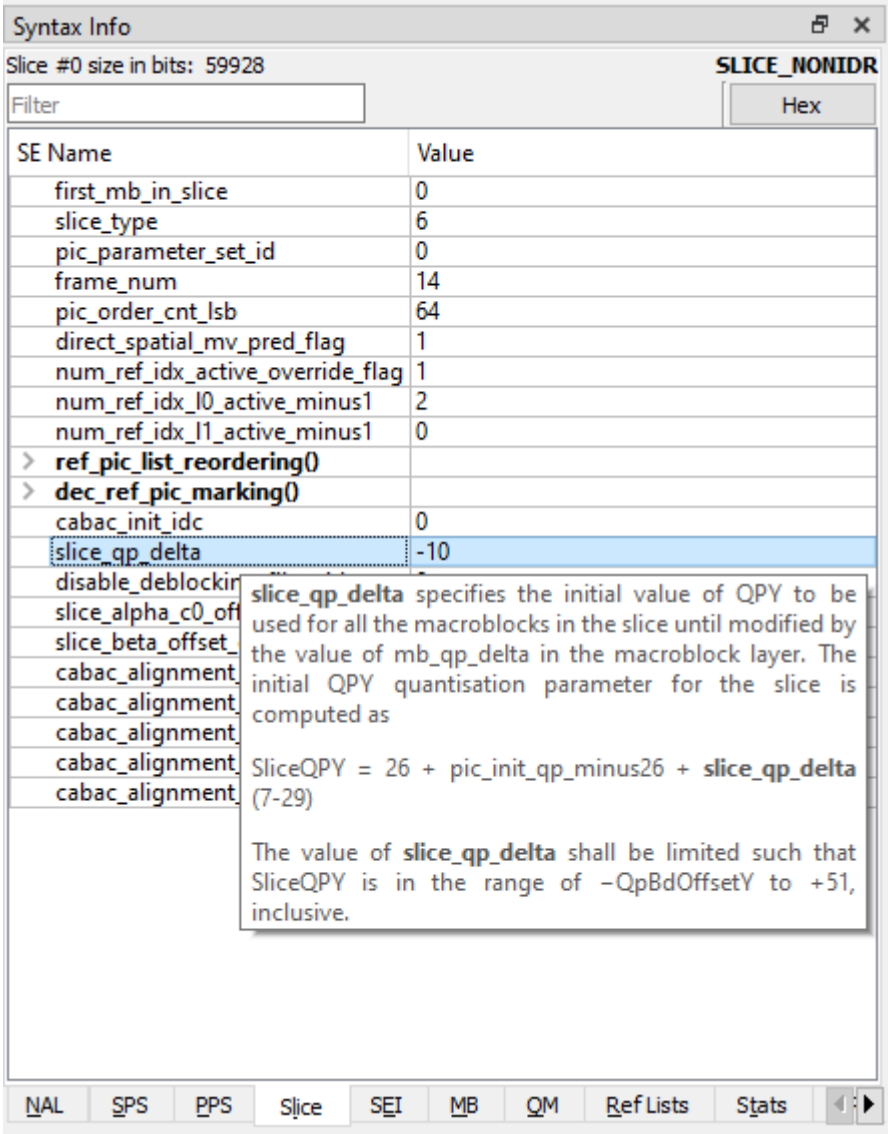
Extract selected units

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

Slice #0 size in bits: 115024 **SLICE_NONIDR**

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	5
pic_parameter_set_id	0
frame_num	13
pic_order_cnt_lsb	68
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	4
ref_pic_list_reordering()	
pred_weight_table()	
dec_ref_pic_marking()	
cabac_init_idc	0
slice_qp_delta	-10
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	-1
slice_beta_offset_div2	-1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

Slice #0 size in bits: 44544 **SLICE_NONIDR**

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	6
pic_parameter_set_id	0
frame_num	15
pic_order_cnt_lsb	62
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	0
num_ref_idx_l1_active_minus1	1
ref_pic_list_reordering()	
cabac_init_idc	0
slice_qp_delta	-10
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	-1
slice_beta_offset_div2	-1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

Syntax Info

Filter | Hex

Pos	R	V	SE Name	Value	Bits	Bits, %
0x000288ff:7	384	351	mb_skip_flag	0	0	0.00
0x000288ff:7	363	351	mb_type	0	2	4.00
0x00028900:1	260	214	coded_block_pattern	31	5	10.00
0x00028900:6	310	4	mb_qp_delta	0	1	2.00

mb_qp_delta can change the value of QPY in the macroblock layer. The decoded value of **mb_qp_delta** shall be in the range of −(26 + QpBdOffsetY / 2) to +(25 + QpBdOffsetY / 2), inclusive. **mb_qp_delta** shall be inferred to be equal to 0 when it is not present for any macroblock (including P_Skip and B_Skip macroblock types).

The value of QPY is derived as

QPY = ((QPY,PREV + **mb_qp_delta** + 52 + 2 * QpBdOffsetY) % (52 + QpBdOffsetY)) − QpBdOffsetY (7-36)

where QPY,PREV is the luma quantisation parameter, QPY, of the previous macroblock in decoding order in the current slice. For the first macroblock in the slice QPY,PREV is initially set equal to SliceQPY derived in Equation 7-29 at the start of each slice.

The value of QP'Y is derived as

QP'Y = QPY + QpBdOffsetY (7-37)

The variable TransformBypassModeFlag is derived as follows.

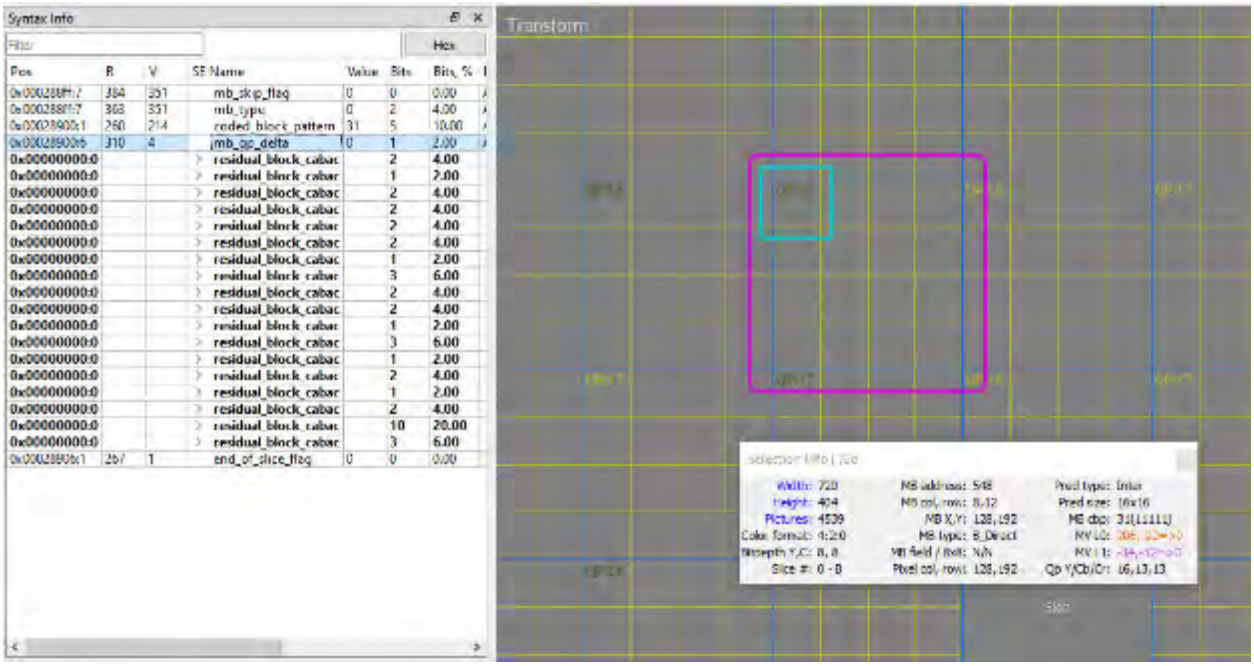
- If qpprime_y_zero_transform_bypass_flag is equal to 1 and QP'Y is equal to 0, TransformBypassModeFlag is set equal to 1.
- Otherwise (qpprime_y_zero_transform_bypass_flag is equal to 0 or QP'Y is not equal to 0),

TransformBypassModeFlag is set equal to 0.

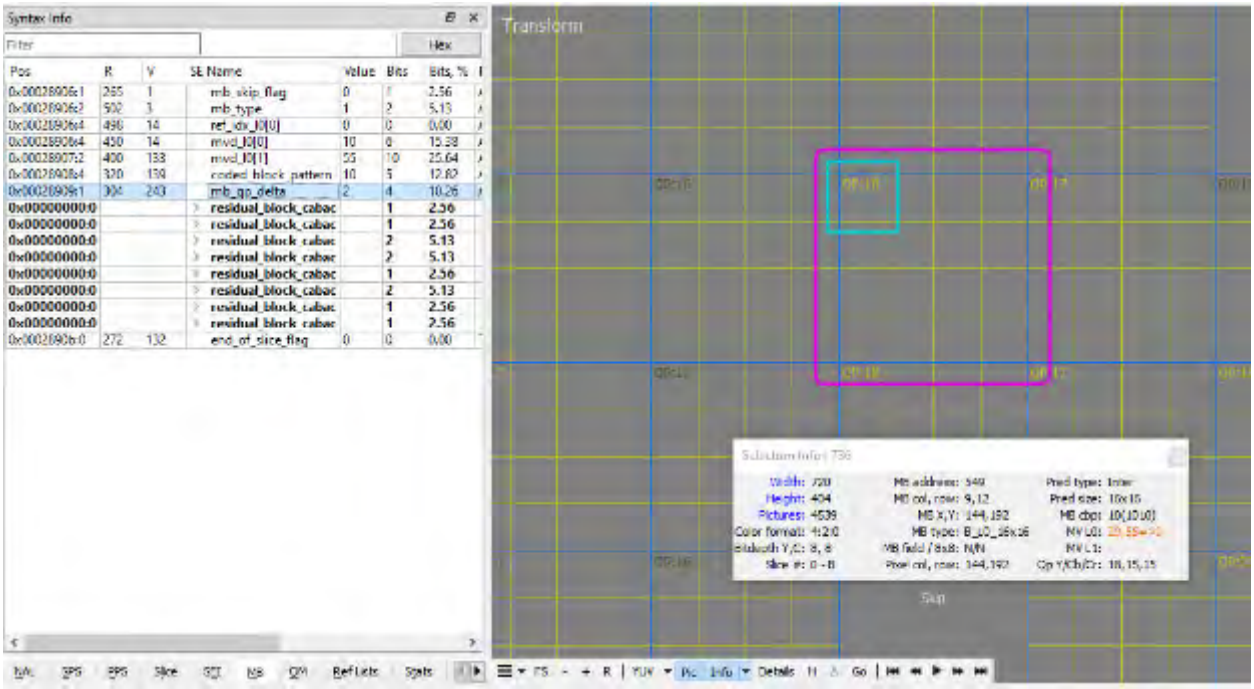
Selection Info | 736

Width:	720	MB address:	548	Pred type:	Inter
Height:	404	MB col, row:	8,12	Pred size:	16x16
Pictures:	4539	MB X,Y:	128,192	MB cbp:	31(11111)
Color format:	4:2:0	MB type:	B_Direct	MV L0:	206,-22=>0
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/N	MV L1:	-34,-52=>0
Slice #:	0 - B	Pixel col, row:	N/A	Qp Y/Cb/Cr:	16,13,13

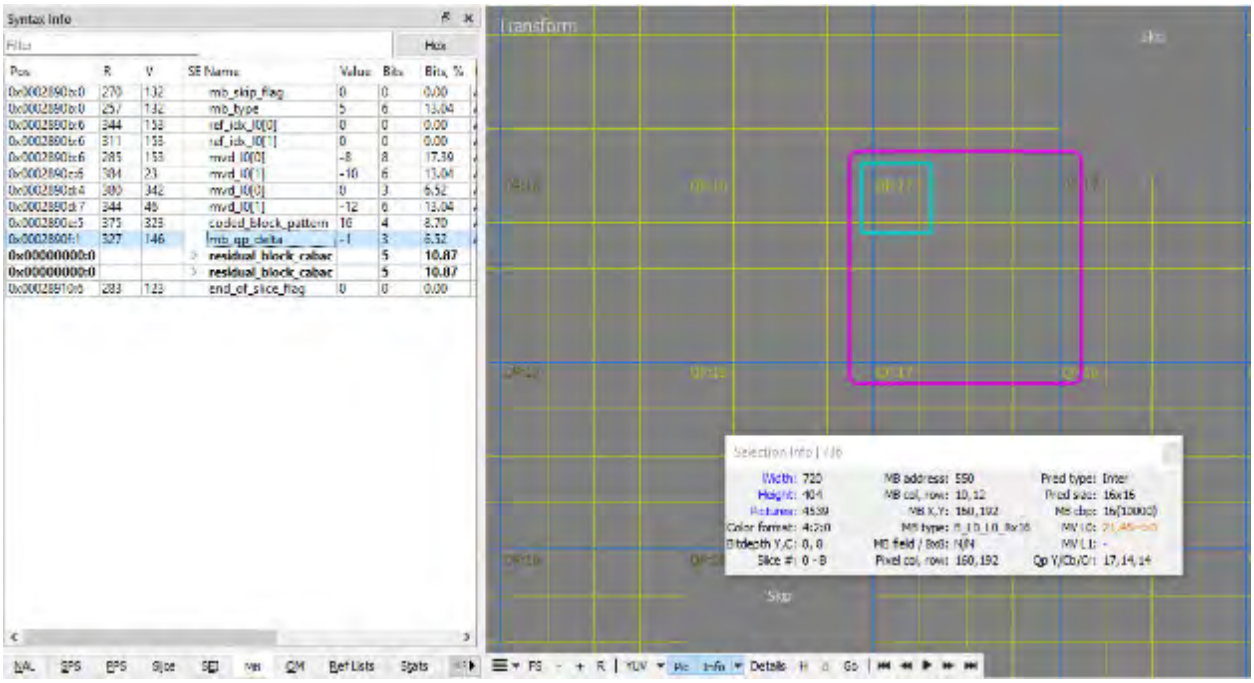
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.



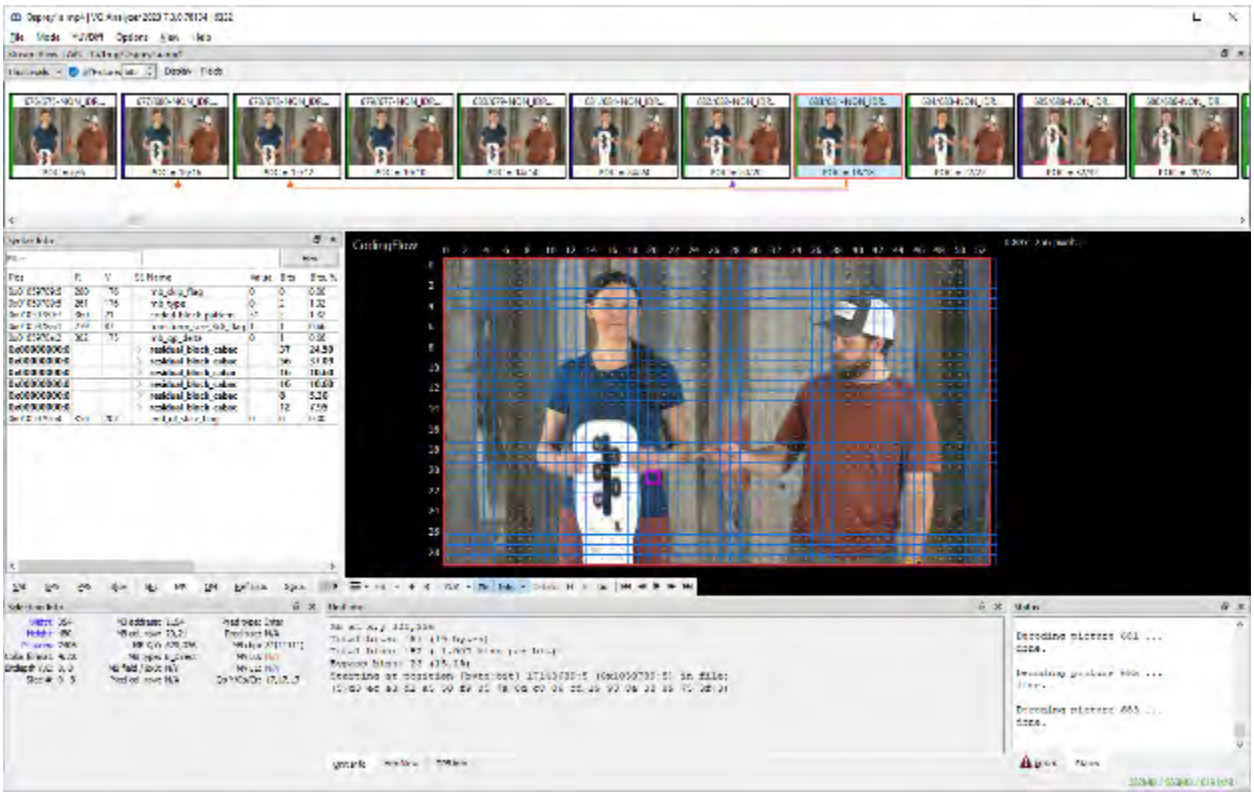
Screenshots of VQ Analyzer software analysis of trailer video available on Amazon Prime Video.

392. For another example, on information and belief, Amazon performs a method of encoding video in a manner that is substantially the same as described above when Amazon Prime Video contents are encoded to produce a bitstream that indicates it can be decoded by an H.265-compliant decoder, such that it is covered by claim 1 of the ’148 Patent.

Screenshots of Amazon Prime Video indicating “Codec: hevc.”

393. For another example, on information and belief, Amazon performs a method of encoding a video sequence in a manner that is covered by claim 1 of the ’148 Patent for Amazon.com advertisements, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 217.

394. When encoding Amazon.com advertisements, on information and belief, Amazon performs encoding a digital video sequence for use in a video coding application to produce an encoded video bit-stream representative of the digital video sequence, the digital video sequence comprising a number of frames, each frame of said sequence comprising an array of pixels divided into a plurality of blocks, each block comprising a certain number of said pixels, as demonstrated in the screenshots below using VQ Analyzer software.

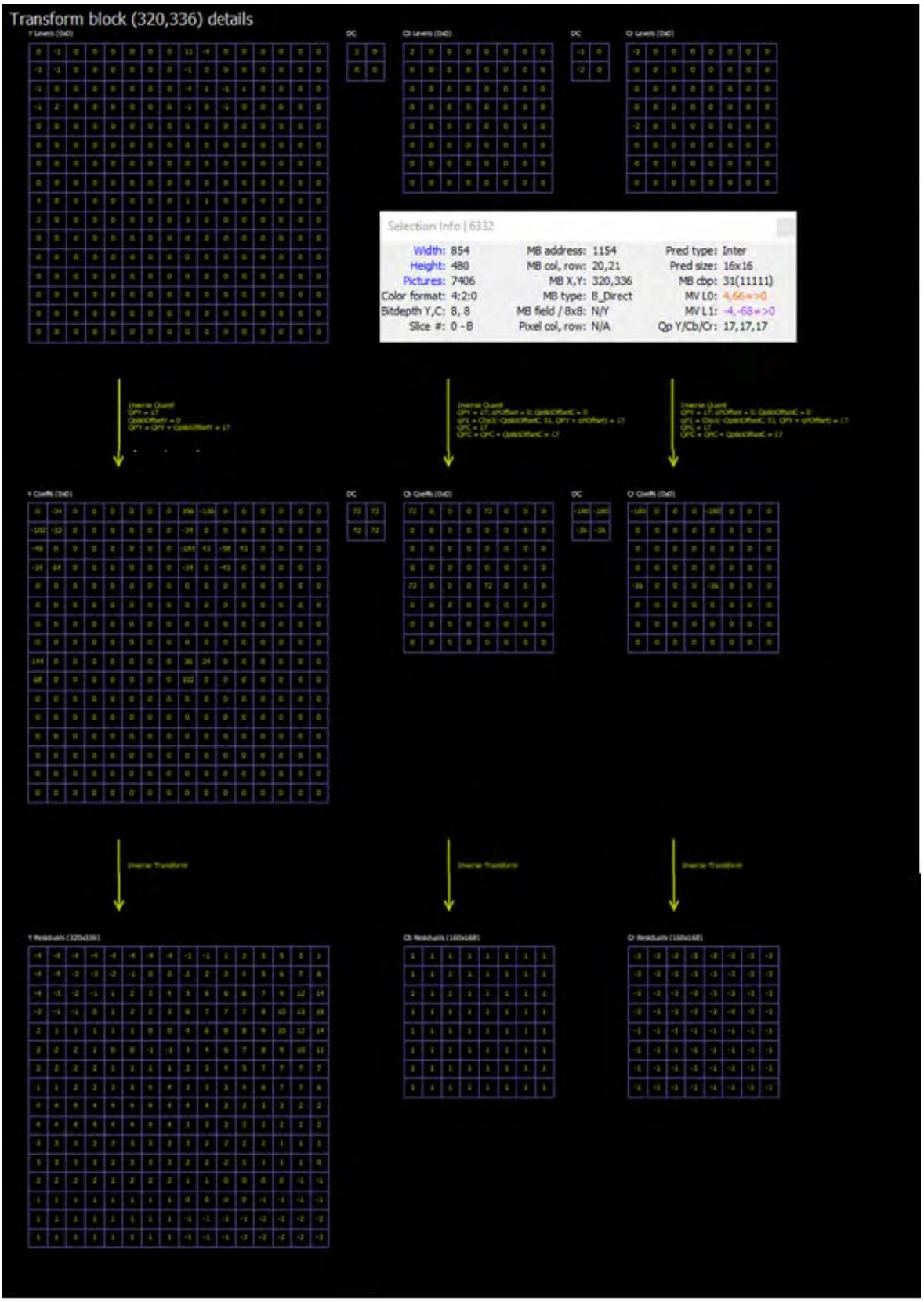


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

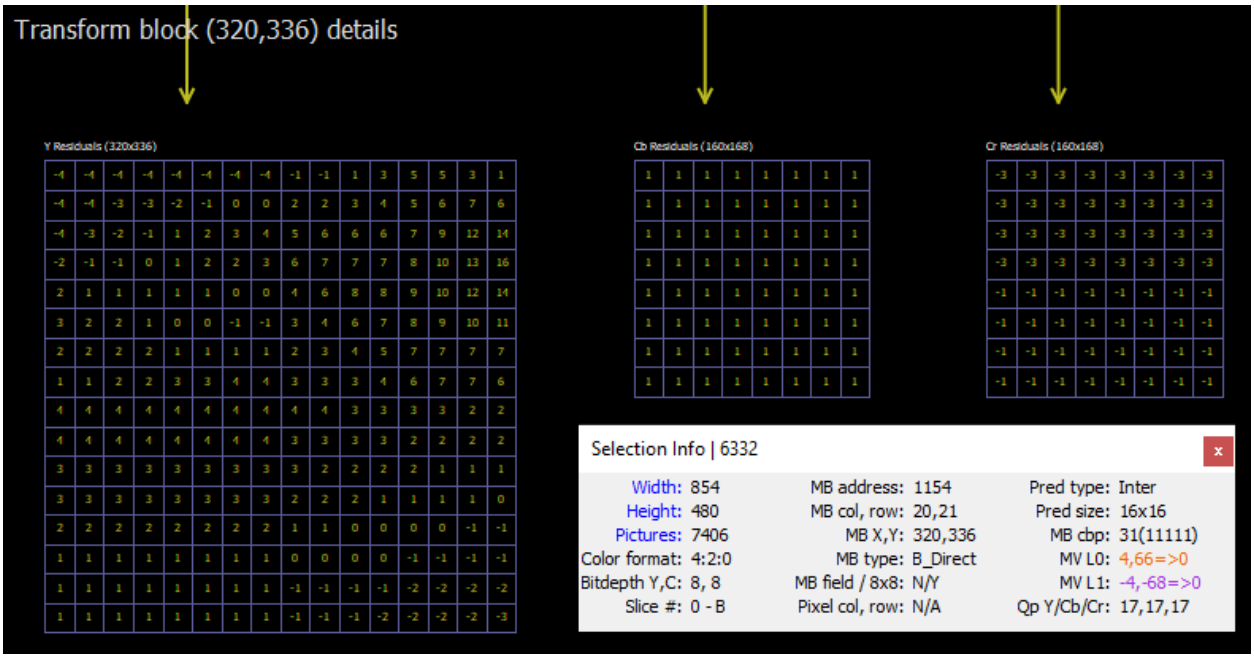
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

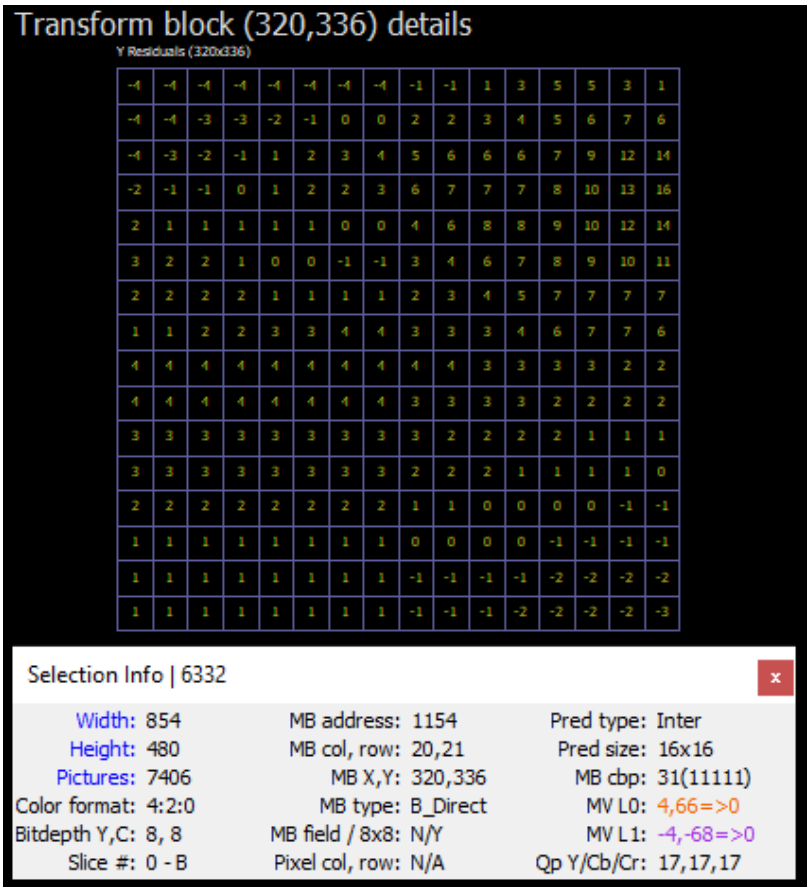
395. When encoding Amazon.com advertisements, on information and belief, Amazon performs encoding a frame of the digital video sequence by applying motion compensated prediction to blocks of pixels for producing corresponding blocks of prediction error values, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

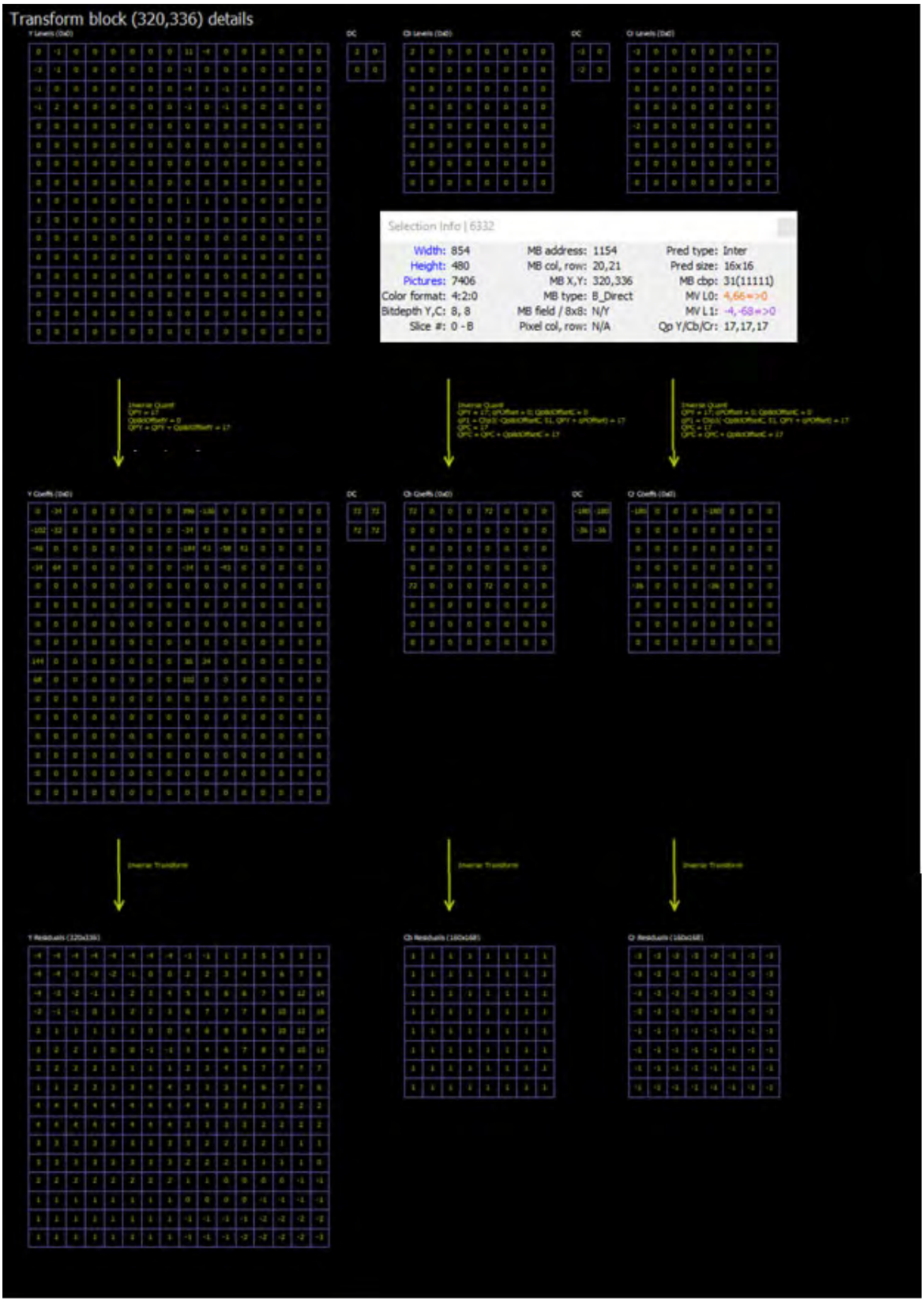


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

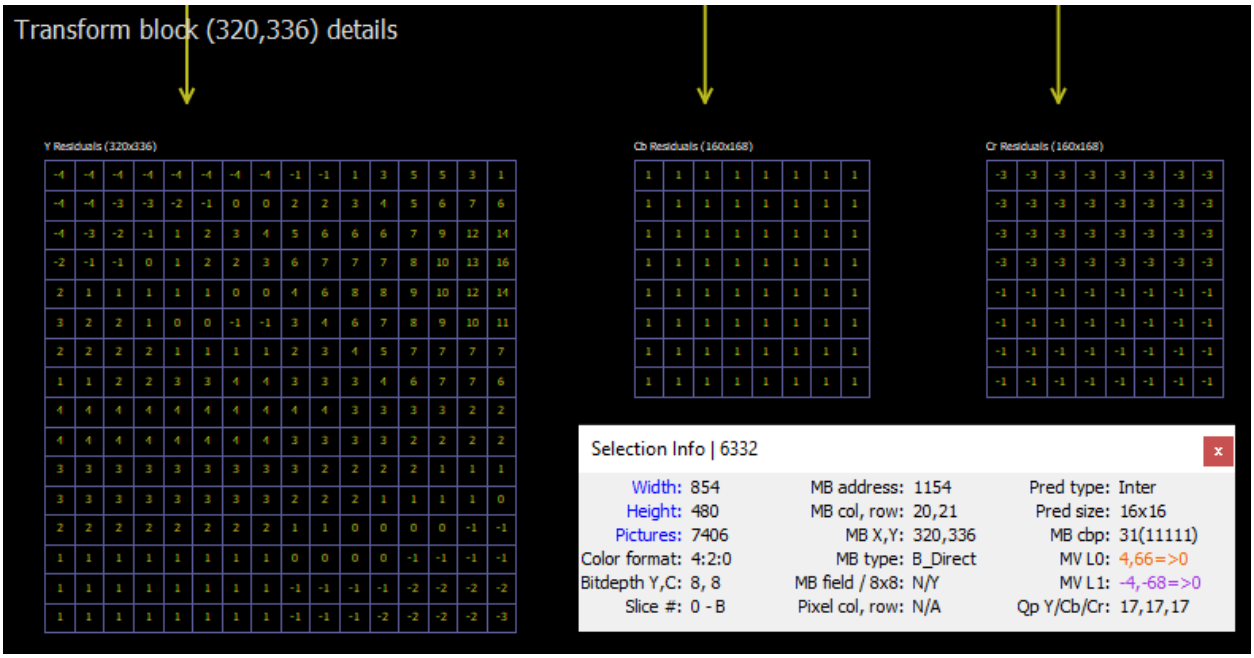


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

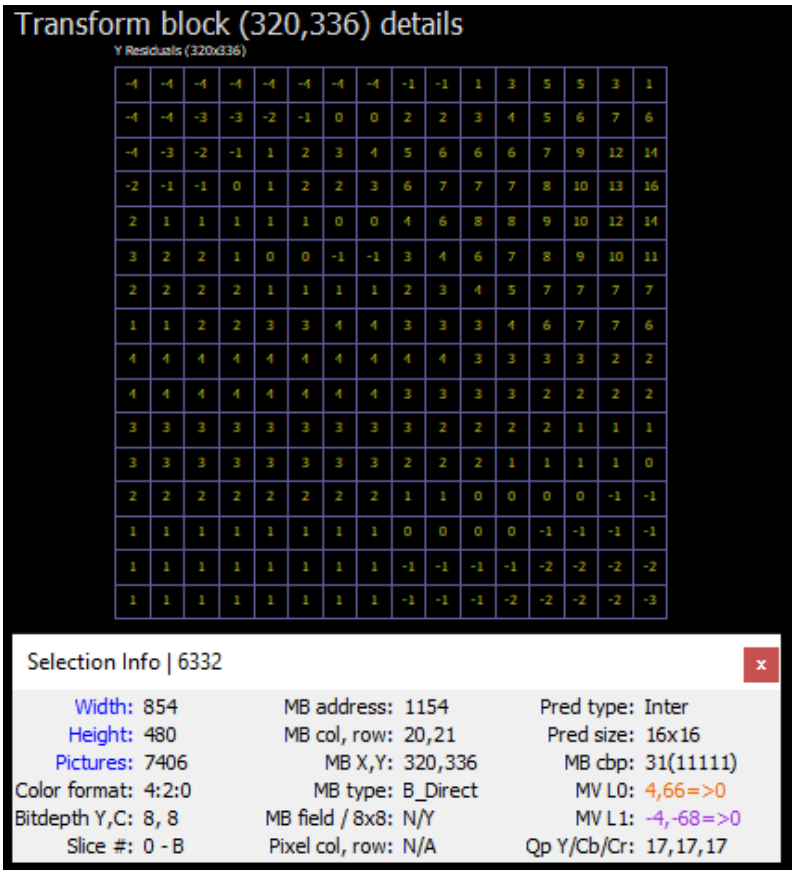
396. When encoding Amazon.com advertisements, on information and belief, Amazon performs applying a transform coding technique to said blocks of prediction error values to produce sets of transform coefficient values representative of said blocks of prediction error values, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

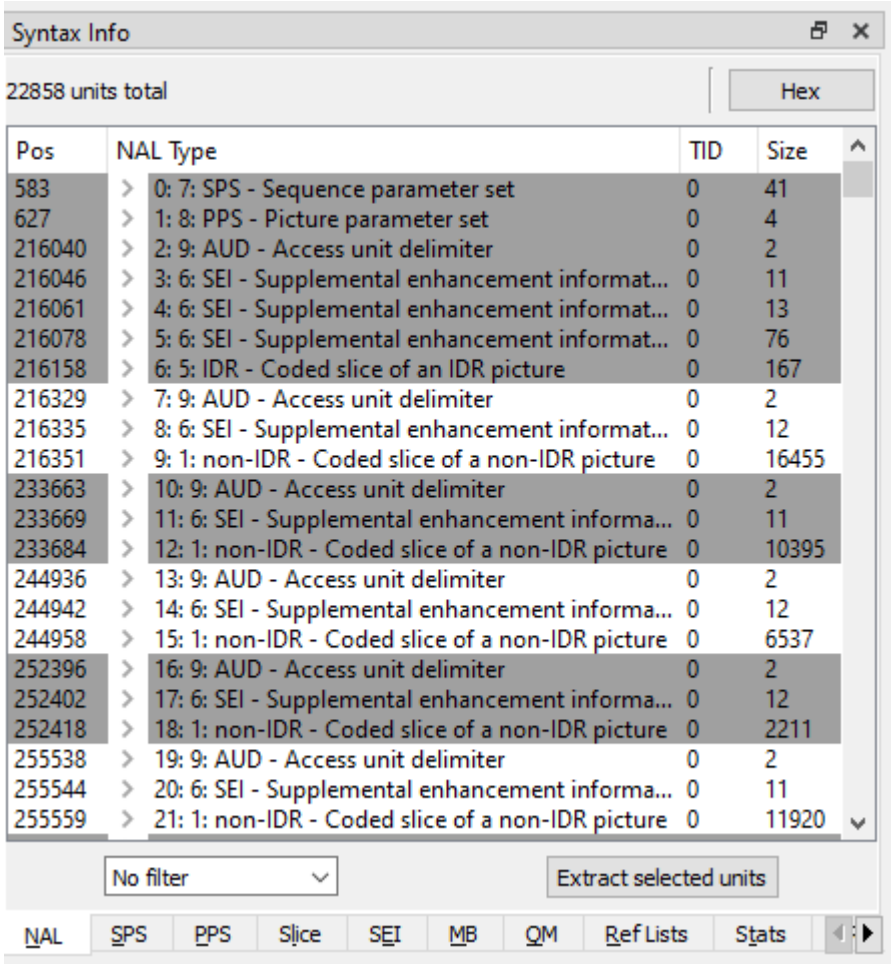


Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

397. When encoding Amazon.com advertisements, on information and belief, Amazon performs defining a default level of quantization for use in encoding of the digital video sequence to quantize the sets of transform coefficient values, as demonstrated in the screenshots below using VQ Analyzer software.



Pos	NAL Type	TID	Size
583	0: 7: SPS - Sequence parameter set	0	41
627	1: 8: PPS - Picture parameter set	0	4
216040	2: 9: AUD - Access unit delimiter	0	2
216046	3: 6: SEI - Supplemental enhancement informat...	0	11
216061	4: 6: SEI - Supplemental enhancement informat...	0	13
216078	5: 6: SEI - Supplemental enhancement informat...	0	76
216158	6: 5: IDR - Coded slice of an IDR picture	0	167
216329	7: 9: AUD - Access unit delimiter	0	2
216335	8: 6: SEI - Supplemental enhancement informat...	0	12
216351	9: 1: non-IDR - Coded slice of a non-IDR picture	0	16455
233663	10: 9: AUD - Access unit delimiter	0	2
233669	11: 6: SEI - Supplemental enhancement informa...	0	11
233684	12: 1: non-IDR - Coded slice of a non-IDR picture	0	10395
244936	13: 9: AUD - Access unit delimiter	0	2
244942	14: 6: SEI - Supplemental enhancement informa...	0	12
244958	15: 1: non-IDR - Coded slice of a non-IDR picture	0	6537
252396	16: 9: AUD - Access unit delimiter	0	2
252402	17: 6: SEI - Supplemental enhancement informa...	0	12
252418	18: 1: non-IDR - Coded slice of a non-IDR picture	0	2211
255538	19: 9: AUD - Access unit delimiter	0	2
255544	20: 6: SEI - Supplemental enhancement informa...	0	11
255559	21: 1: non-IDR - Coded slice of a non-IDR picture	0	11920

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info

22858 units total | Hex

Pos	NAL Type	TID	Size
7095...	2265: 6: SEI - Supplemental enhancement infor...	0	11
7095...	2266: 1: non-IDR - Coded slice of a non-IDR pic...	0	31118
7127...	2267: 9: AUD - Access unit delimiter	0	2
7127...	2268: 6: SEI - Supplemental enhancement infor...	0	11
7127...	2269: 1: non-IDR - Coded slice of a non-IDR pic...	0	10669
7138...	2270: 9: AUD - Access unit delimiter	0	2
7138...	2271: 6: SEI - Supplemental enhancement infor...	0	12
7138...	2272: 1: non-IDR - Coded slice of a non-IDR pic...	0	6878
7146...	2273: 9: AUD - Access unit delimiter	0	2
7146...	2274: 6: SEI - Supplemental enhancement infor...	0	12
7146...	2275: 1: non-IDR - Coded slice of a non-IDR pic...	0	8125
7155...	2276: 9: AUD - Access unit delimiter	0	2
7155...	2277: 6: SEI - Supplemental enhancement infor...	0	11
7155...	2278: 1: non-IDR - Coded slice of a non-IDR pic...	0	48909
7205...	2279: 9: AUD - Access unit delimiter	0	2
7205...	2280: 6: SEI - Supplemental enhancement infor...	0	11
7205...	2281: 1: non-IDR - Coded slice of a non-IDR pic...	0	15442
7221...	2282: 9: AUD - Access unit delimiter	0	2
7221...	2283: 6: SEI - Supplemental enhancement infor...	0	12
7221...	2284: 1: non-IDR - Coded slice of a non-IDR pic...	0	9130
7231...	2285: 9: AUD - Access unit delimiter	0	2
7231...	2286: 6: SEI - Supplemental enhancement infor...	0	12

No filter | Extract selected units

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info

Filter | Hex

SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	1
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	2
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	
deblocking_filter_control_pr	
constrained_intra_pred_flag	
redundant_pic_cnt_present_	
transform_8x8_mode_flag	
pic_scaling_matrix_present_	
second_chroma_qp_index_c	
rbsp_stop_one_bit	
rbsp_alignment_zero_bit	
rbsp_alignment_zero_bit	0

pic_init_qp_minus26 specifies the initial value minus 26 of SliceQPY for each slice. The initial value is modified at the slice layer when a non-zero value of slice_qp_delta is decoded, and is modified further when a non-zero value of mb_qp_delta is decoded at the macroblock layer. The value of **pic_init_qp_minus26** shall be in the range of −(26 + QpBdOffsetY) to +25, inclusive.

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info | 6332

Slice #0 size in bits: 55024 SLICE_NONIDR

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	6
pic_parameter_set_id	0
frame_num	7
pic_order_cnt_lsb	18
delta_pic_order_cnt_bottom	0
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	1
num_ref_idx_l1_active_minus1	0
ref_pic_list_reordering()	
cabac_init_idc	0
slice_qp_delta	-9
disable_deblockin	
slice_alpha_c0_of	
slice_beta_offset_	
cabac_alignment	
cabac_alignment	
cabac_alignment	
cabac_alignment	
cabac_alignment	
cabac_alignment	

slice_qp_delta specifies the initial value of QPY to be used for all the macroblocks in the slice until modified by the value of mb_qp_delta in the macroblock layer. The initial QPY quantisation parameter for the slice is computed as

SliceQPY = 26 + pic_init_qp_minus26 + **slice_qp_delta** (7-29)

The value of **slice_qp_delta** shall be limited such that SliceQPY is in the range of −QpBdOffsetY to +51, inclusive.

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info

Slice #0 size in bits: 85352 **SLICE_NONIDR**

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	6
pic_parameter_set_id	0
frame_num	6
pic_order_cnt_lsb	20
delta_pic_order_cnt_bottom	0
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	1
num_ref_idx_l1_active_minus1	0
> **ref_pic_list_reordering()**	
> **dec_ref_pic_marking()**	
cabac_init_idc	0
slice_qp_delta	-10
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	0
slice_beta_offset_div2	0
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info

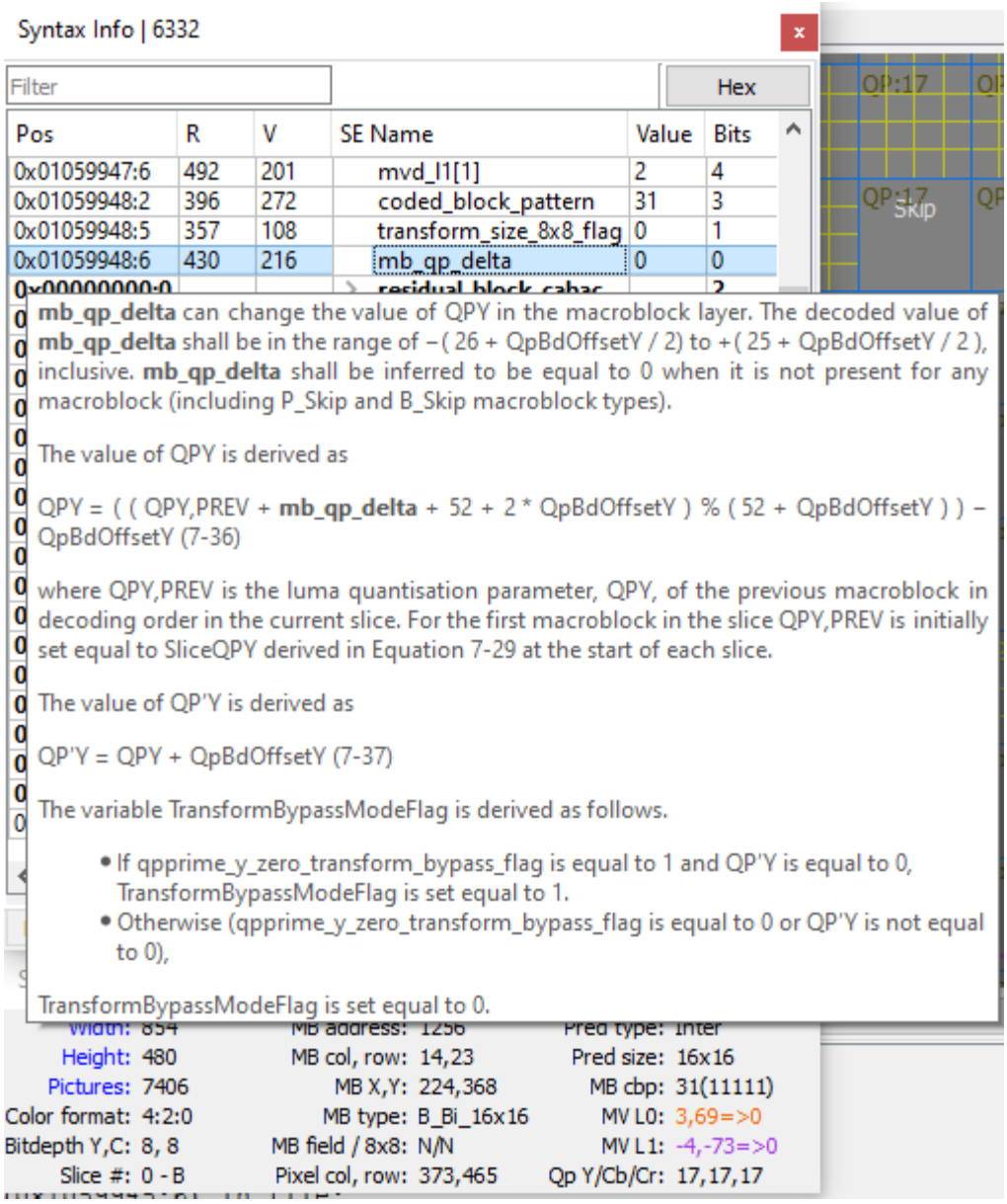
Slice #0 size in bits: 65000 SLICE_NONIDR

Filter Hex

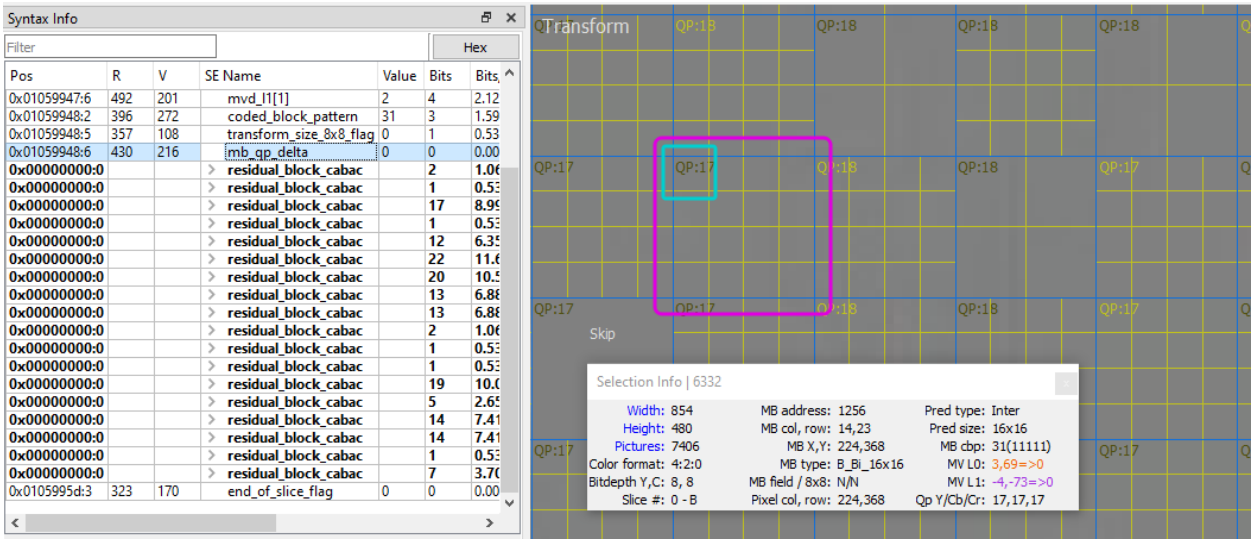
SE Name	Value
first_mb_in_slice	0
slice_type	6
pic_parameter_set_id	0
frame_num	7
pic_order_cnt_lsb	22
delta_pic_order_cnt_bottom	0
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	1
num_ref_idx_l1_active_minus1	0
ref_pic_list_reordering()	
cabac_init_idc	0
slice_qp_delta	-9
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	0
slice_beta_offset_div2	0
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

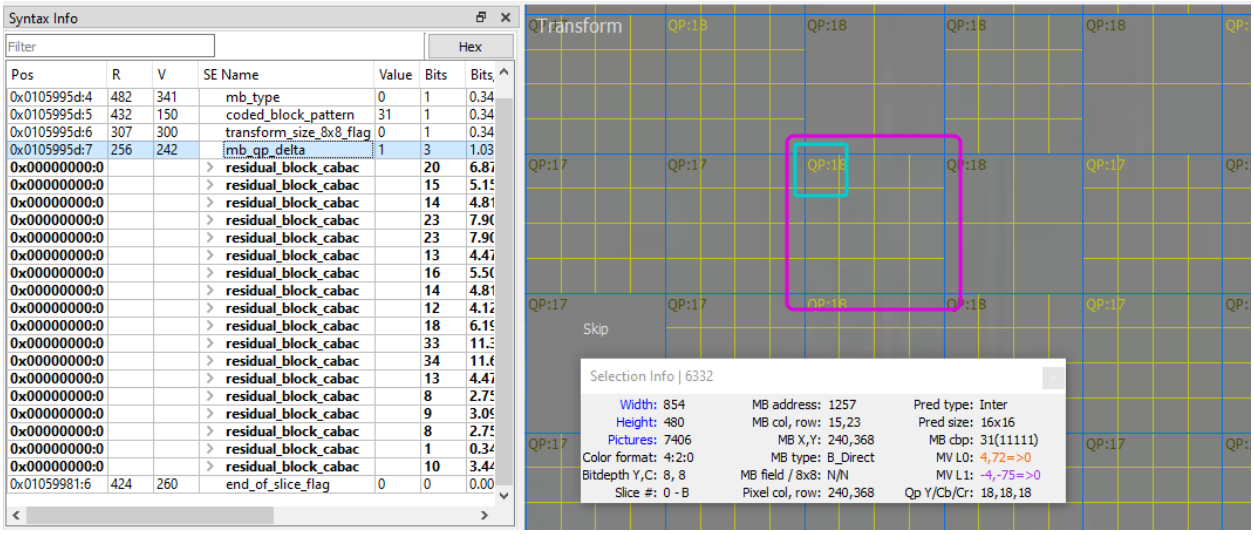
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



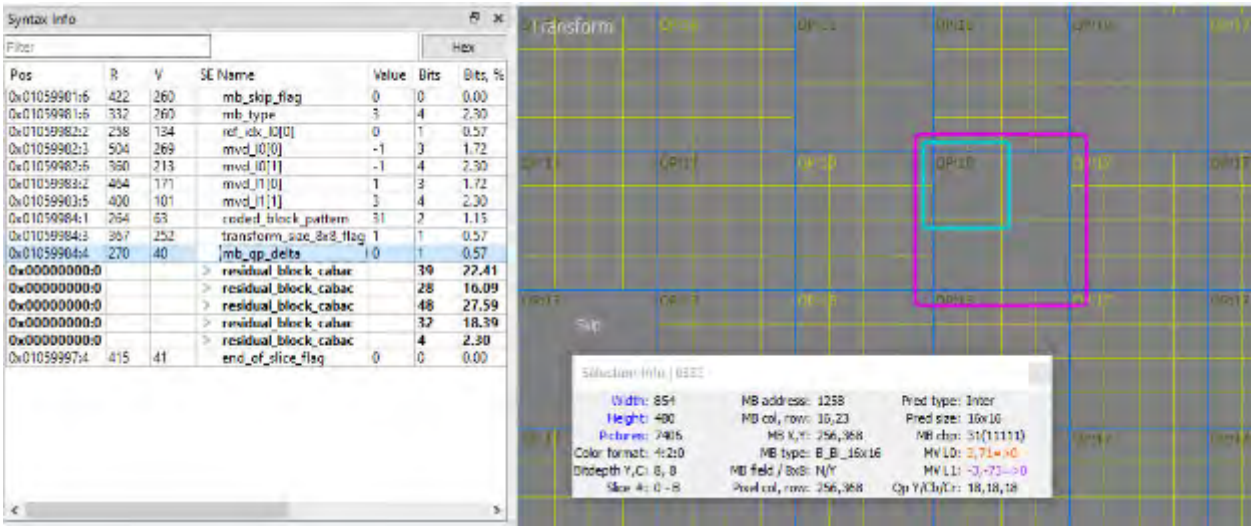
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



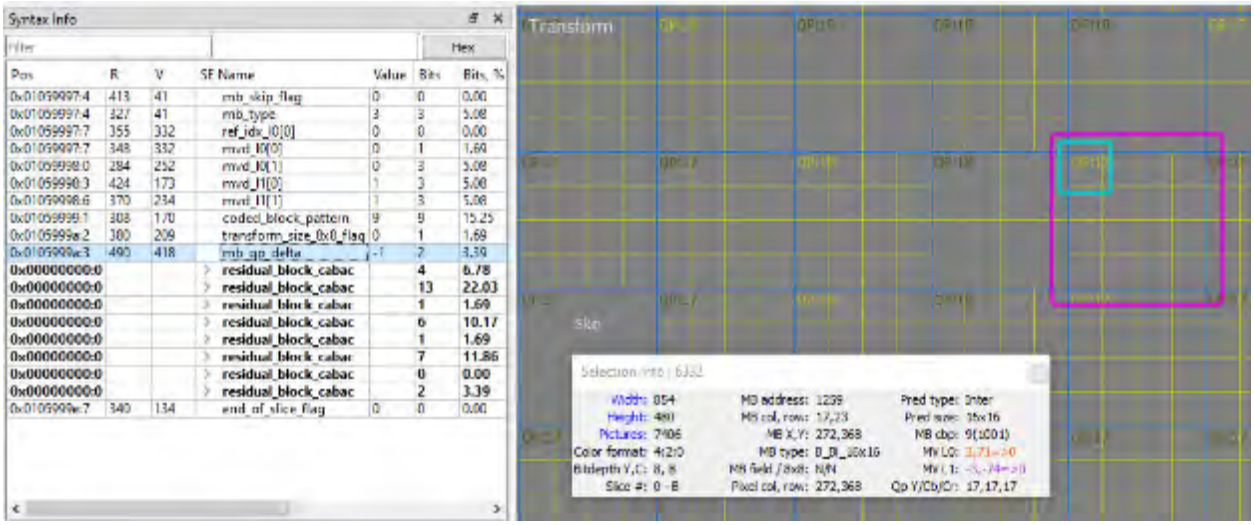
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

398. When encoding Amazon.com advertisements, on information and belief, Amazon performs providing an indication of the default level of quantization to a decoding process, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info

22858 units total | Hex

Pos	NAL Type	TID	Size
583	0: 7: SPS - Sequence parameter set	0	41
627	1: 8: PPS - Picture parameter set	0	4
216040	2: 9: AUD - Access unit delimiter	0	2
216046	3: 6: SEI - Supplemental enhancement informat...	0	11
216061	4: 6: SEI - Supplemental enhancement informat...	0	13
216078	5: 6: SEI - Supplemental enhancement informat...	0	76
216158	6: 5: IDR - Coded slice of an IDR picture	0	167
216329	7: 9: AUD - Access unit delimiter	0	2
216335	8: 6: SEI - Supplemental enhancement informat...	0	12
216351	9: 1: non-IDR - Coded slice of a non-IDR picture	0	16455
233663	10: 9: AUD - Access unit delimiter	0	2
233669	11: 6: SEI - Supplemental enhancement informa...	0	11
233684	12: 1: non-IDR - Coded slice of a non-IDR picture	0	10395
244936	13: 9: AUD - Access unit delimiter	0	2
244942	14: 6: SEI - Supplemental enhancement informa...	0	12
244958	15: 1: non-IDR - Coded slice of a non-IDR picture	0	6537
252396	16: 9: AUD - Access unit delimiter	0	2
252402	17: 6: SEI - Supplemental enhancement informa...	0	12
252418	18: 1: non-IDR - Coded slice of a non-IDR picture	0	2211
255538	19: 9: AUD - Access unit delimiter	0	2
255544	20: 6: SEI - Supplemental enhancement informa...	0	11
255559	21: 1: non-IDR - Coded slice of a non-IDR picture	0	11920

No filter | Extract selected units

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info

22858 units total | Hex

Pos	NAL Type	TID	Size
7095...	2265: 6: SEI - Supplemental enhancement infor...	0	11
7095...	2266: 1: non-IDR - Coded slice of a non-IDR pic...	0	31118
7127...	2267: 9: AUD - Access unit delimiter	0	2
7127...	2268: 6: SEI - Supplemental enhancement infor...	0	11
7127...	2269: 1: non-IDR - Coded slice of a non-IDR pic...	0	10669
7138...	2270: 9: AUD - Access unit delimiter	0	2
7138...	2271: 6: SEI - Supplemental enhancement infor...	0	12
7138...	2272: 1: non-IDR - Coded slice of a non-IDR pic...	0	6878
7146...	2273: 9: AUD - Access unit delimiter	0	2
7146...	2274: 6: SEI - Supplemental enhancement infor...	0	12
7146...	2275: 1: non-IDR - Coded slice of a non-IDR pic...	0	8125
7155...	2276: 9: AUD - Access unit delimiter	0	2
7155...	2277: 6: SEI - Supplemental enhancement infor...	0	11
7155...	2278: 1: non-IDR - Coded slice of a non-IDR pic...	0	48909
7205...	2279: 9: AUD - Access unit delimiter	0	2
7205...	2280: 6: SEI - Supplemental enhancement infor...	0	11
7205...	2281: 1: non-IDR - Coded slice of a non-IDR pic...	0	15442
7221...	2282: 9: AUD - Access unit delimiter	0	2
7221...	2283: 6: SEI - Supplemental enhancement infor...	0	12
7221...	2284: 1: non-IDR - Coded slice of a non-IDR pic...	0	9130
7231...	2285: 9: AUD - Access unit delimiter	0	2
7231...	2286: 6: SEI - Supplemental enhancement infor...	0	12

No filter | Extract selected units

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info

Filter | Hex

SE Name	Value
pic_parameter_set_id	0
seq_parameter_set_id	0
entropy_coding_mode_flag	1
bottom_field_pic_order_in_frame_present_flag	1
num_slice_groups_minus1	0
num_ref_idx_l0_active_minus1	2
num_ref_idx_l1_active_minus1	0
weighted_pred_flag	1
weighted_bipred_idc	2
pic_init_qp_minus26	0
pic_init_qs_minus26	0
chroma_qp_index_offset	
deblocking_filter_control_pr	
constrained_intra_pred_flag	
redundant_pic_cnt_present_	
transform_8x8_mode_flag	
pic_scaling_matrix_present_	
second_chroma_qp_index_c	
rbsp_stop_one_bit	
rbsp_alignment_zero_bit	
rbsp_alignment_zero_bit	0

pic_init_qp_minus26 specifies the initial value minus 26 of SliceQPY for each slice. The initial value is modified at the slice layer when a non-zero value of slice_qp_delta is decoded, and is modified further when a non-zero value of mb_qp_delta is decoded at the macroblock layer. The value of **pic_init_qp_minus26** shall be in the range of −(26 + QpBdOffsetY) to +25, inclusive.

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info | 6332

Slice #0 size in bits: 55024 SLICE_NONIDR

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	6
pic_parameter_set_id	0
frame_num	7
pic_order_cnt_lsb	18
delta_pic_order_cnt_bottom	0
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	1
num_ref_idx_l1_active_minus1	0
ref_pic_list_reordering()	
cabac_init_idc	0
slice_qp_delta	-9
disable_deblockin	
slice_alpha_c0_of	
slice_beta_offset_	
cabac_alignment	
cabac_alignment	
cabac_alignment	
cabac_alignment	
cabac_alignment	
cabac_alignment	

slice_qp_delta specifies the initial value of QPY to be used for all the macroblocks in the slice until modified by the value of mb_qp_delta in the macroblock layer. The initial QPY quantisation parameter for the slice is computed as

SliceQPY = 26 + pic_init_qp_minus26 + **slice_qp_delta** (7-29)

The value of **slice_qp_delta** shall be limited such that SliceQPY is in the range of −QpBdOffsetY to +51, inclusive.

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info

Slice #0 size in bits: 85352 **SLICE_NONIDR**

Filter Hex

SE Name	Value
first_mb_in_slice	0
slice_type	6
pic_parameter_set_id	0
frame_num	6
pic_order_cnt_lsb	20
delta_pic_order_cnt_bottom	0
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	1
num_ref_idx_l1_active_minus1	0
ref_pic_list_reordering()	
dec_ref_pic_marking()	
cabac_init_idc	0
slice_qp_delta	-10
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	0
slice_beta_offset_div2	0
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

Syntax Info

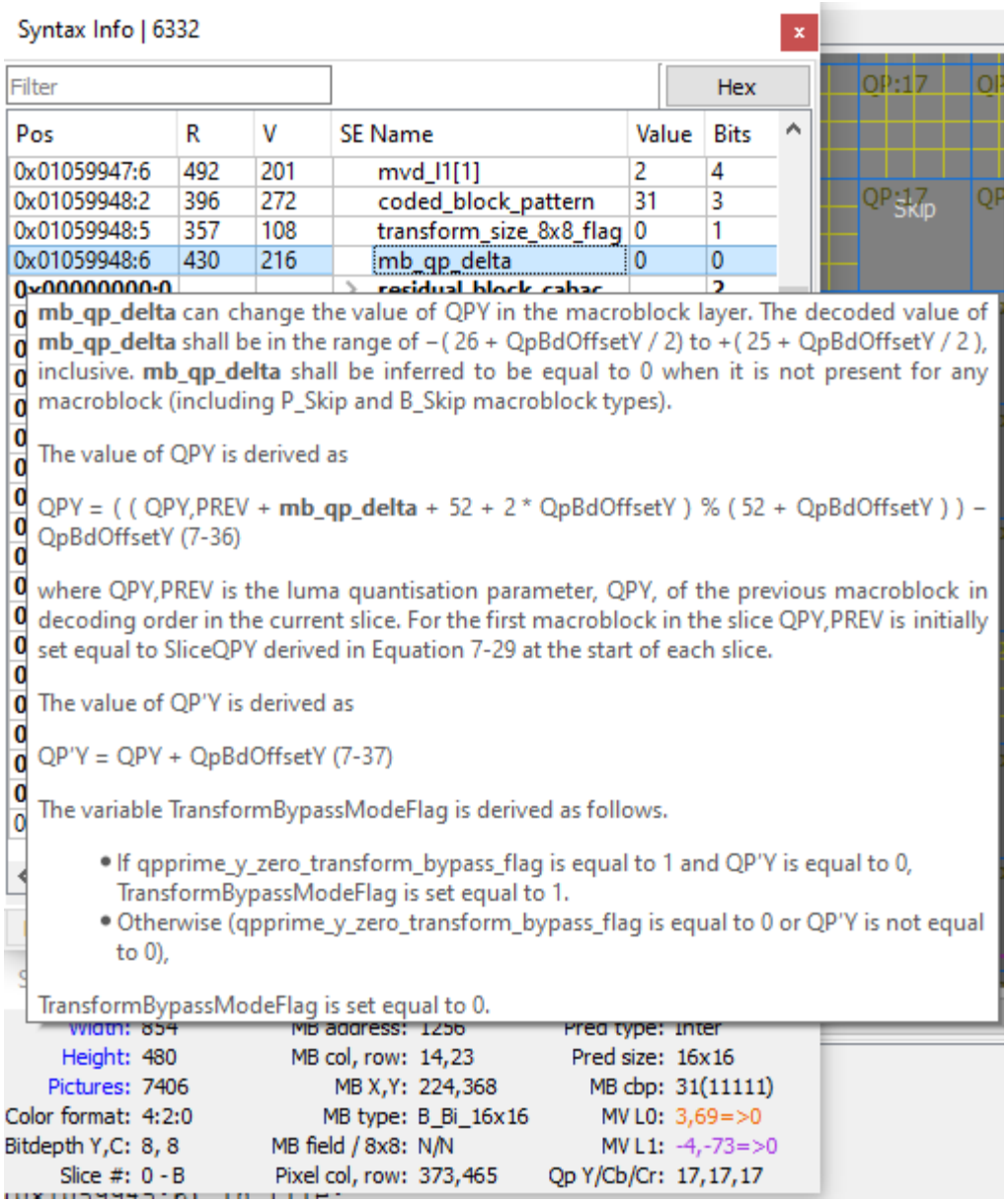
Slice #0 size in bits: 65000 SLICE_NONIDR

Filter Hex

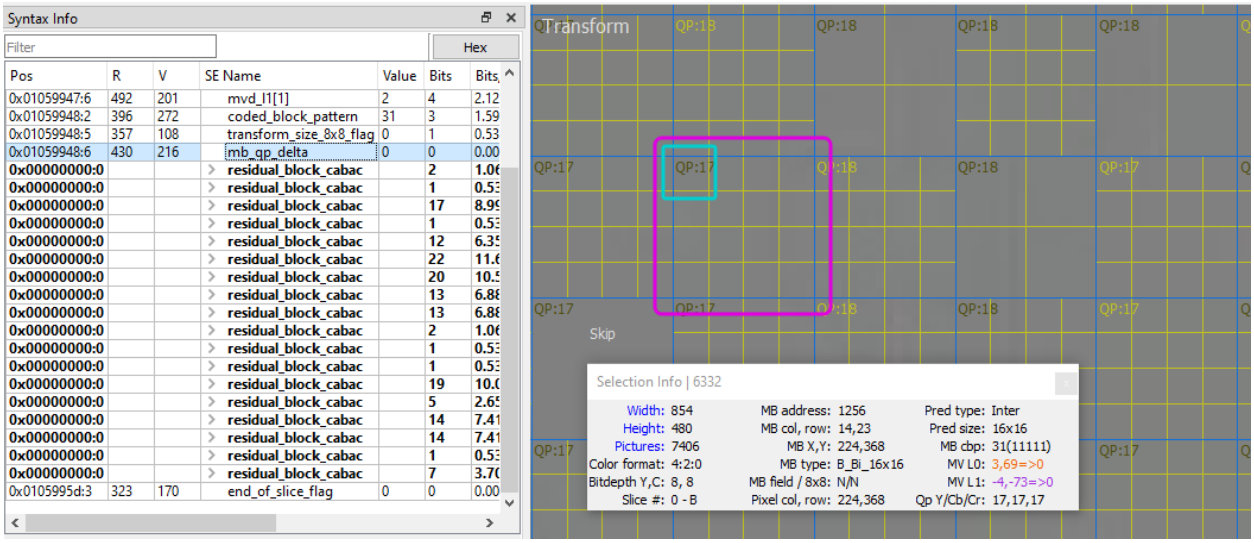
SE Name	Value
first_mb_in_slice	0
slice_type	6
pic_parameter_set_id	0
frame_num	7
pic_order_cnt_lsb	22
delta_pic_order_cnt_bottom	0
direct_spatial_mv_pred_flag	1
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	1
num_ref_idx_l1_active_minus1	0
ref_pic_list_reordering()	
cabac_init_idc	0
slice_qp_delta	-9
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	0
slice_beta_offset_div2	0
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

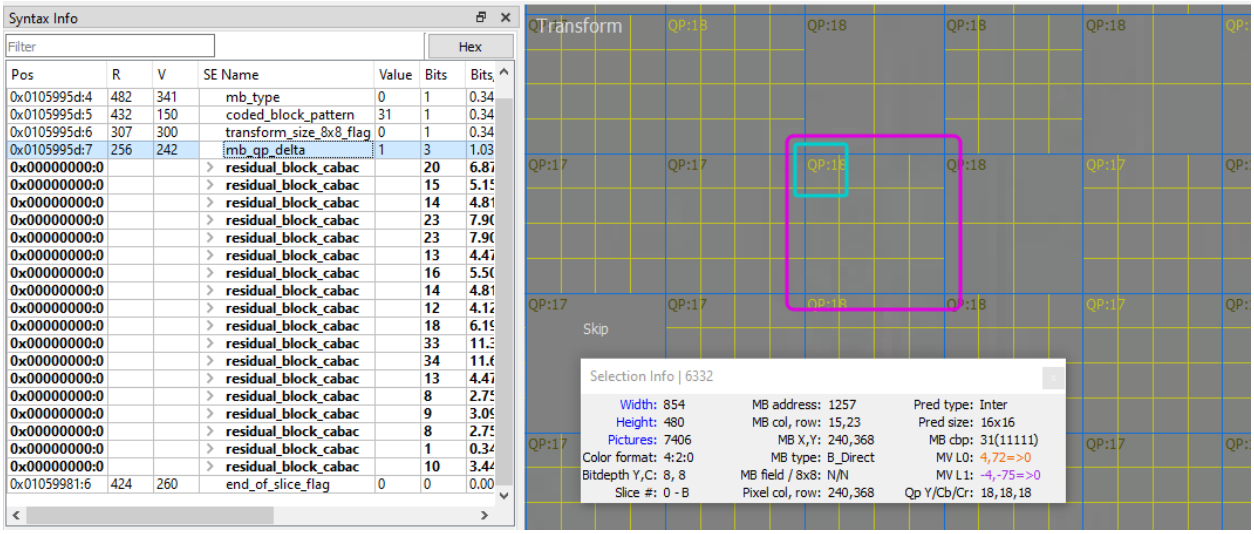
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



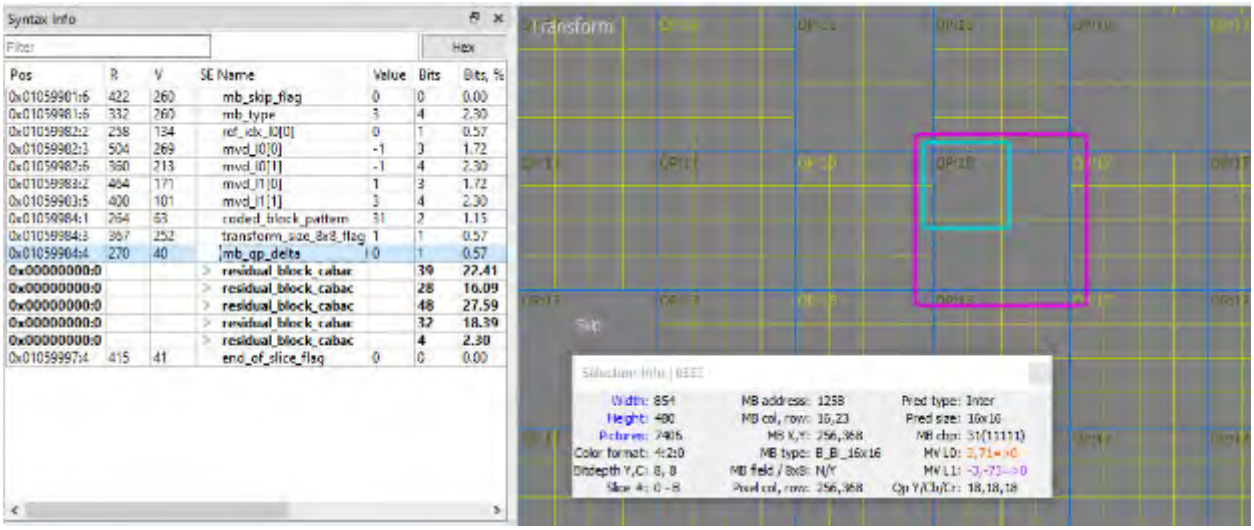
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



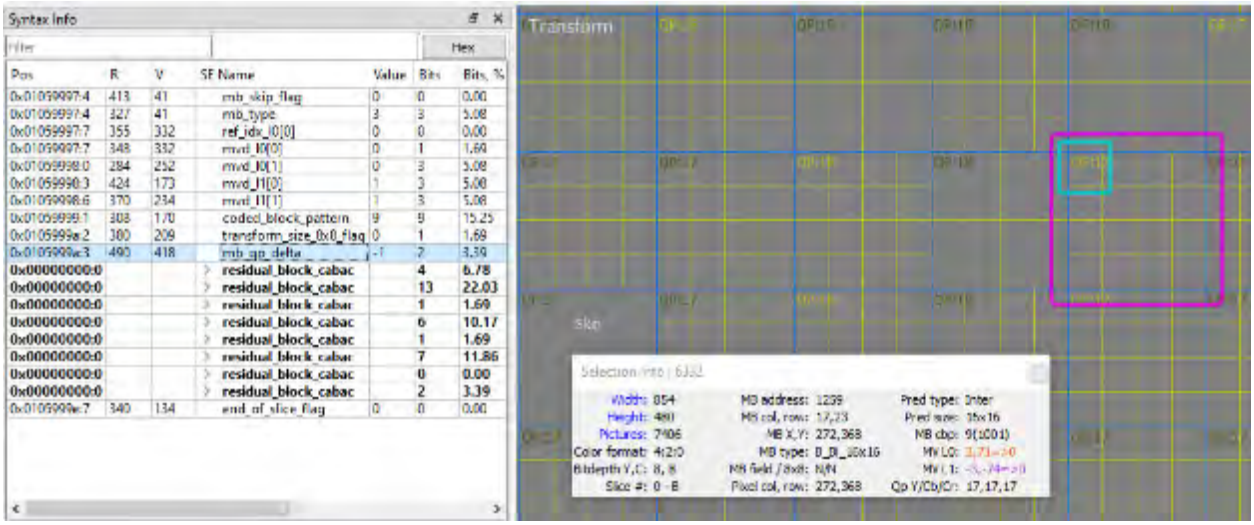
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.



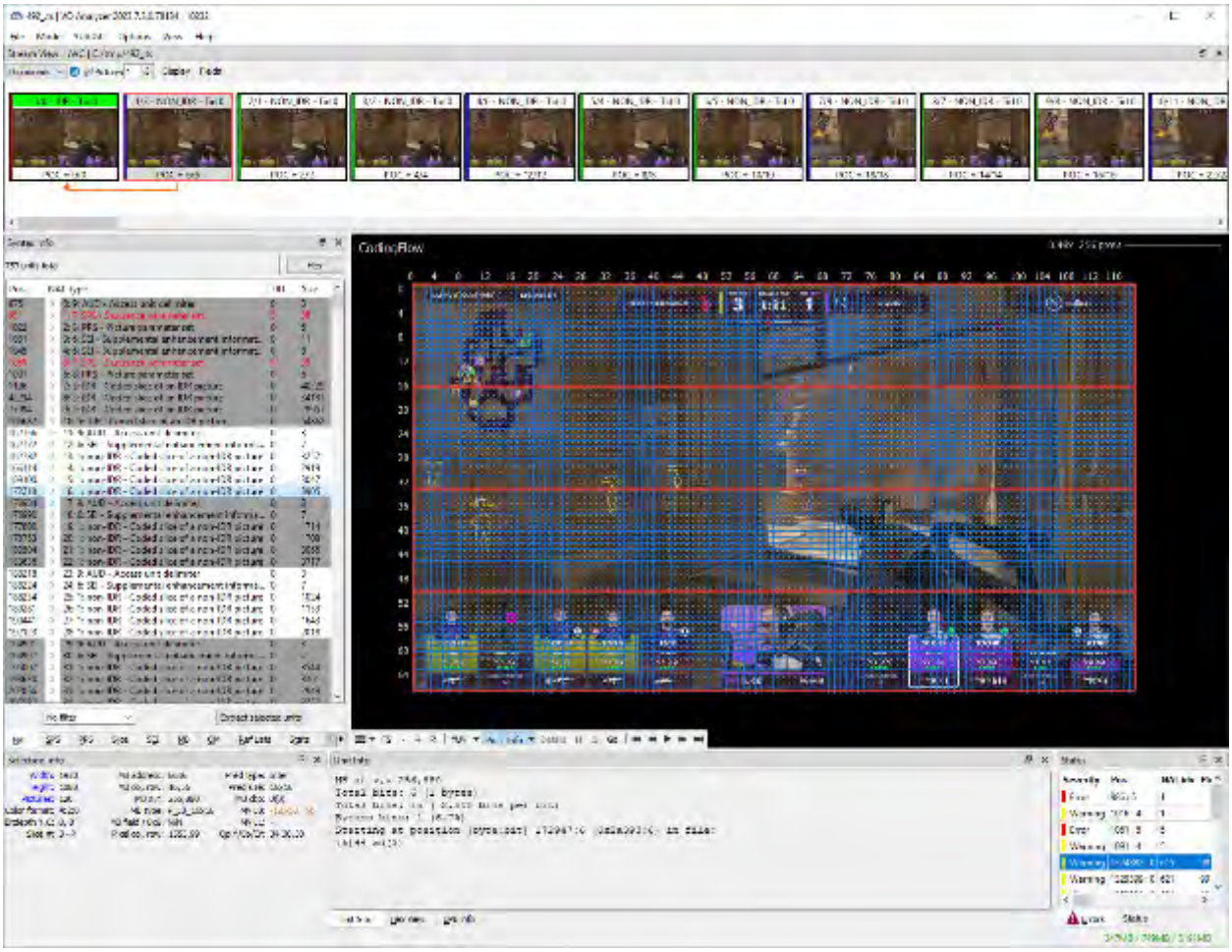
Screenshots of VQ Analyzer software analysis of video advertisements available on Amazon.com.

399. For another example, on information and belief, Amazon performs a method of encoding video in a manner that is substantially the same as described above when Amazon.com videos are encoded to produce a bitstream that indicates it can be decoded by an H.265-compliant decoder, such that it is covered by claim 1 of the ’148 Patent.

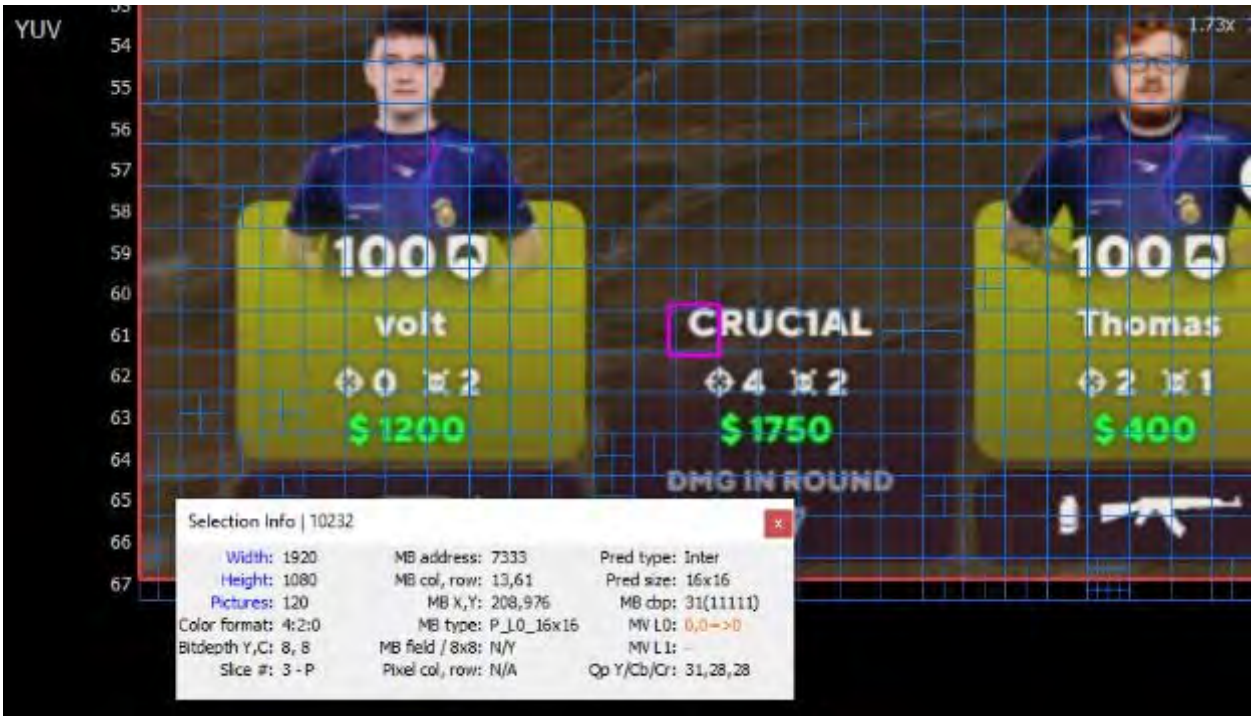
400. For another example, on information and belief, Amazon performs a method of encoding a video sequence in a manner that is covered by claim 1 of the ’148 Patent for Twitch.tv

video, as demonstrated in the screenshots using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.264-compliant decoder. *See supra* at paragraph 222.

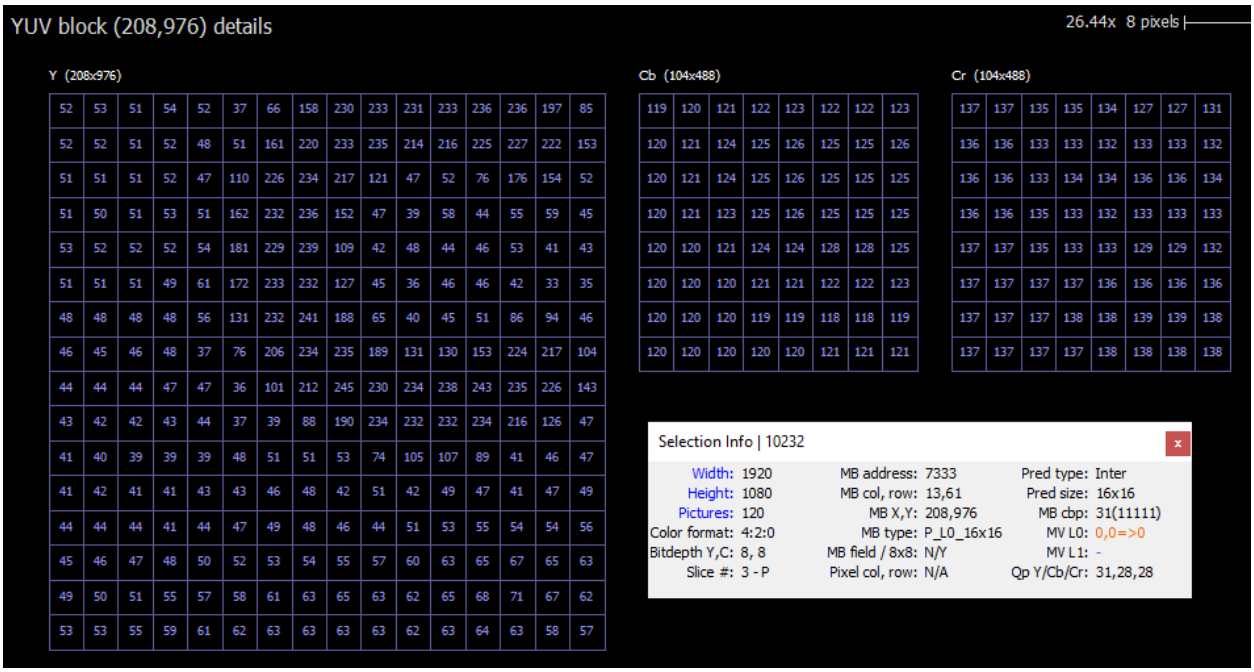
401. When encoding Twitch.tv video, on information and belief, Amazon performs encoding a digital video sequence for use in a video coding application to produce an encoded video bit-stream representative of the digital video sequence, the digital video sequence comprising a number of frames, each frame of said sequence comprising an array of pixels divided into a plurality of blocks, each block comprising a certain number of said pixels, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



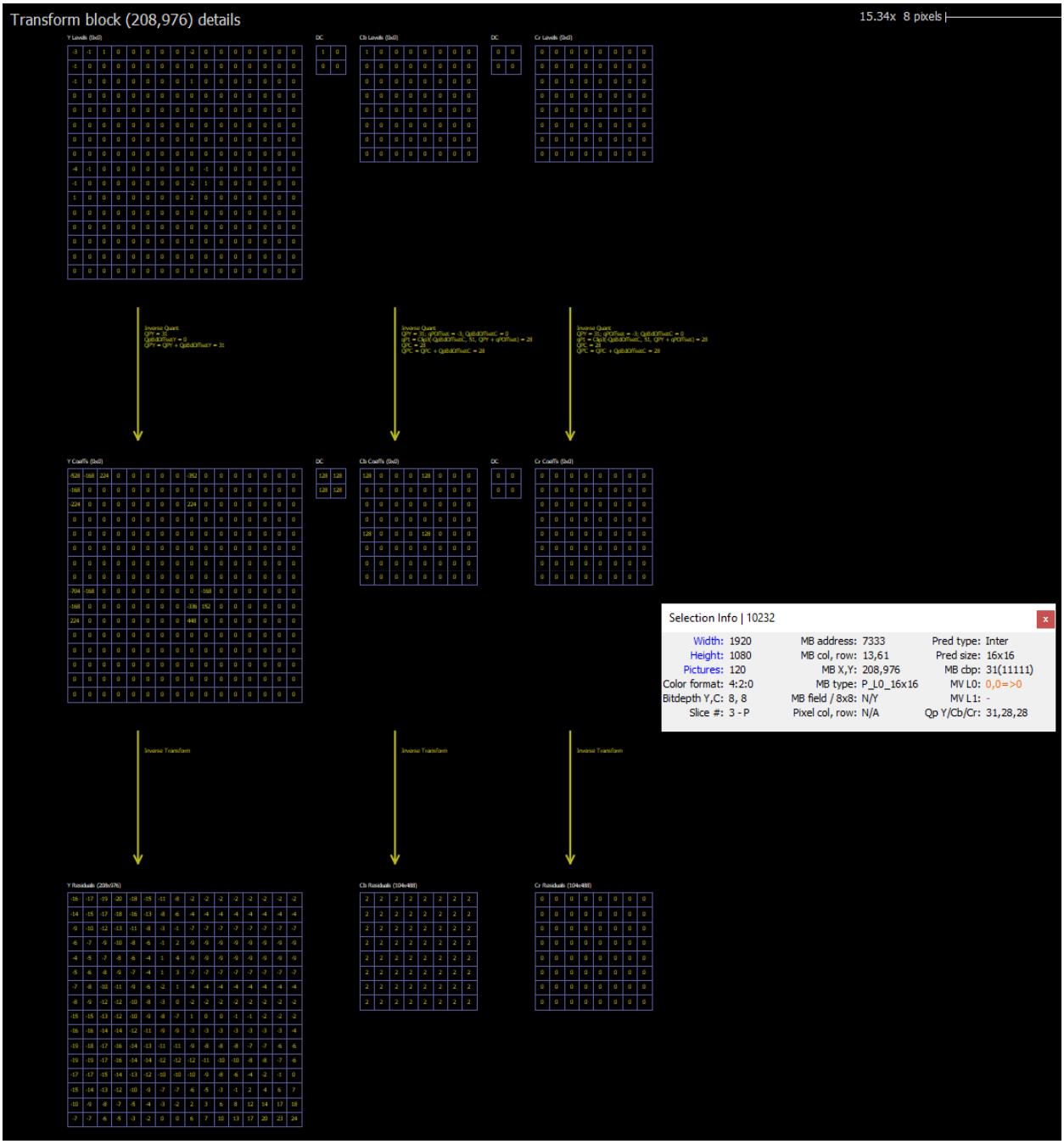
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



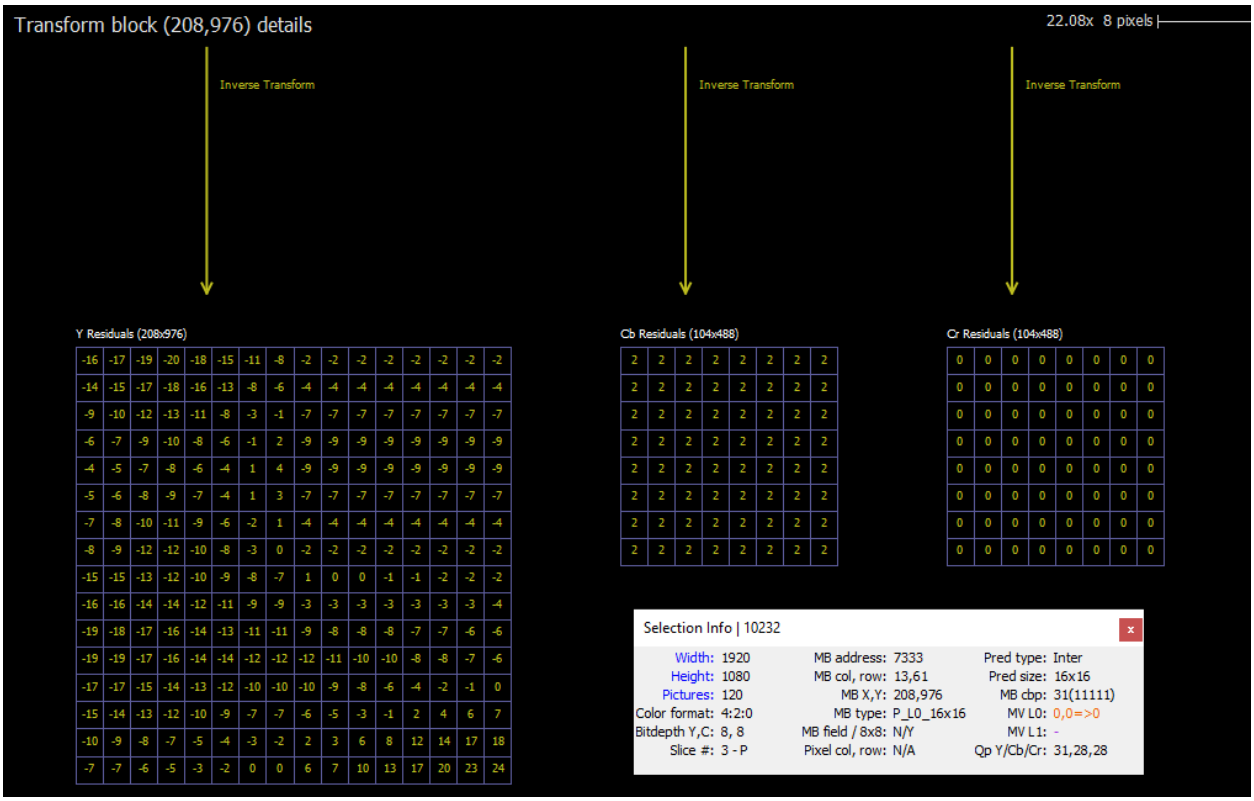
Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

402. When encoding Twitch.tv video, on information and belief, Amazon performs encoding a frame of the digital video sequence by applying motion compensated prediction to

blocks of pixels for producing corresponding blocks of prediction error values, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Transform block (208,976) details

Inverse Transform

Y Residuals (208x976)

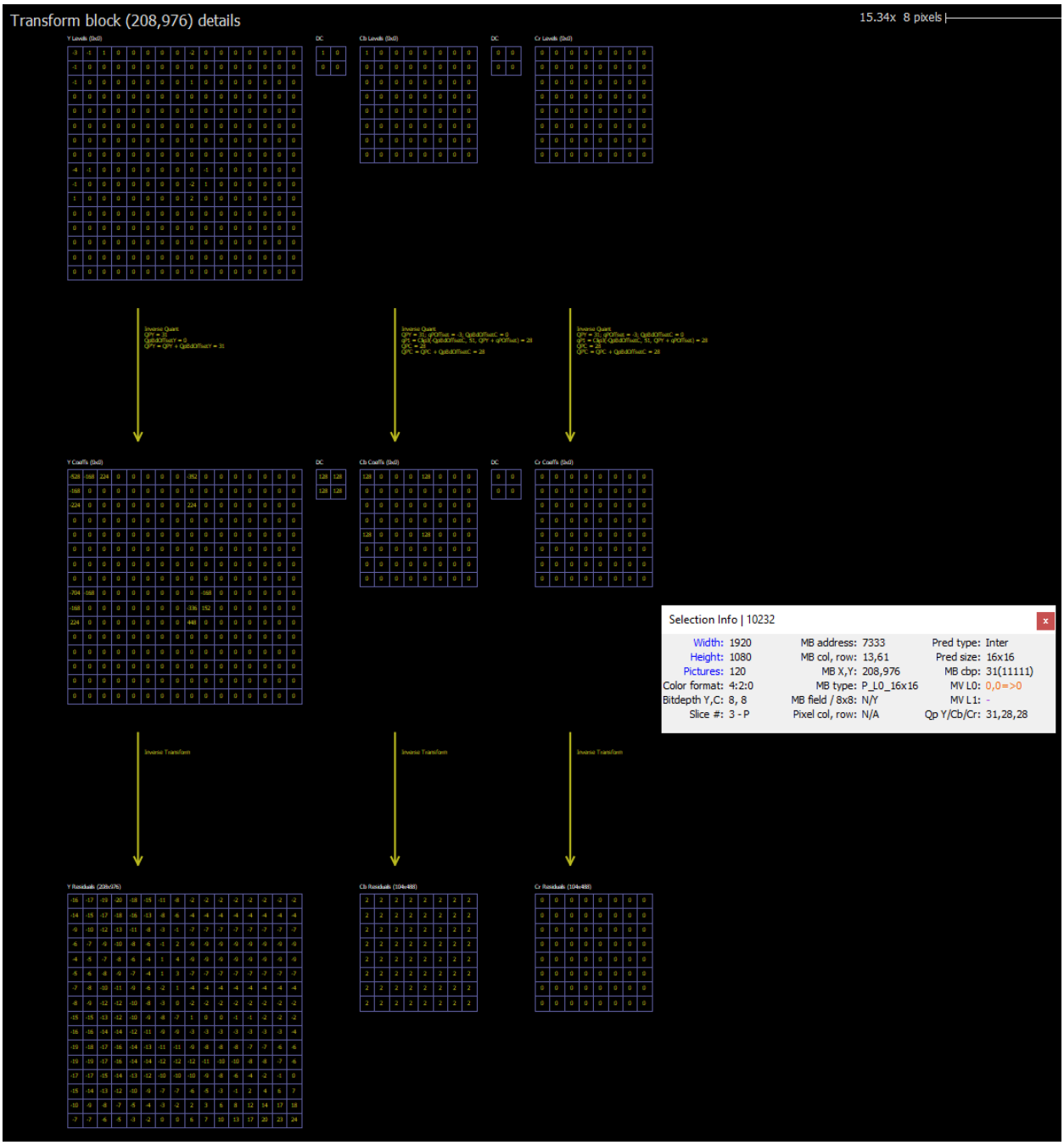
-16	-17	-19	-20	-18	-15	-11	-8	-2	-2	-2	-2	-2	-2	-2	-2
-14	-15	-17	-18	-16	-13	-8	-6	-4	-4	-4	-4	-4	-4	-4	-4
-9	-10	-12	-13	-11	-8	-3	-1	-7	-7	-7	-7	-7	-7	-7	-7
-6	-7	-9	-10	-8	-6	-1	2	-9	-9	-9	-9	-9	-9	-9	-9
-4	-5	-7	-8	-6	-4	1	4	-9	-9	-9	-9	-9	-9	-9	-9
-5	-6	-8	-9	-7	-4	1	3	-7	-7	-7	-7	-7	-7	-7	-7
-7	-8	-10	-11	-9	-6	-2	1	-4	-4	-4	-4	-4	-4	-4	-4
-8	-9	-12	-12	-10	-8	-3	0	-2	-2	-2	-2	-2	-2	-2	-2
-15	-15	-13	-12	-10	-9	-8	-7	1	0	0	-1	-1	-2	-2	-2
-16	-16	-14	-14	-12	-11	-9	-9	-3	-3	-3	-3	-3	-3	-3	-4
-19	-18	-17	-16	-14	-13	-11	-11	-9	-8	-8	-8	-7	-7	-6	-6
-19	-19	-17	-16	-14	-14	-12	-12	-12	-11	-10	-10	-8	-8	-7	-6
-17	-17	-15	-14	-13	-12	-10	-10	-10	-9	-8	-6	-4	-2	-1	0
-15	-14	-13	-12	-10	-9	-7	-7	-6	-5	-3	-1	2	4	6	7
-10	-9	-8	-7	-5	-4	-3	-2	2	3	6	8	12	14	17	18
-7	-7	-6	-5	-3	-2	0	0	6	7	10	13	17	20	23	24

Selection Info | 10232

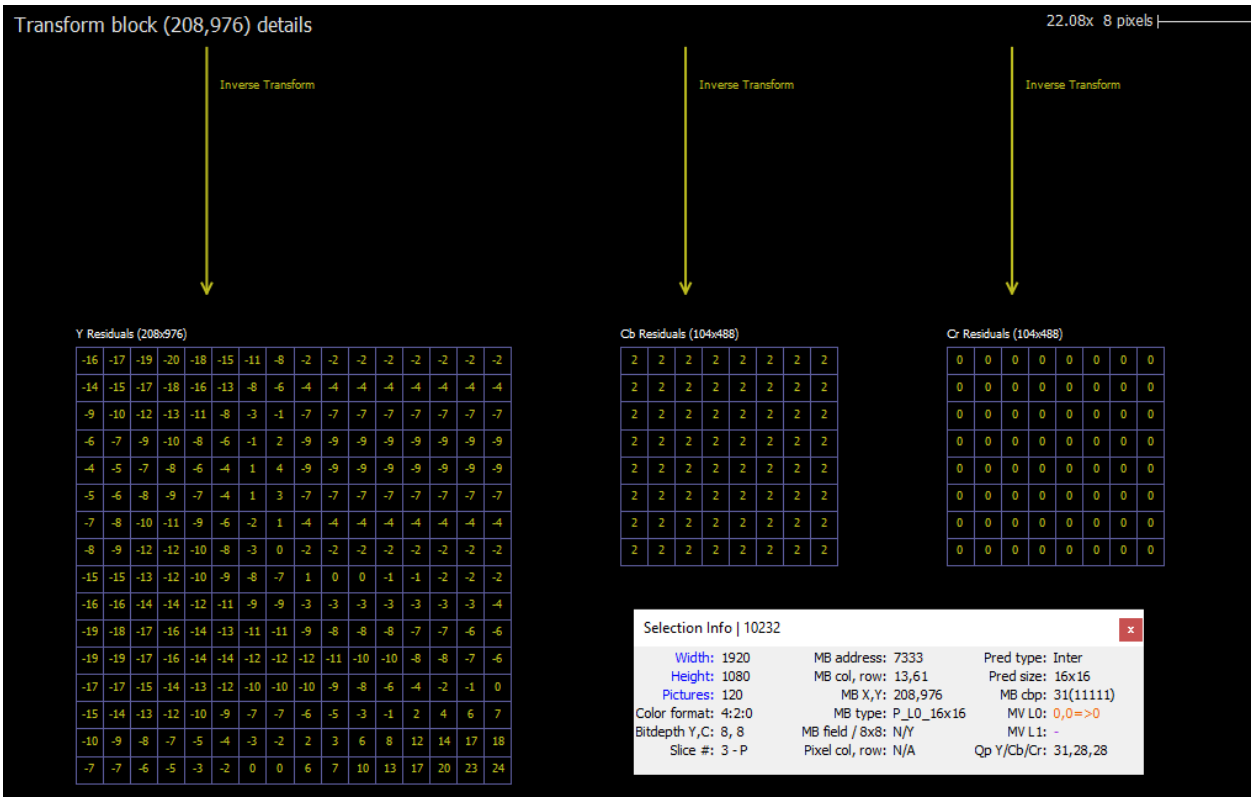
Width:	1920	MB address:	7333	Pred type:	Inter
Height:	1080	MB col, row:	13,61	Pred size:	16x16
Pictures:	120	MB X,Y:	208,976	MB cbp:	31(11111)
Color format:	4:2:0	MB type:	P_L0_16x16	MV L0:	0,0=>0
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/Y	MV L1:	-
Slice #:	3 - P	Pixel col, row:	N/A	Qp Y/Cb/Cr:	31,28,28

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

403. When encoding Twitch.tv video, on information and belief, Amazon performs applying a transform coding technique to said blocks of prediction error values to produce sets of transform coefficient values representative of said blocks of prediction error values, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Transform block (208,976) details

Inverse Transform

Y Residuals (208x976)

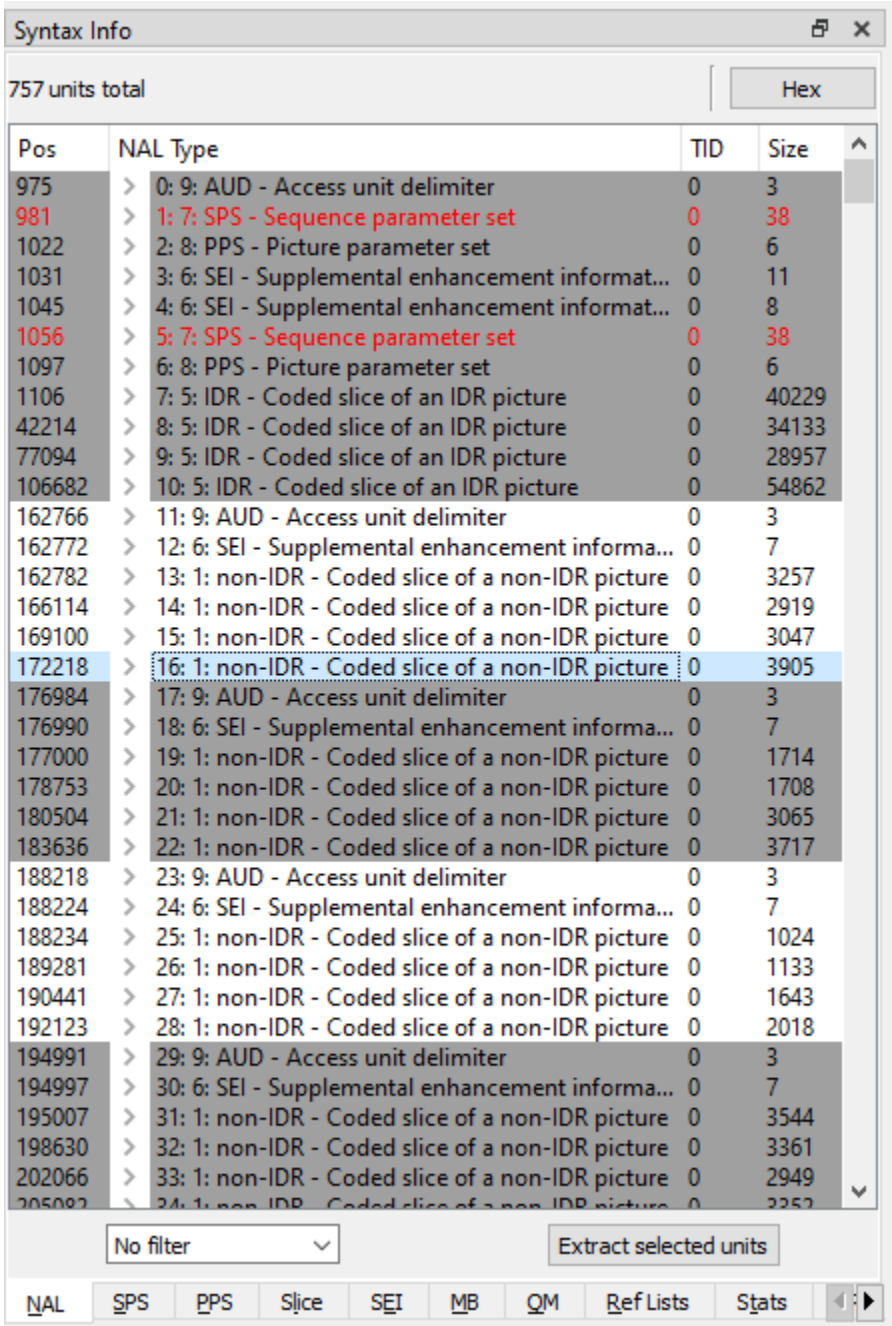
-16	-17	-19	-20	-18	-15	-11	-8	-2	-2	-2	-2	-2	-2	-2	-2
-14	-15	-17	-18	-16	-13	-8	-6	-4	-4	-4	-4	-4	-4	-4	-4
-9	-10	-12	-13	-11	-8	-3	-1	-7	-7	-7	-7	-7	-7	-7	-7
-6	-7	-9	-10	-8	-6	-1	2	-9	-9	-9	-9	-9	-9	-9	-9
-4	-5	-7	-8	-6	-4	1	4	-9	-9	-9	-9	-9	-9	-9	-9
-5	-6	-8	-9	-7	-4	1	3	-7	-7	-7	-7	-7	-7	-7	-7
-7	-8	-10	-11	-9	-6	-2	1	-4	-4	-4	-4	-4	-4	-4	-4
-8	-9	-12	-12	-10	-8	-3	0	-2	-2	-2	-2	-2	-2	-2	-2
-15	-15	-13	-12	-10	-9	-8	-7	1	0	0	-1	-1	-2	-2	-2
-16	-16	-14	-14	-12	-11	-9	-9	-3	-3	-3	-3	-3	-3	-3	-4
-19	-18	-17	-16	-14	-13	-11	-11	-9	-8	-8	-8	-7	-7	-6	-6
-19	-19	-17	-16	-14	-14	-12	-12	-12	-11	-10	-10	-8	-8	-7	-6
-17	-17	-15	-14	-13	-12	-10	-10	-10	-9	-8	-6	-4	-2	-1	0
-15	-14	-13	-12	-10	-9	-7	-7	-6	-5	-3	-1	2	4	6	7
-10	-9	-8	-7	-5	-4	-3	-2	2	3	6	8	12	14	17	18
-7	-7	-6	-5	-3	-2	0	0	6	7	10	13	17	20	23	24

Selection Info | 10232

Width:	1920	MB address:	7333	Pred type:	Inter
Height:	1080	MB col, row:	13,61	Pred size:	16x16
Pictures:	120	MB X,Y:	208,976	MB cbp:	31(11111)
Color format:	4:2:0	MB type:	P_L0_16x16	MV L0:	0,0=>0
Bitdepth Y,C:	8, 8	MB field / 8x8:	N/Y	MV L1:	-
Slice #:	3 - P	Pixel col, row:	N/A	Qp Y/Cb/Cr:	31,28,28

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

404. When encoding Twitch.tv video, on information and belief, Amazon performs defining a default level of quantization for use in encoding of the digital video sequence to quantize the sets of transform coefficient values, as demonstrated in the screenshots below using VQ Analyzer software.



Pos	NAL Type	TID	Size
975	0: 9: AUD - Access unit delimiter	0	3
981	1: 7: SPS - Sequence parameter set	0	38
1022	2: 8: PPS - Picture parameter set	0	6
1031	3: 6: SEI - Supplemental enhancement informat...	0	11
1045	4: 6: SEI - Supplemental enhancement informat...	0	8
1056	5: 7: SPS - Sequence parameter set	0	38
1097	6: 8: PPS - Picture parameter set	0	6
1106	7: 5: IDR - Coded slice of an IDR picture	0	40229
42214	8: 5: IDR - Coded slice of an IDR picture	0	34133
77094	9: 5: IDR - Coded slice of an IDR picture	0	28957
106682	10: 5: IDR - Coded slice of an IDR picture	0	54862
162766	11: 9: AUD - Access unit delimiter	0	3
162772	12: 6: SEI - Supplemental enhancement informa...	0	7
162782	13: 1: non-IDR - Coded slice of a non-IDR picture	0	3257
166114	14: 1: non-IDR - Coded slice of a non-IDR picture	0	2919
169100	15: 1: non-IDR - Coded slice of a non-IDR picture	0	3047
172218	16: 1: non-IDR - Coded slice of a non-IDR picture	0	3905
176984	17: 9: AUD - Access unit delimiter	0	3
176990	18: 6: SEI - Supplemental enhancement informa...	0	7
177000	19: 1: non-IDR - Coded slice of a non-IDR picture	0	1714
178753	20: 1: non-IDR - Coded slice of a non-IDR picture	0	1708
180504	21: 1: non-IDR - Coded slice of a non-IDR picture	0	3065
183636	22: 1: non-IDR - Coded slice of a non-IDR picture	0	3717
188218	23: 9: AUD - Access unit delimiter	0	3
188224	24: 6: SEI - Supplemental enhancement informa...	0	7
188234	25: 1: non-IDR - Coded slice of a non-IDR picture	0	1024
189281	26: 1: non-IDR - Coded slice of a non-IDR picture	0	1133
190441	27: 1: non-IDR - Coded slice of a non-IDR picture	0	1643
192123	28: 1: non-IDR - Coded slice of a non-IDR picture	0	2018
194991	29: 9: AUD - Access unit delimiter	0	3
194997	30: 6: SEI - Supplemental enhancement informa...	0	7
195007	31: 1: non-IDR - Coded slice of a non-IDR picture	0	3544
198630	32: 1: non-IDR - Coded slice of a non-IDR picture	0	3361
202066	33: 1: non-IDR - Coded slice of a non-IDR picture	0	2949

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info

757 units total

Hex

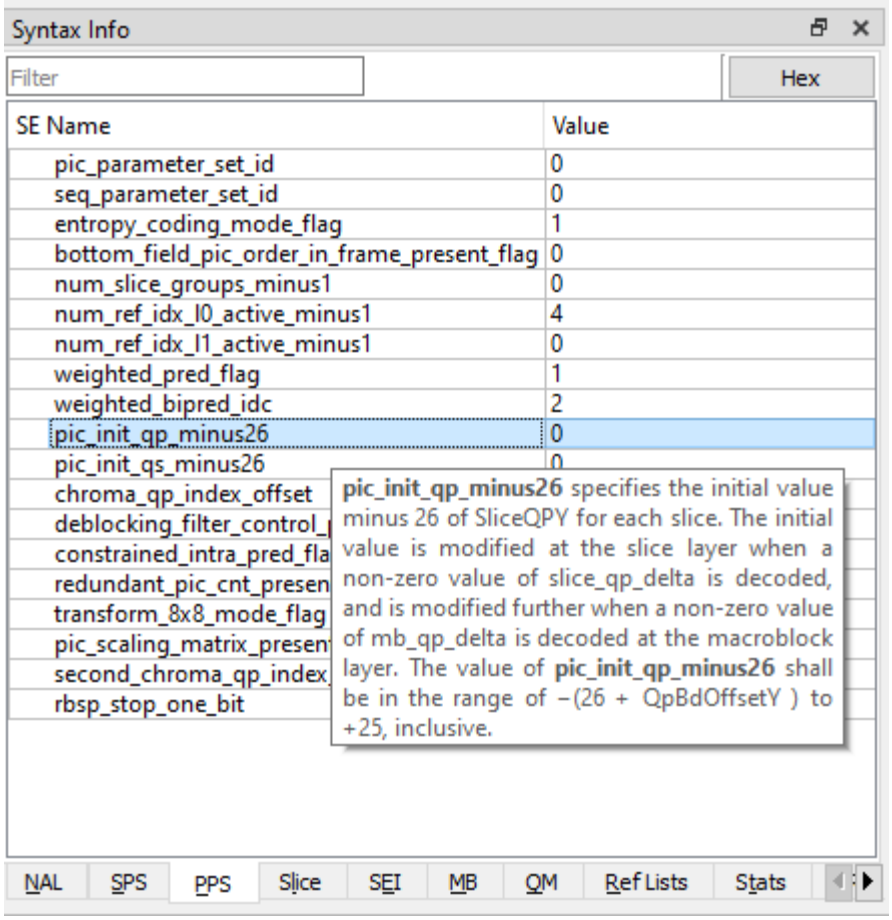
Pos	NAL Type	TID	Size
975	0: 9: AUD - Access unit delimiter	0	3
981	1: 7: SPS - Sequence parameter set	0	38
1022	2: 8: PPS - Picture parameter set	0	6
1031	3: 6: SEI - Supplemental enhancement informat...	0	11
1045	4: 6: SEI - Supplemental enhancement informat...	0	8
1056	5: 7: SPS - Sequence parameter set	0	38
1097	6: 8: PPS - Picture parameter set	0	6
1106	7: 5: IDR - Coded slice of an IDR picture	0	40229
42214	8: 5: IDR - Coded slice of an IDR picture	0	34133
77094	9: 5: IDR - Coded slice of an IDR picture	0	28957
106682	10: 5: IDR - Coded slice of an IDR picture	0	54862
162766	11: 9: AUD - Access unit delimiter	0	3
162772	12: 6: SEI - Supplemental enhancement informa...	0	7
162782	13: 1: non-IDR - Coded slice of a non-IDR picture	0	3257
166114	14: 1: non-IDR - Coded slice of a non-IDR picture	0	2919
169100	15: 1: non-IDR - Coded slice of a non-IDR picture	0	3047
172218	16: 1: non-IDR - Coded slice of a non-IDR picture	0	3905
176984	17: 9: AUD - Access unit delimiter	0	3
176990	18: 6: SEI - Supplemental enhancement informa...	0	7
177000	19: 1: non-IDR - Coded slice of a non-IDR picture	0	1714
178753	20: 1: non-IDR - Coded slice of a non-IDR picture	0	1708
180504	21: 1: non-IDR - Coded slice of a non-IDR picture	0	3065
183636	22: 1: non-IDR - Coded slice of a non-IDR picture	0	3717
188218	23: 9: AUD - Access unit delimiter	0	3
188224	24: 6: SEI - Supplemental enhancement informa...	0	7
188234	25: 1: non-IDR - Coded slice of a non-IDR picture	0	1024
189281	26: 1: non-IDR - Coded slice of a non-IDR picture	0	1133
190441	27: 1: non-IDR - Coded slice of a non-IDR picture	0	1643
192123	28: 1: non-IDR - Coded slice of a non-IDR picture	0	2018
194991	29: 9: AUD - Access unit delimiter	0	3
194997	30: 6: SEI - Supplemental enhancement informa...	0	7
195007	31: 1: non-IDR - Coded slice of a non-IDR picture	0	3544
198630	32: 1: non-IDR - Coded slice of a non-IDR picture	0	3361
202066	33: 1: non-IDR - Coded slice of a non-IDR picture	0	2949

No filter

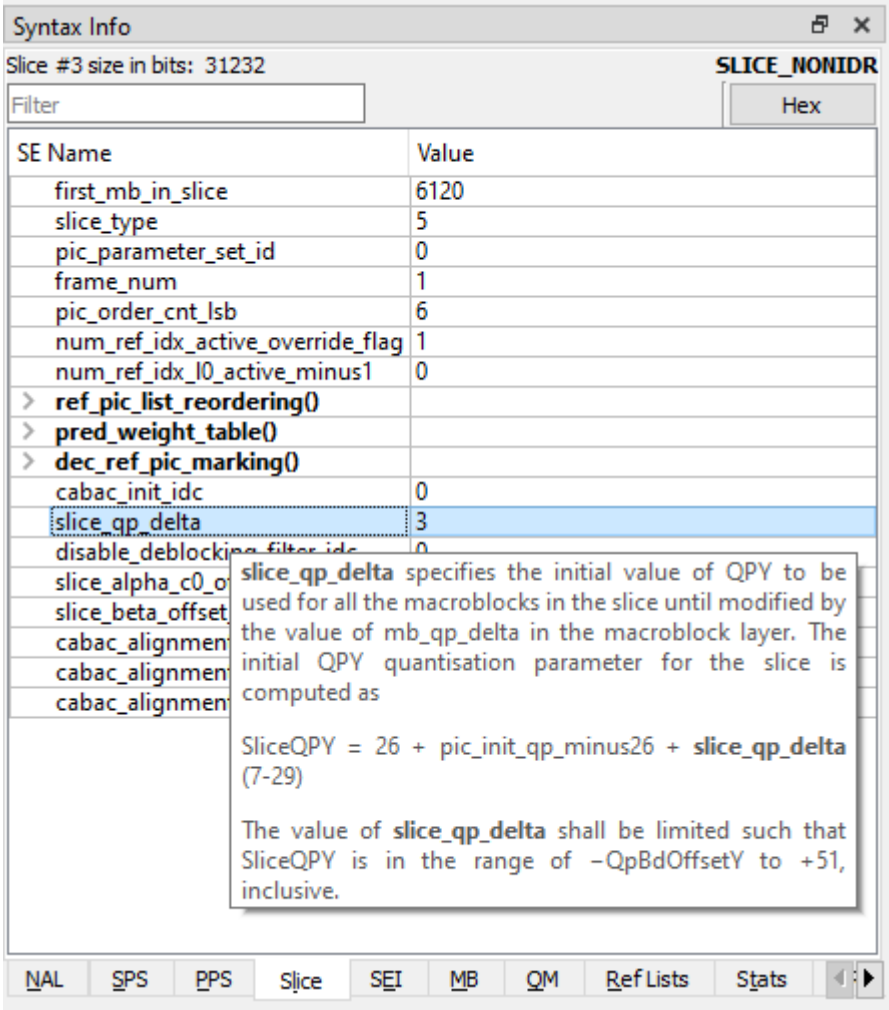
Extract selected units

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info

Slice #2 size in bits: 24376 **SLICE_NONIDR**

Filter Hex

SE Name	Value
first_mb_in_slice	4080
slice_type	5
pic_parameter_set_id	0
frame_num	1
pic_order_cnt_lsb	6
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	0
ref_pic_list_reordering()	
pred_weight_table()	
dec_ref_pic_marking()	
cabac_init_idc	0
slice_qp_delta	2
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	-1
slice_beta_offset_div2	-1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info

Slice #1 size in bits: 23352 **SLICE_NONIDR**

Filter Hex

SE Name	Value
first_mb_in_slice	2040
slice_type	5
pic_parameter_set_id	0
frame_num	1
pic_order_cnt_lsb	6
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	0
ref_pic_list_reordering()	
pred_weight_table()	
dec_ref_pic_marking()	
cabac_init_idc	0
slice_qp_delta	5
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	-1
slice_beta_offset_div2	-1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info

Filter | Hex

Pos	R	V	SE Name	Value	Bits	Bits, %
0x0002a250:4	494	74	mb_skip_flag	0	0	0.00
0x0002a250:4	418	74	mb_type	1	1	3.45
0x0002a250:5	415	149	mvd_l0[0]	0	1	3.45
0x0002a250:6	436	299	mvd_l0[1]	0	0	0.00
0x0002a250:6	306	299	mvd_l0[0]	1	6	20.69
0x0002a251:4	304	206	mvd_l0[1]	0	1	3.45
0x0002a251:5	438	412	coded_block_pattern	19	5	17.24
0x0002a252:2	357	196	transform_size_8x8_flag	1	0	0.00
0x0002a252:2	304	196	mb_qp_delta	5	7	24.14
0x00000000:0			residual_block_cabac		2	6.90

mb_qp_delta can change the value of QPY in the macroblock layer. The decoded value of **mb_qp_delta** shall be in the range of −(26 + QpBdOffsetY / 2) to +(25 + QpBdOffsetY / 2), inclusive. **mb_qp_delta** shall be inferred to be equal to 0 when it is not present for any macroblock (including P_Skip and B_Skip macroblock types).

The value of QPY is derived as

QPY = ((QPY,PREV + **mb_qp_delta** + 52 + 2 * QpBdOffsetY) % (52 + QpBdOffsetY)) − QpBdOffsetY (7-36)

where QPY,PREV is the luma quantisation parameter, QPY, of the previous macroblock in decoding order in the current slice. For the first macroblock in the slice QPY,PREV is initially set equal to SliceQPY derived in Equation 7-29 at the start of each slice.

The value of QP'Y is derived as

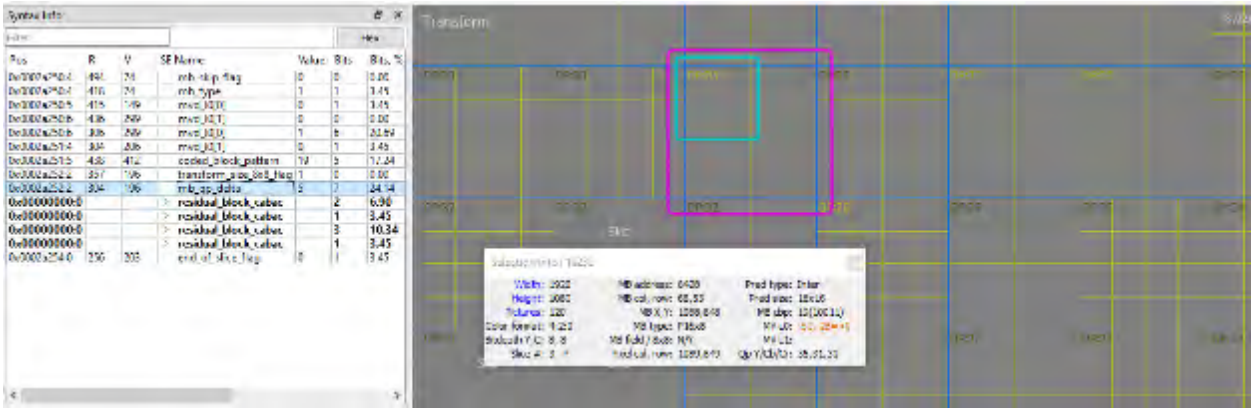
QP'Y = QPY + QpBdOffsetY (7-37)

The variable TransformBypassModeFlag is derived as follows.

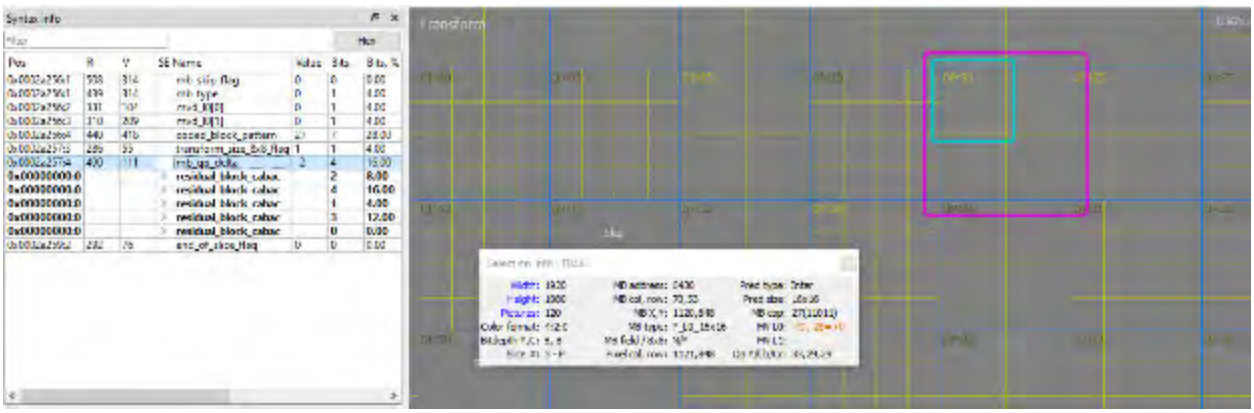
- If qpprime_y_zero_transform_bypass_flag is equal to 1 and QP'Y is equal to 0, TransformBypassModeFlag is set equal to 1.
- Otherwise (qpprime_y_zero_transform_bypass_flag is equal to 0 or QP'Y is not equal to 0),

TransformBypassModeFlag is set equal to 0.

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

405. When encoding Twitch.tv video, on information and belief, Amazon performs providing an indication of the default level of quantization to a decoding process, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info

757 units total | Hex

Pos	NAL Type	TID	Size
975	0: 9: AUD - Access unit delimiter	0	3
981	1: 7: SPS - Sequence parameter set	0	38
1022	2: 8: PPS - Picture parameter set	0	6
1031	3: 6: SEI - Supplemental enhancement informat...	0	11
1045	4: 6: SEI - Supplemental enhancement informat...	0	8
1056	5: 7: SPS - Sequence parameter set	0	38
1097	6: 8: PPS - Picture parameter set	0	6
1106	7: 5: IDR - Coded slice of an IDR picture	0	40229
42214	8: 5: IDR - Coded slice of an IDR picture	0	34133
77094	9: 5: IDR - Coded slice of an IDR picture	0	28957
106682	10: 5: IDR - Coded slice of an IDR picture	0	54862
162766	11: 9: AUD - Access unit delimiter	0	3
162772	12: 6: SEI - Supplemental enhancement informa...	0	7
162782	13: 1: non-IDR - Coded slice of a non-IDR picture	0	3257
166114	14: 1: non-IDR - Coded slice of a non-IDR picture	0	2919
169100	15: 1: non-IDR - Coded slice of a non-IDR picture	0	3047
172218	16: 1: non-IDR - Coded slice of a non-IDR picture	0	3905
176984	17: 9: AUD - Access unit delimiter	0	3
176990	18: 6: SEI - Supplemental enhancement informa...	0	7
177000	19: 1: non-IDR - Coded slice of a non-IDR picture	0	1714
178753	20: 1: non-IDR - Coded slice of a non-IDR picture	0	1708
180504	21: 1: non-IDR - Coded slice of a non-IDR picture	0	3065
183636	22: 1: non-IDR - Coded slice of a non-IDR picture	0	3717
188218	23: 9: AUD - Access unit delimiter	0	3
188224	24: 6: SEI - Supplemental enhancement informa...	0	7
188234	25: 1: non-IDR - Coded slice of a non-IDR picture	0	1024
189281	26: 1: non-IDR - Coded slice of a non-IDR picture	0	1133
190441	27: 1: non-IDR - Coded slice of a non-IDR picture	0	1643
192123	28: 1: non-IDR - Coded slice of a non-IDR picture	0	2018
194991	29: 9: AUD - Access unit delimiter	0	3
194997	30: 6: SEI - Supplemental enhancement informa...	0	7
195007	31: 1: non-IDR - Coded slice of a non-IDR picture	0	3544
198630	32: 1: non-IDR - Coded slice of a non-IDR picture	0	3361
202066	33: 1: non-IDR - Coded slice of a non-IDR picture	0	2949

No filter | Extract selected units

NAL | SPS | PPS | Slice | SEI | MB | QM | Ref Lists | Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info

757 units total

Hex

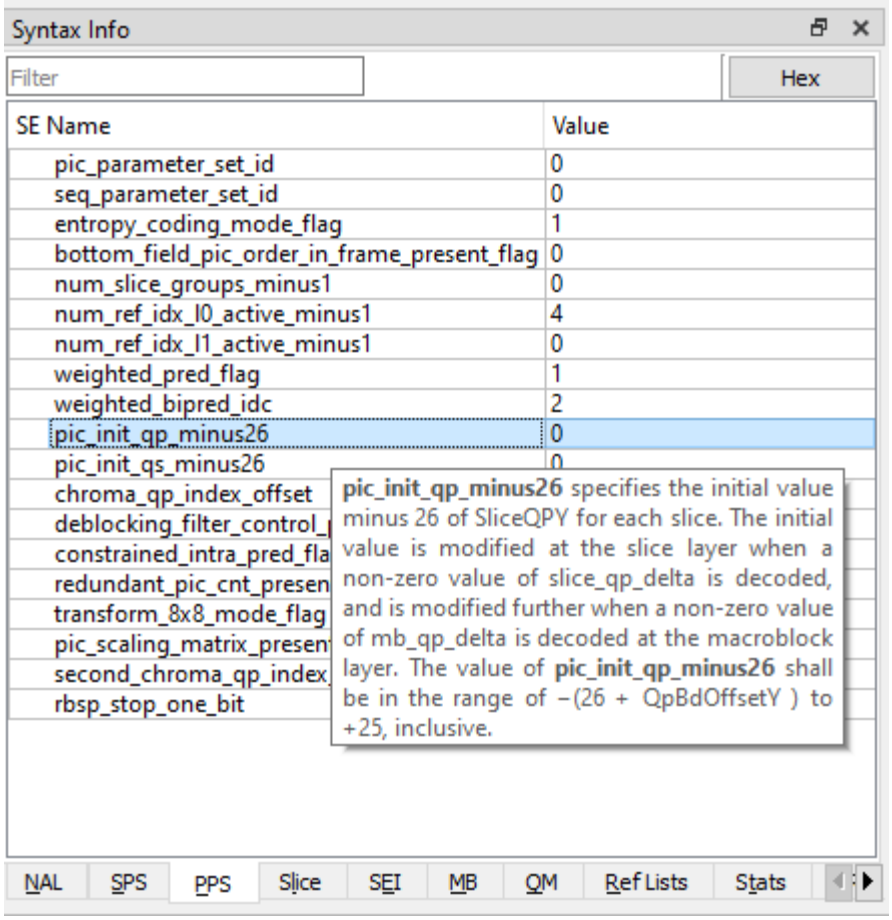
Pos	NAL Type	TID	Size
975	0: 9: AUD - Access unit delimiter	0	3
981	1: 7: SPS - Sequence parameter set	0	38
1022	2: 8: PPS - Picture parameter set	0	6
1031	3: 6: SEI - Supplemental enhancement informat...	0	11
1045	4: 6: SEI - Supplemental enhancement informat...	0	8
1056	5: 7: SPS - Sequence parameter set	0	38
1097	6: 8: PPS - Picture parameter set	0	6
1106	7: 5: IDR - Coded slice of an IDR picture	0	40229
42214	8: 5: IDR - Coded slice of an IDR picture	0	34133
77094	9: 5: IDR - Coded slice of an IDR picture	0	28957
106682	10: 5: IDR - Coded slice of an IDR picture	0	54862
162766	11: 9: AUD - Access unit delimiter	0	3
162772	12: 6: SEI - Supplemental enhancement informa...	0	7
162782	13: 1: non-IDR - Coded slice of a non-IDR picture	0	3257
166114	14: 1: non-IDR - Coded slice of a non-IDR picture	0	2919
169100	15: 1: non-IDR - Coded slice of a non-IDR picture	0	3047
172218	16: 1: non-IDR - Coded slice of a non-IDR picture	0	3905
176984	17: 9: AUD - Access unit delimiter	0	3
176990	18: 6: SEI - Supplemental enhancement informa...	0	7
177000	19: 1: non-IDR - Coded slice of a non-IDR picture	0	1714
178753	20: 1: non-IDR - Coded slice of a non-IDR picture	0	1708
180504	21: 1: non-IDR - Coded slice of a non-IDR picture	0	3065
183636	22: 1: non-IDR - Coded slice of a non-IDR picture	0	3717
188218	23: 9: AUD - Access unit delimiter	0	3
188224	24: 6: SEI - Supplemental enhancement informa...	0	7
188234	25: 1: non-IDR - Coded slice of a non-IDR picture	0	1024
189281	26: 1: non-IDR - Coded slice of a non-IDR picture	0	1133
190441	27: 1: non-IDR - Coded slice of a non-IDR picture	0	1643
192123	28: 1: non-IDR - Coded slice of a non-IDR picture	0	2018
194991	29: 9: AUD - Access unit delimiter	0	3
194997	30: 6: SEI - Supplemental enhancement informa...	0	7
195007	31: 1: non-IDR - Coded slice of a non-IDR picture	0	3544
198630	32: 1: non-IDR - Coded slice of a non-IDR picture	0	3361
202066	33: 1: non-IDR - Coded slice of a non-IDR picture	0	2949

No filter

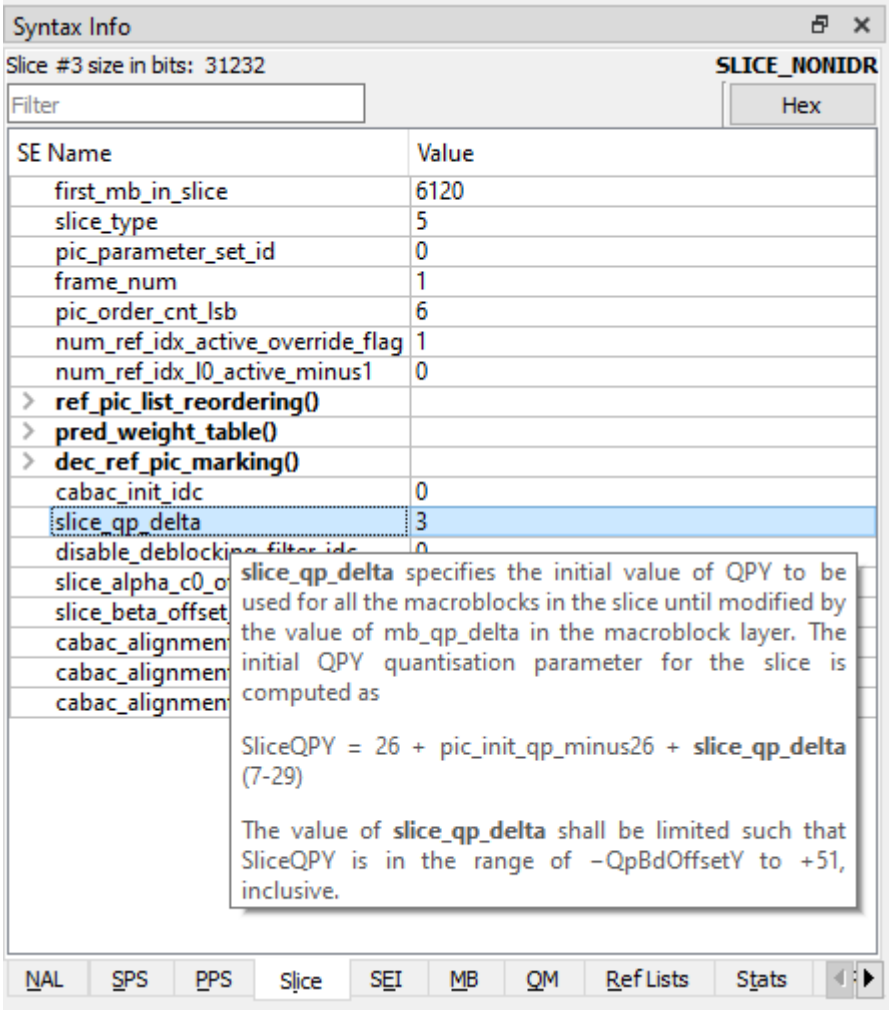
Extract selected units

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info

Slice #2 size in bits: 24376 **SLICE_NONIDR**

Filter Hex

SE Name	Value
first_mb_in_slice	4080
slice_type	5
pic_parameter_set_id	0
frame_num	1
pic_order_cnt_lsb	6
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	0
ref_pic_list_reordering()	
pred_weight_table()	
dec_ref_pic_marking()	
cabac_init_idc	0
slice_qp_delta	2
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	-1
slice_beta_offset_div2	-1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info

Slice #1 size in bits: 23352 **SLICE_NONIDR**

Filter Hex

SE Name	Value
first_mb_in_slice	2040
slice_type	5
pic_parameter_set_id	0
frame_num	1
pic_order_cnt_lsb	6
num_ref_idx_active_override_flag	1
num_ref_idx_l0_active_minus1	0
ref_pic_list_reordering()	
pred_weight_table()	
dec_ref_pic_marking()	
cabac_init_idc	0
slice_qp_delta	5
disable_deblocking_filter_idc	0
slice_alpha_c0_offset_div2	-1
slice_beta_offset_div2	-1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1
cabac_alignment_one_bit	1

NAL SPS PPS Slice SEI MB QM Ref Lists Stats

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Syntax Info

Filter | Hex

Pos	R	V	SE Name	Value	Bits	Bits, %
0x0002a250:4	494	74	mb_skip_flag	0	0	0.00
0x0002a250:4	418	74	mb_type	1	1	3.45
0x0002a250:5	415	149	mvd_l0[0]	0	1	3.45
0x0002a250:6	436	299	mvd_l0[1]	0	0	0.00
0x0002a250:6	306	299	mvd_l0[0]	1	6	20.69
0x0002a251:4	304	206	mvd_l0[1]	0	1	3.45
0x0002a251:5	438	412	coded_block_pattern	19	5	17.24
0x0002a252:2	357	196	transform_size_8x8_flag	1	0	0.00
0x0002a252:2	304	196	mb_qp_delta	5	7	24.14
0x00000000:0			residual_block_cabac		2	6.90

mb_qp_delta can change the value of QPY in the macroblock layer. The decoded value of **mb_qp_delta** shall be in the range of −(26 + QpBdOffsetY / 2) to +(25 + QpBdOffsetY / 2), inclusive. **mb_qp_delta** shall be inferred to be equal to 0 when it is not present for any macroblock (including P_Skip and B_Skip macroblock types).

The value of QPY is derived as

QPY = ((QPY,PREV + **mb_qp_delta** + 52 + 2 * QpBdOffsetY) % (52 + QpBdOffsetY)) − QpBdOffsetY (7-36)

where QPY,PREV is the luma quantisation parameter, QPY, of the previous macroblock in decoding order in the current slice. For the first macroblock in the slice QPY,PREV is initially set equal to SliceQPY derived in Equation 7-29 at the start of each slice.

The value of QP'Y is derived as

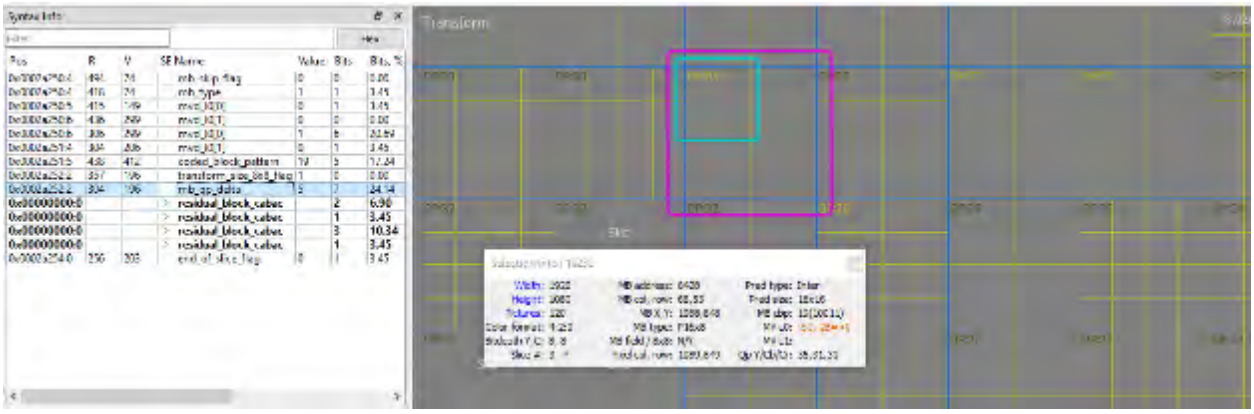
QP'Y = QPY + QpBdOffsetY (7-37)

The variable TransformBypassModeFlag is derived as follows.

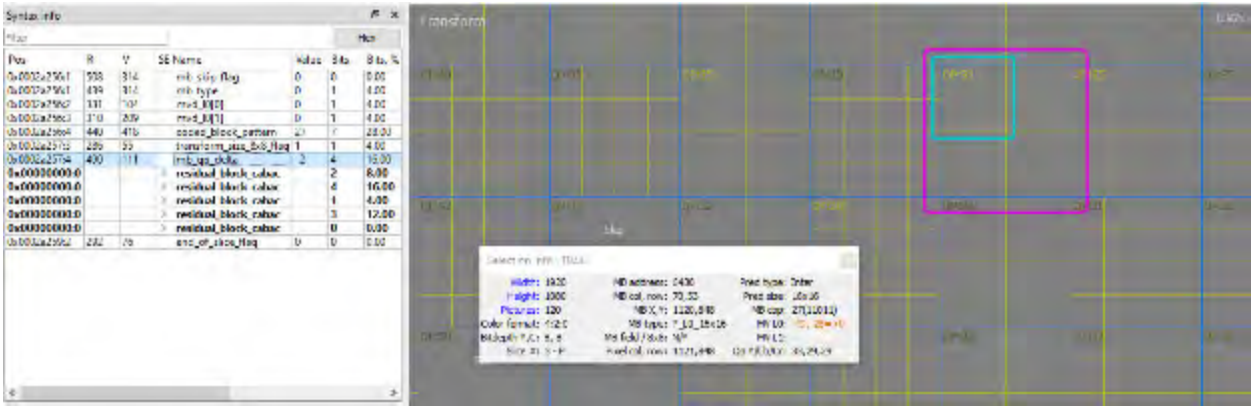
- If qpprime_y_zero_transform_bypass_flag is equal to 1 and QP'Y is equal to 0, TransformBypassModeFlag is set equal to 1.
- Otherwise (qpprime_y_zero_transform_bypass_flag is equal to 0 or QP'Y is not equal to 0),

TransformBypassModeFlag is set equal to 0.

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.



Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

Screenshots of VQ Analyzer software analysis of video available on Twitch.tv.

L. Amazon Makes, Imports, Uses, Sells, and/or Offers for Sale Products and Services that Infringe the ’991 Patent

406. The Accused Products infringe one or more claims of the ’991 Patent, including, for example, claim 1.

407. As just one example of infringement, on information and belief, Amazon performs a method of encoding a video signal in a manner that is covered by claim 1 of the ’991 Patent for Amazon Prime Video contents, as demonstrated in the screenshots below using VQ Analyzer software on a bitstream that indicates it can be decoded by an H.265-compliant decoder.

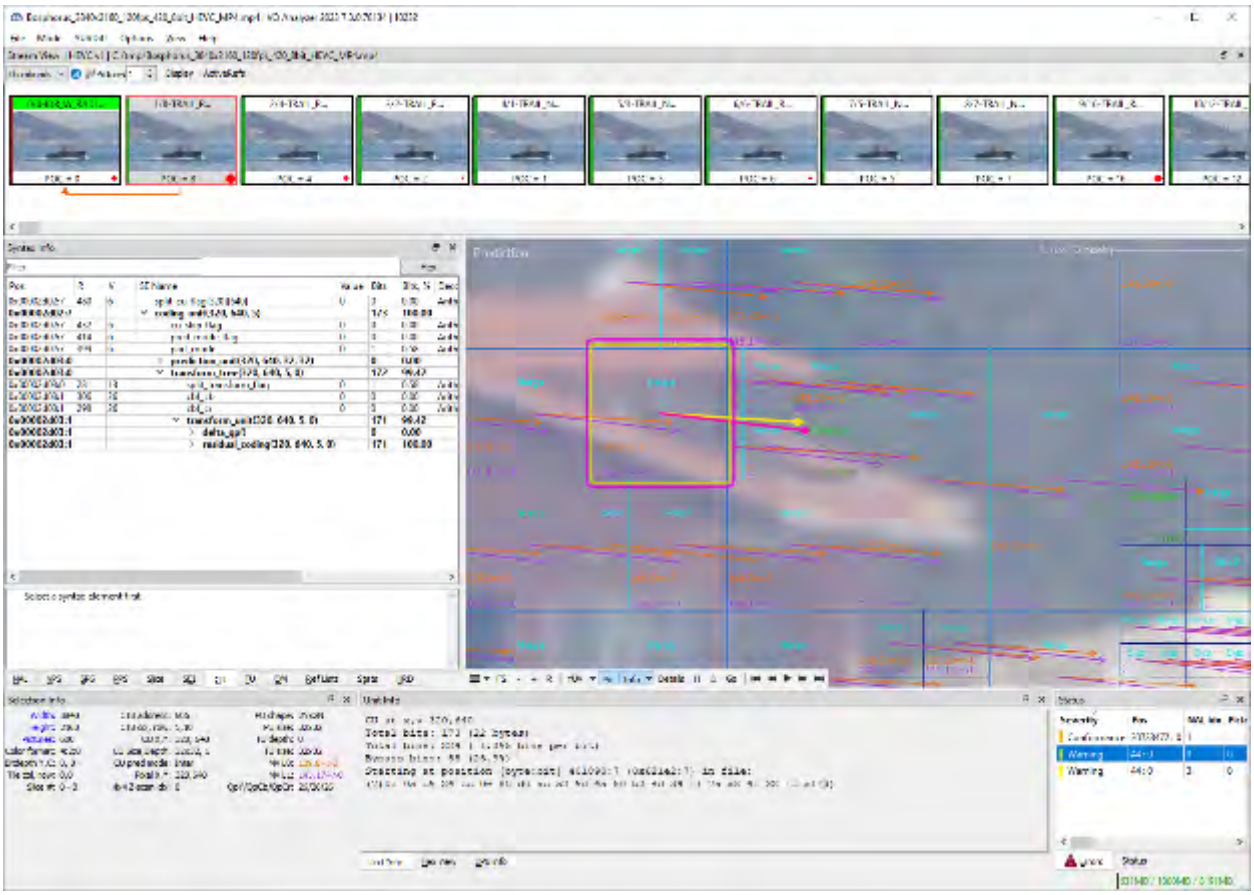
Screenshots of Amazon Prime Video indicating "Codec: hevc."

408. Encoded Amazon Prime Video bitstreams are encrypted such that the encrypted bitstream cannot be examined by using VQ Analyzer software.

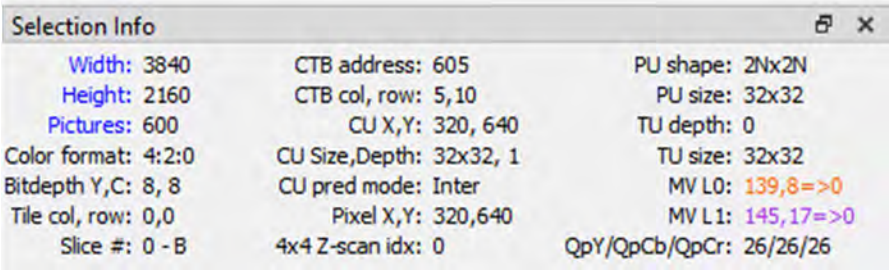
409. On information and belief, when encoding Amazon Prime Video content, Amazon performs a method of encoding that infringes one or more claims of the '991 Patent, including, for example, claim 1, by using the same or similar currently most popular and readily available techniques used by multiple other well-known encoding methods, which infringe the '991 Patent.

410. For example, the UVG H.265 test bitstreams are encoded in a manner that infringes one or more claims of the '991 Patent, including, for example, claim 1.

411. When encoding Amazon Prime Video trailers, on information and belief, Amazon produces a bitstream that is the same or substantially similar to the encoded UVG H.265 test bitstreams by performing calculating, by a processor, a difference signal representing differences between sample values of a predicted block of data and values for an original input block, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of UVG H.265 test bitstream.



Screenshots of VQ Analyzer software analysis of UVG H.265 test bitstream.

Syntax Info

Filter | Hex

Pos	R	V	SE Name	Value	Bits	Bits, %
0x00002d02:7	480	6	split_cu_flag[320][640]	0	0	0.00
0x00002d02:7			**coding_unit(320, 640, 5)**		**173**	**100.00**
0x00002d02:7	432	6	cu_skip_flag	0	0	0.00
0x00002d02:7	414	6	pred_mode_flag	0	0	0.00
0x00002d02:7	399	6	part_mode	0	1	0.58
0x00002d03:0			**prediction_unit(320, 640, 32, 32)**		**0**	**0.00**
0x00002d03:0			**transform_tree(320, 640, 5, 0)**		**172**	**99.42**
0x00002d03:0	281	13	split_transform_flag	0	1	0.58
0x00002d03:1	306	26	cbf_cb	0	0	0.00
0x00002d03:1	298	26	cbf_cr	0	0	0.00
0x00002d03:1			**transform_unit(320, 640, 5, 0)**		**171**	**99.42**
0x00002d03:1			**delta_qp()**		**0**	**0.00**
0x00002d03:1			**residual_coding(320, 640, 5, 0)**		**171**	**100.00**
0x00002d03:1	285	26	last_sig_coeff_x_prefix	4	3	1.75
0x00002d03:4	454	208	last_sig_coeff_y_prefix	6	3	1.75
0x00002d03:7	289	43	last_sig_coeff_x_suffix	0	1	0.58
0x00002d04:0	289	86	last_sig_coeff_y_suffix	1	2	1.17
0x00002d04:2	289	58	sig_coeff_flag	1	1	0.58
0x00002d04:3	322	116	coeff_abs_level_greater1_flag	0	0	0.00
0x00002d04:3	315	116	coeff_abs_level_greater1_flag	0	0	0.00
0x00002d04:3	300	116	coeff_sign_flag	0	1	0.58
0x00002d04:4	300	232	coeff_sign_flag	1	1	0.58
0x00002d04:5	300	165	coded_sub_block_flag	0	1	0.58

Select a syntax element first

NAL | VPS | SPS | PPS | Slice | SEI | CU | TU | QM | Ref Lists | Stats | HRD

Screenshots of VQ Analyzer software analysis of UVG H.265 test bitstream.

412. When encoding Amazon Prime Video trailers, on information and belief, Amazon produces a bitstream that is the same or substantially similar to the encoded UVG H.265 test bitstreams by performing, by a processor, transform coding to the difference signal, thereby creating a first representation of a first component of the difference signal, as demonstrated in the screenshots below using VQ Analyzer software.

Selection Info

Width:	3840	CTB address:	605	PU shape:	2Nx2N
Height:	2160	CTB col, row:	5,10	PU size:	32x32
Pictures:	600	CU X,Y:	320, 640	TU depth:	0
Color format:	4:2:0	CU Size,Depth:	32x32, 1	TU size:	32x32
Bitdepth Y,C:	8, 8	CU pred mode:	Inter	MV L0:	139,8=>0
Tile col, row:	0,0	Pixel X,Y:	320,640	MV L1:	145,17=>0
Slice #:	0 - B	4x4 Z-scan idx:	0	QpY/QpCb/QpCr:	26/26/26

Screenshots of VQ Analyzer software analysis of UVG H.265 test bitstream.

Syntax Info

Filter

Hex

Pos	R	V	SE Name	Value	Bits	Bits, %
0x00002d02:7	480	6	split_cu_flag[320][640]	0	0	0.00
0x00002d02:7			**coding_unit(320, 640, 5)**		**173**	**100.00**
0x00002d02:7	432	6	cu_skip_flag	0	0	0.00
0x00002d02:7	414	6	pred_mode_flag	0	0	0.00
0x00002d02:7	399	6	part_mode	0	1	0.58
0x00002d03:0			**prediction_unit(320, 640, 32, 32)**		**0**	**0.00**
0x00002d03:0			**transform_tree(320, 640, 5, 0)**		**172**	**99.42**
0x00002d03:0	281	13	split_transform_flag	0	1	0.58
0x00002d03:1	306	26	cbf_cb	0	0	0.00
0x00002d03:1	298	26	cbf_cr	0	0	0.00
0x00002d03:1			**transform_unit(320, 640, 5, 0)**		**171**	**99.42**
0x00002d03:1			**delta_qp()**		**0**	**0.00**
0x00002d03:1			**residual_coding(320, 640, 5, 0)**		**171**	**100.00**
0x00002d03:1	285	26	last_sig_coeff_x_prefix	4	3	1.75
0x00002d03:4	454	208	last_sig_coeff_y_prefix	6	3	1.75
0x00002d03:7	289	43	last_sig_coeff_x_suffix	0	1	0.58
0x00002d04:0	289	86	last_sig_coeff_y_suffix	1	2	1.17
0x00002d04:2	289	58	sig_coeff_flag	1	1	0.58
0x00002d04:3	322	116	coeff_abs_level_greater1_flag	0	0	0.00
0x00002d04:3	315	116	coeff_abs_level_greater1_flag	0	0	0.00
0x00002d04:3	300	116	coeff_sign_flag	0	1	0.58
0x00002d04:4	300	232	coeff_sign_flag	1	1	0.58
0x00002d04:5	300	165	coded_sub_block_flag	0	1	0.58

Select a syntax element first

NAL VPS SPS PPS Slice SEI CU TU QM Ref Lists Stats HRD

Screenshots of VQ Analyzer software analysis of UVG H.265 test bitstream.

413. When encoding Amazon Prime Video trailers, on information and belief, Amazon produces a bitstream that is the same or substantially similar to the encoded UVG H.265 test bitstreams by performing, by a processor, spatial coding to the difference signal, thereby creating a second representation of a first component of the difference signal, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info

Filter | Hex

Pos	R	V	SE Name	Value	Bits	Bits, %	Decoding type
0x00002d00:7			**sao(5, 10)**		**15**	**2.43**	
0x00002d00:7	329	326	sao_merge_left_flag	0	1	6.67	Arithmetic
0x00002d01:0	256	251	sao_merge_up_flag	0	2	13.33	Arithmetic
0x00002d01:2	464	446	sao_type_idx_luma	2	3	20.00	Arithmetic
0x00002d01:5	304	162	sao_offset_abs[0][0]	1	2	13.33	Bypass
0x00002d01:7	304	40	sao_offset_abs[0][1]	0	1	6.67	Bypass
0x00002d02:0	304	80	sao_offset_abs[0][2]	0	1	6.67	Bypass
0x00002d02:1	304	160	sao_offset_abs[0][3]	1	2	13.33	Bypass
0x00002d02:3	304	32	sao_eo_class_luma	0	2	13.33	Bypass
0x00002d02:5	304	129	sao_type_idx_chroma	0	1	6.67	Arithmetic
0x00002d02:6			**coding_quadtree(320, 640, 6, 0)**		**603**	**97.57**	
0x00002d4e:1	492	316	end_of_slice_segment_flag	0	0	0.00	Termination

Screenshots of VQ Analyzer software analysis of UVG H.265 test bitstream.

Syntax Info | 10232

Filter | Hex

Pos	R	V	SE Name	Value	Bits	B
0x00002d00:7			**sao(5, 10)**		**15**	**2.**
0x00002d02:6			**coding_quadtree(320, 640, 6, 0)**		**603**	**9**
0x00002d02:6	496	259	split_cu_flag[320][640]	1	1	0.
0x00002d02:7			**coding_quadtree(320, 640, 5, 1)**		**173**	**2**
0x00002d02:7	480	6	split_cu_flag[320][640]	0	0	0.
0x00002d02:7			**coding_unit(320, 640, 5)**		**173**	**1**
0x00002d02:7	432	6	cu_skip_flag	0	0	0.
0x00002d02:7	414	6	pred_mode_flag	0	0	0.
0x00002d02:7	399	6	part_mode	0	1	0.
0x00002d03:0			**prediction_unit(320, 640, 32, 32)**		**0**	**0.**
0x00002d03:0			**transform_tree(320, 640, 5, 0)**		**172**	**9**
0x00002d03:0	281	13	split_transform_flag	0	1	0.
0x00002d03:1	306	26	cbf_cb	0	0	0.
0x00002d03:1	298	26	cbf_cr	0	0	0.
0x00002d03:1			**transform_unit(320, 640, 5, 0)**		**171**	**9**
0x00002d03:1			**delta_qp()**		**0**	**0.**
0x00002d03:1			**residual_coding(320, 640, 5, 0)**		**171**	**1**
0x00002d03:1	285	26	last_sig_coeff_x_prefix	4	3	1.
0x00002d03:4	454	208	last_sig_coeff_y_prefix	6	3	1.
0x00002d03:7	289	43	last_sig_coeff_x_suffix	0	1	0.
0x00002d04:0	289	86	last_sig_coeff_y_suffix	1	2	1.
0x00002d04:2	289	58	sig_coeff_flag	1	1	0.
0x00002d04:3	322	116	coeff_abs_level_greater1_flag	0	0	0.
0x00002d04:3	315	116	coeff_abs_level_greater1_flag	0	0	0.
0x00002d04:3	300	116	coeff_sign_flag	0	1	0.
0x00002d04:4	300	232	coeff_sign_flag	1	1	0.
0x00002d04:5	300	165	coded_sub_block_flag	0	1	0.
0x00002d04:6	344	330	coded_sub_block_flag	1	1	0.
0x00002d04:7	316	288	sig_coeff_flag	0	0	0.
0x00002d04:7	296	288	sig_coeff_flag	1	4	2.
0x00002d05:3	304	185	sig_coeff_flag	0	0	0.
0x00002d05:3	272	185	sig_coeff_flag	0	1	0.
0x00002d05:4	484	370	sig_coeff_flag	0	0	0.
0x00002d05:4	436	370	sig_coeff_flag	0	0	0.
0x00002d05:4	397	370	sig_coeff_flag	1	3	1.
0x00002d05:7	296	82	sig_coeff_flag	0	0	0.
0x00002d05:7	257	82	sig_coeff_flag	0	1	0.
0x00002d06:0	440	164	sig_coeff_flag	0	0	0.
0x00002d06:0	389	164	sig_coeff_flag	1	1	0.
0x00002d06:1	422	329	sig_coeff_flag	0	1	0.
0x00002d06:2	338	153	sig_coeff_flag	1	1	0.
0x00002d06:3	360	306	sig_coeff_flag	0	1	0.
0x00002d06:4	300	192	sig_coeff_flag	0	1	0.
0x00002d06:5	256	41	sig_coeff_flag	1	1	0.

Screenshots of VQ Analyzer software analysis of UVG H.265 test bitstream.

414. When encoding Amazon Prime Video trailers, on information and belief, Amazon produces a bitstream that is the same or substantially similar to the encoded UVG H.265 test

bitstreams by performing joining, by a processor, the first and second representations to form a coded prediction error signal, as demonstrated in the screenshots below using VQ Analyzer software.

Syntax Info

Filter | Hex

Pos	R	V	SE Name	Value	Bits	Bits, %	Decoding type
0x00002d00:7			**sao(5, 10)**		**15**	**2.43**	
0x00002d00:7	329	326	sao_merge_left_flag	0	1	6.67	Arithmetic
0x00002d01:0	256	251	sao_merge_up_flag	0	2	13.33	Arithmetic
0x00002d01:2	464	446	sao_type_idx_luma	2	3	20.00	Arithmetic
0x00002d01:5	304	162	sao_offset_abs[0][0]	1	2	13.33	Bypass
0x00002d01:7	304	40	sao_offset_abs[0][1]	0	1	6.67	Bypass
0x00002d02:0	304	80	sao_offset_abs[0][2]	0	1	6.67	Bypass
0x00002d02:1	304	160	sao_offset_abs[0][3]	1	2	13.33	Bypass
0x00002d02:3	304	32	sao_eo_class_luma	0	2	13.33	Bypass
0x00002d02:5	304	129	sao_type_idx_chroma	0	1	6.67	Arithmetic
0x00002d02:6			**coding_quadtree(320, 640, 6, 0)**		**603**	**97.57**	
0x00002d4e:1	492	316	end_of_slice_segment_flag	0	0	0.00	Termination

Screenshots of VQ Analyzer software analysis of UVG H.265 test bitstream.

Syntax Info | 10232

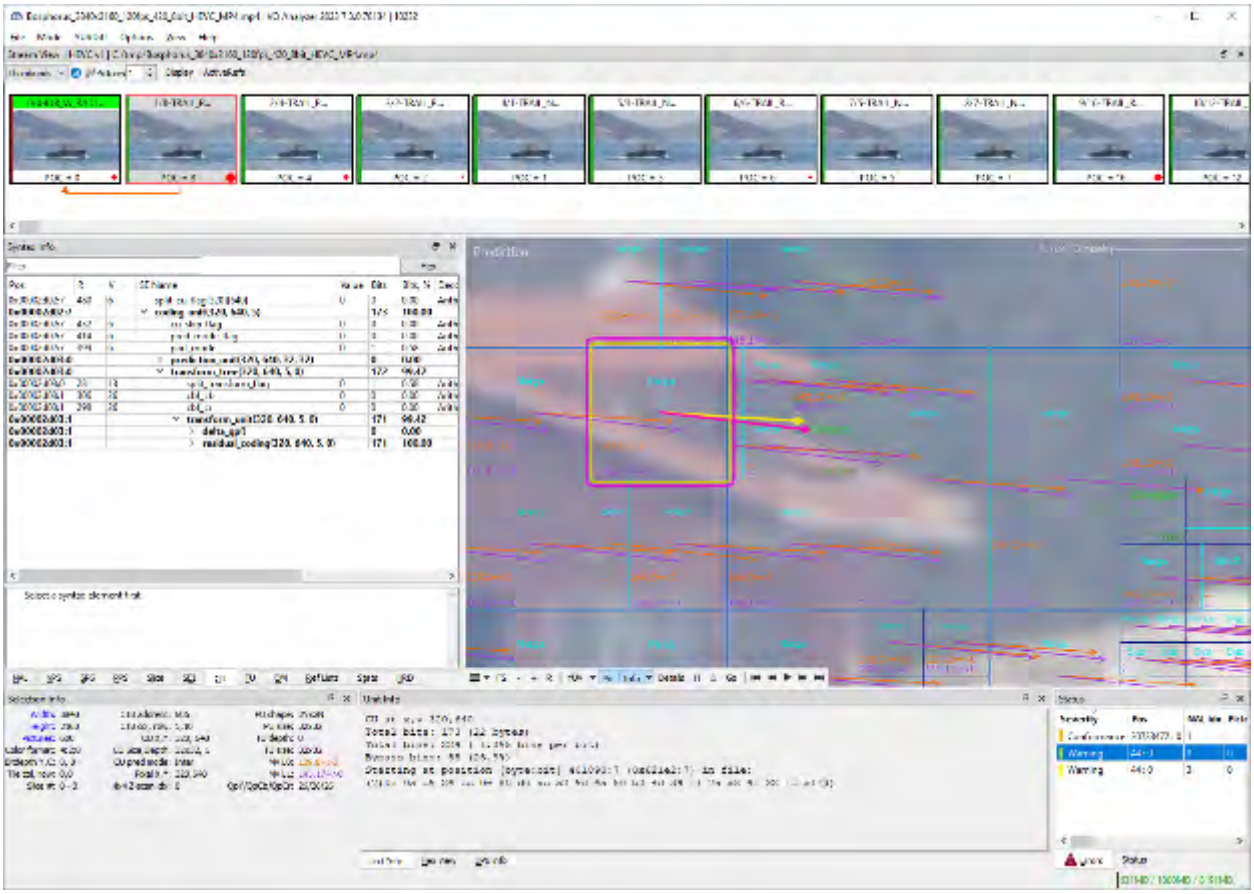
Filter | Hex

Pos	R	V	SE Name	Value	Bits	B
0x00002d00:7			**sao(5, 10)**		**15**	**2.**
0x00002d02:6			**coding_quadtree(320, 640, 6, 0)**		**603**	**9**
0x00002d02:6	496	259	split_cu_flag[320][640]	1	1	0.
0x00002d02:7			**coding_quadtree(320, 640, 5, 1)**		**173**	**2**
0x00002d02:7	480	6	split_cu_flag[320][640]	0	0	0.
0x00002d02:7			**coding_unit(320, 640, 5)**		**173**	**1**
0x00002d02:7	432	6	cu_skip_flag	0	0	0.
0x00002d02:7	414	6	pred_mode_flag	0	0	0.
0x00002d02:7	399	6	part_mode	0	1	0.
0x00002d03:0			**prediction_unit(320, 640, 32, 32)**		**0**	**0.**
0x00002d03:0			**transform_tree(320, 640, 5, 0)**		**172**	**9**
0x00002d03:0	281	13	split_transform_flag	0	1	0.
0x00002d03:1	306	26	cbf_cb	0	0	0.
0x00002d03:1	298	26	cbf_cr	0	0	0.
0x00002d03:1			**transform_unit(320, 640, 5, 0)**		**171**	**9**
0x00002d03:1			**delta_qp()**		**0**	**0.**
0x00002d03:1			**residual_coding(320, 640, 5, 0)**		**171**	**1**
0x00002d03:1	285	26	last_sig_coeff_x_prefix	4	3	1.
0x00002d03:4	454	208	last_sig_coeff_y_prefix	6	3	1.
0x00002d03:7	289	43	last_sig_coeff_x_suffix	0	1	0.
0x00002d04:0	289	86	last_sig_coeff_y_suffix	1	2	1.
0x00002d04:2	289	58	sig_coeff_flag	1	1	0.
0x00002d04:3	322	116	coeff_abs_level_greater1_flag	0	0	0.
0x00002d04:3	315	116	coeff_abs_level_greater1_flag	0	0	0.
0x00002d04:3	300	116	coeff_sign_flag	0	1	0.
0x00002d04:4	300	232	coeff_sign_flag	1	1	0.
0x00002d04:5	300	165	coded_sub_block_flag	0	1	0.
0x00002d04:6	344	330	coded_sub_block_flag	1	1	0.
0x00002d04:7	316	288	sig_coeff_flag	0	0	0.
0x00002d04:7	296	288	sig_coeff_flag	1	4	2.
0x00002d05:3	304	185	sig_coeff_flag	0	0	0.
0x00002d05:3	272	185	sig_coeff_flag	0	1	0.
0x00002d05:4	484	370	sig_coeff_flag	0	0	0.
0x00002d05:4	436	370	sig_coeff_flag	0	0	0.
0x00002d05:4	397	370	sig_coeff_flag	1	3	1.
0x00002d05:7	296	82	sig_coeff_flag	0	0	0.
0x00002d05:7	257	82	sig_coeff_flag	0	1	0.
0x00002d06:0	440	164	sig_coeff_flag	0	0	0.
0x00002d06:0	389	164	sig_coeff_flag	1	1	0.
0x00002d06:1	422	329	sig_coeff_flag	0	1	0.
0x00002d06:2	338	153	sig_coeff_flag	1	1	0.
0x00002d06:3	360	306	sig_coeff_flag	0	1	0.
0x00002d06:4	300	192	sig_coeff_flag	0	1	0.
0x00002d06:5	256	41	sig_coeff_flag	1	1	0.

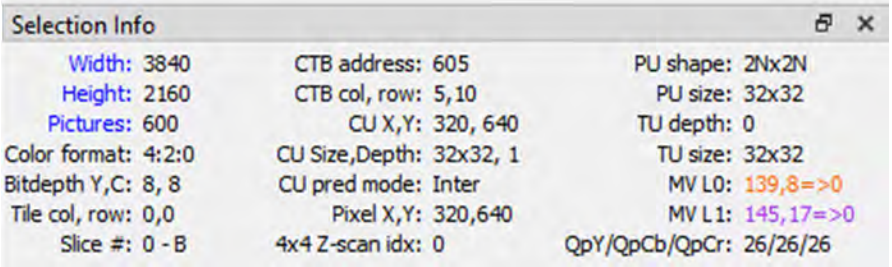
Screenshots of VQ Analyzer software analysis of UVG H.265 test bitstream.

415. When encoding Amazon Prime Video trailers, on information and belief, Amazon produces a bitstream that is the same or substantially similar to the encoded UVG H.265 test

bitstreams by performing at least one of transmitting or storing the coded prediction error signal, as demonstrated in the screenshots below using VQ Analyzer software.



Screenshots of VQ Analyzer software analysis of UVG H.265 test bitstream.



Screenshots of VQ Analyzer software analysis of UVG H.265 test bitstream.

Syntax Info

Filter | Hex

Pos	R	V	SE Name	Value	Bits	Bits, %
0x00002d02:7	480	6	split_cu_flag[320][640]	0	0	0.00
0x00002d02:7			**coding_unit(320, 640, 5)**		**173**	**100.00**
0x00002d02:7	432	6	cu_skip_flag	0	0	0.00
0x00002d02:7	414	6	pred_mode_flag	0	0	0.00
0x00002d02:7	399	6	part_mode	0	1	0.58
0x00002d03:0			**prediction_unit(320, 640, 32, 32)**		**0**	**0.00**
0x00002d03:0			**transform_tree(320, 640, 5, 0)**		**172**	**99.42**
0x00002d03:0	281	13	split_transform_flag	0	1	0.58
0x00002d03:1	306	26	cbf_cb	0	0	0.00
0x00002d03:1	298	26	cbf_cr	0	0	0.00
0x00002d03:1			**transform_unit(320, 640, 5, 0)**		**171**	**99.42**
0x00002d03:1			**delta_qp()**		**0**	**0.00**
0x00002d03:1			**residual_coding(320, 640, 5, 0)**		**171**	**100.00**
0x00002d03:1	285	26	last_sig_coeff_x_prefix	4	3	1.75
0x00002d03:4	454	208	last_sig_coeff_y_prefix	6	3	1.75
0x00002d03:7	289	43	last_sig_coeff_x_suffix	0	1	0.58
0x00002d04:0	289	86	last_sig_coeff_y_suffix	1	2	1.17
0x00002d04:2	289	58	sig_coeff_flag	1	1	0.58
0x00002d04:3	322	116	coeff_abs_level_greater1_flag	0	0	0.00
0x00002d04:3	315	116	coeff_abs_level_greater1_flag	0	0	0.00
0x00002d04:3	300	116	coeff_sign_flag	0	1	0.58
0x00002d04:4	300	232	coeff_sign_flag	1	1	0.58
0x00002d04:5	300	165	coded_sub_block_flag	0	1	0.58

Select a syntax element first

NAL | VPS | SPS | PPS | Slice | SEI | CU | TU | QM | Ref Lists | Stats | HRD

Screenshots of VQ Analyzer software analysis of UVG H.265 test bitstream.

416. For another example, the well-known H.265 Reference Software, available at https://vcgit.hhi.fraunhofer.de/jvet/HM, utilizes a method that infringes one or more claims of the ’991 Patent, including, for example, claim 1.

417. When encoding Amazon Prime Video trailers, on information and belief, Amazon produces a bitstream using a method substantially similar to that which is utilized by the H.265 Reference Software shown below to perform calculating, by a processor, a difference signal

representing differences between sample values of a predicted block of data and values for an original input block.

```
HM reference software for HEVC
==============================

This software package is the reference software for Rec. ITU-T H.265 | ISO/IEC 23008-2 High Efficiency Video Coding (HEVC). The reference
software includes both encoder and decoder functionality.

Reference software is useful in aiding users of a video coding standard to establish and test conformance and interoperability, and to
educate users and demonstrate the capabilities of the standard. For these purposes, this software is provided as an aid for the study and
implementation of Rec. ITU-T H.265 | ISO/IEC 23008-2 High Efficiency Video Coding.

The software has been jointly developed by the ITU-T Video Coding Experts Group (VCEG, Question 6 of ITU-T Study Group 16) and the ISO/IEC
Moving Picture Experts Group (MPEG, Working Group 11 of Subcommittee 29 of ISO/IEC Joint Technical Committee 1).

The software is maintained by the Joint Video Experts Team (JVET) which is a joint collaboration of ITU-T Video Coding Experts Group
(VCEG, Question 6 of ITU-T Study Group 16) and the ISO/IEC Moving Picture Experts Group (MPEG, Working Group 5 of Subcommittee 29 of
ISO/IEC Joint Technical Committee 1).
```

H.265 Reference Software at README.md.

```
1244    //===== get residual signal =====
1245    {
1246      // get residual
1247      Pel*  pOrg    = piOrg;
1248      Pel*  pPred   = piPred;
1249      Pel*  pResi   = piResi;
1250
1251      for( UInt uiY = 0; uiY < uiHeight; uiY++ )
1252      {
1253        for( UInt uiX = 0; uiX < uiWidth; uiX++ )
1254        {
1255          pResi[ uiX ] = pOrg[ uiX ] - pPred[ uiX ];
1256        }
1257
1258        pOrg  += uiStride;
1259        pResi += uiStride;
1260        pPred += uiStride;
1261      }
1262    }
```

H.265 Reference Software at TEncSearch.cpp, lines 1244-1262.

```
4589  //! encode residual and calculate rate-distortion for a CU block
4590  Void TEncSearch::encodeResAndCalcRdInterCU( TComDataCU* pcCU, TComYuv* pcYuvOrg, TComYuv* pcYuvPred,
4591                                              TComYuv* pcYuvResi, TComYuv* pcYuvResiBest, TComYuv* pcYuvRec,
4592                                              Bool bSkipResidual DEBUG_STRING_FN_DECLARE(sDebug) )
4593  {
4594    assert ( !pcCU->isIntra(0) );
4595
4596    const UInt cuWidthPixels      = pcCU->getWidth ( 0 );
4597    const UInt cuHeightPixels     = pcCU->getHeight( 0 );
4598    const Int  numValidComponents = pcCU->getPic()->getNumberValidComponents();
```

H.265 Reference Software at TEncSearch.cpp, lines 4590-4665; *see also*, *e.g.*, TEncSearch.cpp, lines 1130-1135, 1184-1193, 1263-1279, 1340-1349, 1431-1462, 1566-1575, 1622-1704.

418. When encoding Amazon Prime Video trailers, on information and belief, Amazon produces a bitstream using a method substantially similar to that which is utilized by the H.265

Reference Software shown below to perform, by a processor, transform coding to the difference signal, thereby creating a first representation of a first component of the difference signal.

```
199
200   Void TEncEntropy::xEncodeTransform( Bool& bCodeDQP, Bool& codeChromaQpAdj, TComTU &rTu )
201   {
202   //pcCU, absPartIdxCU, uiAbsPartIdx, uiDepth+1, uiTrIdx+1, quadrant,
203     TComDataCU *pcCU=rTu.getCU();
204     const UInt uiAbsPartIdx=rTu.GetAbsPartIdxTU();
205     const UInt numValidComponent = pcCU->getPic()->getNumberValidComponents();
206     const Bool bChroma = isChromaEnabled(pcCU->getPic()->getChromaFormat());
207     const UInt uiTrIdx = rTu.GetTransformDepthRel();
208     const UInt uiDepth = rTu.GetTransformDepthTotal();
209   #if ENVIRONMENT_VARIABLE_DEBUG_AND_TEST
210     const Bool bDebugRQT=pcCU->getSlice()->getFinalized() && DebugOptionList::DebugRQT.getInt()!=0;
211     if (bDebugRQT)
212     {
213       printf("x..codeTransform: offsetLuma=%d offsetChroma=%d absPartIdx=%d, uiDepth=%d\n width=%d, height=%d, uiTrIdx=%d,
214              rTu.getCoefficientOffset(COMPONENT_Y), rTu.getCoefficientOffset(COMPONENT_Cb), uiAbsPartIdx, uiDepth, rTu.getR
215     }
216   #endif
217     const UInt uiSubdiv = pcCU->getTransformIdx( uiAbsPartIdx ) > uiTrIdx;// + pcCU->getDepth( uiAbsPartIdx ) > uiDepth;
218     const UInt uiLog2TrafoSize = rTu.GetLog2LumaTrSize();
219
220
```

H.265 Reference Software at TEncEntropy.cpp, lines 199-220; *see also*, *e.g.*, TEncEntropy.cpp, lines 269-301.

```
1244    //===== get residual signal =====
1245    {
1246      // get residual
1247      Pel*  pOrg    = piOrg;
1248      Pel*  pPred   = piPred;
1249      Pel*  pResi   = piResi;
1250
1251      for( UInt uiY = 0; uiY < uiHeight; uiY++ )
1252      {
1253        for( UInt uiX = 0; uiX < uiWidth; uiX++ )
1254        {
1255          pResi[ uiX ] = pOrg[ uiX ] - pPred[ uiX ];
1256        }
1257
1258        pOrg  += uiStride;
1259        pResi += uiStride;
1260        pPred += uiStride;
1261      }
1262    }
```

H.265 Reference Software at TEncSearch.cpp, lines 1244-1262.

```
4589  //! encode residual and calculate rate-distortion for a CU block
4590  Void TEncSearch::encodeResAndCalcRdInterCU( TComDataCU* pcCU, TComYuv* pcYuvOrg, TComYuv* pcYuvPred,
4591                                              TComYuv* pcYuvResi, TComYuv* pcYuvResiBest, TComYuv* pcYuvRec,
4592                                              Bool bSkipResidual DEBUG_STRING_FN_DECLARE(sDebug) )
4593  {
4594    assert ( !pcCU->isIntra(0) );
4595
4596    const UInt cuWidthPixels      = pcCU->getWidth ( 0 );
4597    const UInt cuHeightPixels     = pcCU->getHeight( 0 );
4598    const Int  numValidComponents = pcCU->getPic()->getNumberValidComponents();
```

H.265 Reference Software at TEncSearch.cpp, lines 4590-4665; *see also*, *e.g.*, TEncSearch.cpp, lines 1130-1135, 1184-1193, 1263-1279, 1340-1349, 1431-1462, 1566-1575, 1622-1704.

419. When encoding Amazon Prime Video trailers, on information and belief, Amazon produces a bitstream using a method substantially similar to that which is utilized by the H.265 Reference Software shown below to perform, by a processor, spatial coding to the difference signal, thereby creating a second representation of a first component of the difference signal.

```
1651        const Int  maxOffsetQVal = TComSampleAdaptiveOffset::getMaxOffsetQVal(channelBitDepth);
1652        for(Int i=0; i< 4; i++)
1653        {
1654          codeSaoMaxUvlc((offset[i]<0)?(-offset[i]):(offset[i]),  maxOffsetQVal ); //sao_offset_abs
1655        }
```

H.265 Reference Software at TEncSbac.cpp, lines 1651-1655.

```
1683 Void TEncSbac::codeSAOBlkParam(SAOBlkParam& saoBlkParam, const BitDepths &bitDepths
1684                                , Bool* sliceEnabled
1685                                , Bool leftMergeAvail
1686                                , Bool aboveMergeAvail
1687                                , Bool onlyEstMergeInfo // = false
1688                                )
1689 {
1690
1691   Bool isLeftMerge = false;
1692   Bool isAboveMerge= false;
1693
1694   if(leftMergeAvail)
1695   {
1696     isLeftMerge = ((saoBlkParam[COMPONENT_Y].modeIdc == SAO_MODE_MERGE) && (saoBlkParam[COMPONENT_Y].typeIdc == SAO_MERGE_LEFT));
1697     codeSaoMerge( isLeftMerge?1:0  ); //sao_merge_left_flag
1698   }
1699
1700   if( aboveMergeAvail && !isLeftMerge)
1701   {
1702     isAboveMerge = ((saoBlkParam[COMPONENT_Y].modeIdc == SAO_MODE_MERGE) && (saoBlkParam[COMPONENT_Y].typeIdc == SAO_MERGE_ABOVE));
1703     codeSaoMerge( isAboveMerge?1:0  ); //sao_merge_left_flag
1704   }
```

H.265 Reference Software at TEncSbac.cpp, lines 1683-1704; *see also*, *e.g.*, TEncSbac.cpp, lines 82-202, 1538-1718; TEncSampleAdaptiveOffset.cpp, lines 76-1290; TAppEncCfg.cpp at lines 677-970, 1819-1837, 2370-2375, 3369-3374, 3437-3442.

```
1244    //===== get residual signal =====
1245    {
1246      // get residual
1247      Pel*  pOrg    = piOrg;
1248      Pel*  pPred   = piPred;
1249      Pel*  pResi   = piResi;
1250
1251      for( UInt uiY = 0; uiY < uiHeight; uiY++ )
1252      {
1253        for( UInt uiX = 0; uiX < uiWidth; uiX++ )
1254        {
1255          pResi[ uiX ] = pOrg[ uiX ] - pPred[ uiX ];
1256        }
1257
1258        pOrg  += uiStride;
1259        pResi += uiStride;
1260        pPred += uiStride;
1261      }
1262    }
```

H.265 Reference Software at TEncSearch.cpp, lines 1244-1262.

```
4589  //! encode residual and calculate rate-distortion for a CU block
4590  Void TEncSearch::encodeResAndCalcRdInterCU( TComDataCU* pcCU, TComYuv* pcYuvOrg, TComYuv* pcYuvPred,
4591                                              TComYuv* pcYuvResi, TComYuv* pcYuvResiBest, TComYuv* pcYuvRec,
4592                                              Bool bSkipResidual DEBUG_STRING_FN_DECLARE(sDebug) )
4593  {
4594    assert ( !pcCU->isIntra(0) );
4595
4596    const UInt cuWidthPixels      = pcCU->getWidth ( 0 );
4597    const UInt cuHeightPixels     = pcCU->getHeight( 0 );
4598    const Int  numValidComponents = pcCU->getPic()->getNumberValidComponents();
```

H.265 Reference Software at TEncSearch.cpp, lines 4590-4665; *see also*, *e.g.*, TEncSearch.cpp, lines 1130-1135, 1184-1193, 1263-1279, 1340-1349, 1431-1462, 1566-1575, 1622-1704.

420. When encoding Amazon Prime Video trailers, on information and belief, Amazon produces a bitstream using a method substantially similar to that which is utilized by the H.265 Reference Software shown below to perform joining, by a processor, the first and second representations to form a coded prediction error signal.

```
199
200  Void TEncEntropy::xEncodeTransform( Bool& bCodeDQP, Bool& codeChromaQpAdj, TComTU &rTu )
201  {
202  //pcCU, absPartIdxCU, uiAbsPartIdx, uiDepth+1, uiTrIdx+1, quadrant,
203    TComDataCU *pcCU=rTu.getCU();
204    const UInt uiAbsPartIdx=rTu.GetAbsPartIdxTU();
205    const UInt numValidComponent = pcCU->getPic()->getNumberValidComponents();
206    const Bool bChroma = isChromaEnabled(pcCU->getPic()->getChromaFormat());
207    const UInt uiTrIdx = rTu.GetTransformDepthRel();
208    const UInt uiDepth = rTu.GetTransformDepthTotal();
209  #if ENVIRONMENT_VARIABLE_DEBUG_AND_TEST
210    const Bool bDebugRQT=pcCU->getSlice()->getFinalized() && DebugOptionList::DebugRQT.getInt()!=0;
211    if (bDebugRQT)
212    {
213      printf("x..codeTransform: offsetLuma=%d offsetChroma=%d absPartIdx=%d, uiDepth=%d\n width=%d, height=%d, uiTrIdx=%d,
214             rTu.getCoefficientOffset(COMPONENT_Y), rTu.getCoefficientOffset(COMPONENT_Cb), uiAbsPartIdx, uiDepth, rTu.getR
215    }
216  #endif
217    const UInt uiSubdiv = pcCU->getTransformIdx( uiAbsPartIdx ) > uiTrIdx;// + pcCU->getDepth( uiAbsPartIdx ) > uiDepth;
218    const UInt uiLog2TrafoSize = rTu.GetLog2LumaTrSize();
219
220
```

H.265 Reference Software at TEncEntropy.cpp, lines 199-220; *see also*, *e.g.*, TEncEntropy.cpp, lines 269-301.

```
1651    const Int  maxOffsetQVal = TComSampleAdaptiveOffset::getMaxOffsetQVal(channelBitDepth);
1652    for(Int i=0; i< 4; i++)
1653    {
1654      codeSaoMaxUvlc((offset[i]<0)?(-offset[i]):(offset[i]),  maxOffsetQVal ); //sao_offset_abs
1655    }
```

H.265 Reference Software at TEncSbac.cpp, lines 1651-1655.

```
1683 Void TEncSbac::codeSAOBlkParam(SAOBlkParam& saoBlkParam, const BitDepths &bitDepths
1684                                , Bool* sliceEnabled
1685                                , Bool leftMergeAvail
1686                                , Bool aboveMergeAvail
1687                                , Bool onlyEstMergeInfo // = false
1688                                )
1689 {
1690
1691   Bool isLeftMerge = false;
1692   Bool isAboveMerge= false;
1693
1694   if(leftMergeAvail)
1695   {
1696     isLeftMerge = ((saoBlkParam[COMPONENT_Y].modeIdc == SAO_MODE_MERGE) && (saoBlkParam[COMPONENT_Y].typeIdc == SAO_MERGE_LEFT));
1697     codeSaoMerge( isLeftMerge?1:0  ); //sao_merge_left_flag
1698   }
1699
1700   if( aboveMergeAvail && !isLeftMerge)
1701   {
1702     isAboveMerge = ((saoBlkParam[COMPONENT_Y].modeIdc == SAO_MODE_MERGE) && (saoBlkParam[COMPONENT_Y].typeIdc == SAO_MERGE_ABOVE));
1703     codeSaoMerge( isAboveMerge?1:0  ); //sao_merge_left_flag
1704   }
```

H.265 Reference Software at TEncSbac.cpp, lines 1683-1704; *see also*, *e.g.*, TEncSbac.cpp, lines 82-202, 1538-1718; TEncSampleAdaptiveOffset.cpp, lines 76-1290; TAppEncCfg.cpp at lines 677-970, 1819-1837, 2370-2375, 3369-3374, 3437-3442.

```
1244   //===== get residual signal =====
1245   {
1246     // get residual
1247     Pel*  pOrg    = piOrg;
1248     Pel*  pPred   = piPred;
1249     Pel*  pResi   = piResi;
1250
1251     for( UInt uiY = 0; uiY < uiHeight; uiY++ )
1252     {
1253       for( UInt uiX = 0; uiX < uiWidth; uiX++ )
1254       {
1255         pResi[ uiX ] = pOrg[ uiX ] - pPred[ uiX ];
1256       }
1257
1258       pOrg  += uiStride;
1259       pResi += uiStride;
1260       pPred += uiStride;
1261     }
1262   }
```

H.265 Reference Software at TEncSearch.cpp, lines 1244-1262.

```
4589 //! encode residual and calculate rate-distortion for a CU block
4590 Void TEncSearch::encodeResAndCalcRdInterCU( TComDataCU* pcCU, TComYuv* pcYuvOrg, TComYuv* pcYuvPred,
4591                                            TComYuv* pcYuvResi, TComYuv* pcYuvResiBest, TComYuv* pcYuvRec,
4592                                            Bool bSkipResidual DEBUG_STRING_FN_DECLARE(sDebug) )
4593 {
4594   assert ( !pcCU->isIntra(0) );
4595
4596   const UInt cuWidthPixels      = pcCU->getWidth ( 0 );
4597   const UInt cuHeightPixels     = pcCU->getHeight( 0 );
4598   const Int  numValidComponents = pcCU->getPic()->getNumberValidComponents();
```

H.265 Reference Software at TEncSearch.cpp, lines 4590-4665; *see also*, *e.g.*, TEncSearch.cpp, lines 1130-1135, 1184-1193, 1263-1279, 1340-1349, 1431-1462, 1566-1575, 1622-1704.

421. When encoding Amazon Prime Video trailers, on information and belief, Amazon produces a bitstream using a method substantially similar to that which is utilized by the H.265 Reference Software shown below to perform at least one of transmitting or storing the coded prediction error signal.

```
199
200   Void TEncEntropy::xEncodeTransform( Bool& bCodeDQP, Bool& codeChromaQpAdj, TComTU &rTu )
201   {
202   //pcCU, absPartIdxCU, uiAbsPartIdx, uiDepth+1, uiTrIdx+1, quadrant,
203     TComDataCU *pcCU=rTu.getCU();
204     const UInt uiAbsPartIdx=rTu.GetAbsPartIdxTU();
205     const UInt numValidComponent = pcCU->getPic()->getNumberValidComponents();
206     const Bool bChroma = isChromaEnabled(pcCU->getPic()->getChromaFormat());
207     const UInt uiTrIdx = rTu.GetTransformDepthRel();
208     const UInt uiDepth = rTu.GetTransformDepthTotal();
209   #if ENVIRONMENT_VARIABLE_DEBUG_AND_TEST
210     const Bool bDebugRQT=pcCU->getSlice()->getFinalized() && DebugOptionList::DebugRQT.getInt()!=0;
211     if (bDebugRQT)
212     {
213       printf("x..codeTransform: offsetLuma=%d offsetChroma=%d absPartIdx=%d, uiDepth=%d\n width=%d, height=%d, uiTrIdx=%d,
214              rTu.getCoefficientOffset(COMPONENT_Y), rTu.getCoefficientOffset(COMPONENT_Cb), uiAbsPartIdx, uiDepth, rTu.getR
215     }
216   #endif
217     const UInt uiSubdiv = pcCU->getTransformIdx( uiAbsPartIdx ) > uiTrIdx;// + pcCU->getDepth( uiAbsPartIdx ) > uiDepth;
218     const UInt uiLog2TrafoSize = rTu.GetLog2LumaTrSize();
219
220
```

H.265 Reference Software at TEncEntropy.cpp, lines 199-220; *see also*, *e.g.*, TEncEntropy.cpp, lines 269-301.

```
1651      const Int  maxOffsetQVal = TComSampleAdaptiveOffset::getMaxOffsetQVal(channelBitDepth);
1652      for(Int i=0; i< 4; i++)
1653      {
1654        codeSaoMaxUvlc((offset[i]<0)?(-offset[i]):(offset[i]),  maxOffsetQVal ); //sao_offset_abs
1655      }
```

H.265 Reference Software at TEncSbac.cpp, lines 1651-1655.

```
1683  Void TEncSbac::codeSAOBlkParam(SAOBlkParam& saoBlkParam, const BitDepths &bitDepths
1684                                 , Bool* sliceEnabled
1685                                 , Bool leftMergeAvail
1686                                 , Bool aboveMergeAvail
1687                                 , Bool onlyEstMergeInfo // = false
1688                                 )
1689  {
1690
1691    Bool isLeftMerge = false;
1692    Bool isAboveMerge= false;
1693
1694    if(leftMergeAvail)
1695    {
1696      isLeftMerge = ((saoBlkParam[COMPONENT_Y].modeIdc == SAO_MODE_MERGE) && (saoBlkParam[COMPONENT_Y].typeIdc == SAO_MERGE_LEFT));
1697      codeSaoMerge( isLeftMerge?1:0  ); //sao_merge_left_flag
1698    }
1699
1700    if( aboveMergeAvail && !isLeftMerge)
1701    {
1702      isAboveMerge = ((saoBlkParam[COMPONENT_Y].modeIdc == SAO_MODE_MERGE) && (saoBlkParam[COMPONENT_Y].typeIdc == SAO_MERGE_ABOVE));
1703      codeSaoMerge( isAboveMerge?1:0  ); //sao_merge_left_flag
1704    }
```

H.265 Reference Software at TEncSbac.cpp, lines 1683-1704; *see also*, *e.g.*, TEncSbac.cpp, lines 82-202, 1538-1718; TEncSampleAdaptiveOffset.cpp, lines 76-1290; TAppEncCfg.cpp at lines 677-970, 1819-1837, 2370-2375, 3369-3374, 3437-3442.

```
1244    //===== get residual signal =====
1245    {
1246      // get residual
1247      Pel*  pOrg    = piOrg;
1248      Pel*  pPred   = piPred;
1249      Pel*  pResi   = piResi;
1250
1251      for( UInt uiY = 0; uiY < uiHeight; uiY++ )
1252      {
1253        for( UInt uiX = 0; uiX < uiWidth; uiX++ )
1254        {
1255          pResi[ uiX ] = pOrg[ uiX ] - pPred[ uiX ];
1256        }
1257
1258        pOrg  += uiStride;
1259        pResi += uiStride;
1260        pPred += uiStride;
1261      }
1262    }
```

H.265 Reference Software at TEncSearch.cpp, lines 1244-1262.

```
4589  //! encode residual and calculate rate-distortion for a CU block
4590  Void TEncSearch::encodeResAndCalcRdInterCU( TComDataCU* pcCU, TComYuv* pcYuvOrg, TComYuv* pcYuvPred,
4591                                              TComYuv* pcYuvResi, TComYuv* pcYuvResiBest, TComYuv* pcYuvRec,
4592                                              Bool bSkipResidual DEBUG_STRING_FN_DECLARE(sDebug) )
4593  {
4594    assert ( !pcCU->isIntra(0) );
4595
4596    const UInt cuWidthPixels      = pcCU->getWidth ( 0 );
4597    const UInt cuHeightPixels     = pcCU->getHeight( 0 );
4598    const Int  numValidComponents = pcCU->getPic()->getNumberValidComponents();
```

H.265 Reference Software at TEncSearch.cpp, lines 4590-4665; *see also*, *e.g.*, TEncSearch.cpp, lines 1130-1135, 1184-1193, 1263-1279, 1340-1349, 1431-1462, 1566-1575, 1622-1704.

422. The UVG H.265 test bitstreams and H.265 Reference Software are just two examples of well-known techniques used to encode bitstreams that can be decoded by an H.265-compliant decoder. Both utilize techniques that are covered by the claims of the ’991 Patent, such that on information and belief, when encoding Amazon Prime Video content, Amazon performs a method of encoding that infringes one or more claims of the ’991 Patent, including, for example, claim 1, by using the same or similar currently most popular and readily available techniques used by multiple other well-known encoding methods, which infringe the ’991 Patent.

423. For another example, on information and belief, Amazon performs a method of encoding video in a manner that is substantially the same as described above, just as multiple other well-known encoding methods do, for Amazon.com videos, such that it is covered by claim 1 of the ’991 Patent.

M. Amazon Makes, Imports, Uses, Sells, and/or Offers for Sale Products and Services that Infringe the ’833 Patent

424. The Accused Products infringe one or more claims of the ’833 Patent, including, for example, claim 9.

425. As just one example of infringement, on information and belief, Amazon performs a method of decoding video in a manner that is covered by claim 9 of the ’833 Patent for Amazon Prime Video contents, when Amazon Prime Video contents are decoded by an H.265-compliant decoder, and as performed, for example, during internal testing of video quality and transcoding of Amazon Prime Video contents. *See*, *e.g.*, https://www.amazon.science/blog/how-prime-video-uses-machine-learning-to-ensure-video-quality.

Screenshots of Amazon Prime Video indicating “Codec: hevc.”

426. When decoding Amazon Prime Video contents, on information and belief, Amazon performs a method of decoding that performs receiving an encoded block of pixels including a prediction unit and information identifying a respective spatial motion vector prediction candidate from a merge list constructed by an encoder, corresponding to the decoding process specified by the H.265 standard.

3 Definitions

For the purposes of this Recommendation | International Standard, the following definitions apply.

. . .

3.153 syntax element: An element of data represented in the *bitstream*.

3.154 syntax structure: Zero or more *syntax elements* present together in the *bitstream* in a specified order.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 5, 7-12.

5.10 Variables, syntax elements and tables

Syntax elements in the bitstream are represented in **bold** type. Each syntax element is described by its name (all lower case letters with underscore characters), and one descriptor for its method of coded representation. The decoding process behaves according to the value of the syntax element and to the values of previously decoded syntax elements. When a value of a syntax element is used in the syntax tables or the text, it appears in regular (i.e., not bold) type.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 18.

7 Syntax and semantics

7.1 Method of specifying syntax in tabular form

The syntax tables specify a superset of the syntax of all allowed bitstreams . . .

. . .

. . . When **syntax_element** appears, it specifies that a syntax element is parsed from the bitstream . . .

. . .

7.2 Specification of syntax functions and descriptors

. . .

The following descriptors specify the parsing process of each syntax element:

– ae(v): context-adaptive arithmetic entropy-coded syntax element. The parsing process for this descriptor is specified in clause 9.3.

. . .

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 30-31.

7.3.8.6 Prediction unit syntax

prediction_unit(x0, y0, nPbW, nPbH) {	Descriptor
if(cu_skip_flag[x0][y0]) {	
if(MaxNumMergeCand > 1)	
merge_idx[x0][y0]	ae(v)
} else { /* MODE_INTER */	
merge_flag[x0][y0]	ae(v)
if(merge_flag[x0][y0]) {	
if(MaxNumMergeCand > 1)	
merge_idx[x0][y0]	ae(v)
} else {	
if(slice_type == B)	
inter_pred_idc[x0][y0]	ae(v)
if(inter_pred_idc[x0][y0] != PRED_L1) {	
if(num_ref_idx_l0_active_minus1 > 0)	
ref_idx_l0[x0][y0]	ae(v)
mvd_coding(x0, y0, 0)	
mvp_l0_flag[x0][y0]	ae(v)
}	
if(inter_pred_idc[x0][y0] != PRED_L0) {	
if(num_ref_idx_l1_active_minus1 > 0)	
ref_idx_l1[x0][y0]	ae(v)
if(mvd_l1_zero_flag && inter_pred_idc[x0][y0] == PRED_BI) {	
MvdL1[x0][y0][0] = 0	
MvdL1[x0][y0][1] = 0	
} else	
mvd_coding(x0, y0, 1)	
mvp_l1_flag[x0][y0]	ae(v)
}	
}	
}	
}	

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 55.

7.4.9.6 Prediction unit semantics

. . .

merge_idx[x0][y0] specifies the merging candidate index of the merging candidate list where x0, y0 specify the location (x0, y0) of the top-left luma sample of the considered prediction block relative to the top-left luma sample of the picture.

When merge_idx[x0][y0] is not present, it is inferred to be equal to 0.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 107.

8.5.3.2.2 Derivation process for luma motion vectors for merge mode

. . .

Outputs of this process are as follows, with X being 0 or 1:

. . .

- the reference indices $refIdxLXA_0$, $refIdxLXA_1$, $refIdxLXB_0$, $refIdxLXB_1$ and $refIdxLXB_2$ of the neighbouring prediction units,

. . .

- the motion vectors $mvLXA_0$, $mvLXA_1$, $mvLXB_0$, $mvLXB_1$ and $mvLXB_2$ of the neighbouring prediction units.

. . .

9. The following assignments are made with N being the candidate at position merge_idx[xOrigP][yOrigP] in the merging candidate list mergeCandList (N = mergeCandList[merge_idx[xOrigP][yOrigP]]) and X being replaced by 0 or 1:

refIdxLX = refIdxLXN (8-120)

predFlagLX = predFlagLXN (8-121)

. . .

mvLX[0] = mvLXN[0] (8-122)

mvLX[1] = mvLXN[1] (8-123)

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 147-149.

9 Parsing process

9.1 General

Inputs to this process are bits …

Outputs of this process are syntax element values.

This process is invoked when the descriptor of a syntax element in the syntax tables in clause 7.3 is equal to ue(v), se(v) (see clause 9.2), or ae(v) (see clause 9.3).

9.3 CABAC parsing process for slice segment data

9.3.1 General

This process is invoked when parsing syntax elements with descriptor ae(v) in clauses 7.3.8.1 through 7.3.8.11.

Inputs to this process are a request for a value of a syntax element and values of prior parsed syntax elements.

Output of this process is the value of the syntax element.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 201.

427. When decoding Amazon Prime Video contents, on information and belief, Amazon performs a method of decoding that performs determining a set of spatial motion vector prediction candidates for the encoded block of pixels; the spatial motion vector prediction candidates being provided with motion information, corresponding to the decoding process specified by the H.265 standard.

3 Definitions

For the purposes of this Recommendation | International Standard, the following definitions apply.

. . .

3.12 bitstream: A sequence of bits, **. . .** , that forms the representation of *coded pictures* and associated data forming one or more coded video sequences *(CVSs)*.

. . .

3.25 coded picture: A *coded representation* of a *picture* **. . .**

. . .

3.44 decoding process: The process specified in this Specification that reads a *bitstream* and derives *decoded pictures* from it.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 4 – 7.

8.5.3.2.3 Derivation process for spatial merging candidates

. . .

For the derivation of $availableFlagB_1$, $refIdxLXB_1$, $predFlagLXB_1$ and $mvLXB_1$ the following applies:

– The luma location ($xNbB_1$, $yNbB_1$) inside the neighbouring luma coding block is set equal to (xPb + nPbW − 1, yPb − 1).

– The availability derivation process for a prediction block as specified in clause 6.4.2 is invoked with the luma location (xCb, yCb), the current luma coding block size nCbS, the luma prediction block location (xPb, yPb), the luma prediction block width nPbW, the luma prediction block height nPbH, the luma location ($xNbB_1$, $yNbB_1$) and the partition index partIdx as inputs, and the output is assigned to the prediction block availability flag $availableB_1$.

. . .

For the derivation of $availableFlagB_0$, $refIdxLXB_0$, $predFlagLXB_0$ and $mvLXB_0$ the following applies:

– The luma location ($xNbB_0$, $yNbB_0$) inside the neighbouring luma coding block is set equal to (xPb + nPbW, yPb − 1).

– The availability derivation process for a prediction block as specified in clause 6.4.2 is invoked with the luma location (xCb, yCb), the current luma coding block size nCbS, the luma prediction block location (xPb, yPb), the luma prediction block width nPbW, the luma prediction block height nPbH, the luma location ($xNbB_0$, $yNbB_0$) and the partition index partIdx as inputs, and the output is assigned to the prediction block availability flag $availableB_0$.

. . .

For the derivation of $availableFlagA_0$, $refIdxLXA_0$, $predFlagLXA_0$ and $mvLXA_0$ the following applies:

– The luma location ($xNbA_0$, $yNbA_0$) inside the neighbouring luma coding block is set equal to (xPb − 1, yPb + nPbH).

– The availability derivation process for a prediction block as specified in clause 6.4.2 is invoked with the luma location (xCb, yCb), the current luma coding block size nCbS, the luma prediction block location (xPb, yPb), the luma prediction block width nPbW, the luma prediction block height nPbH, the luma location ($xNbA_0$, $yNbA_0$) and the partition index partIdx as inputs, and the output is assigned to the prediction block availability flag $availableA_0$.

. . .

For the derivation of $availableFlagB_2$, $refIdxLXB_2$, $predFlagLXB_2$ and $mvLXB_2$ the following applies:

– The luma location ($xNbB_2$, $yNbB_2$) inside the neighbouring luma coding block is set equal to (xPb − 1, yPb − 1).

– The availability derivation process for a prediction block as specified in clause 6.4.2 is invoked with the luma location (xCb, yCb), the current luma coding block size nCbS, the luma prediction block location (xPb, yPb), the luma prediction block width nPbW, the luma prediction block height nPbH, the luma location ($xNbB_2$, $yNbB_2$) and the partition index partIdx as inputs, and the output is assigned to the prediction block availability flag $availableB_2$.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 150-152.

8.5.3.2.7 Derivation process for motion vector predictor candidates

. . .

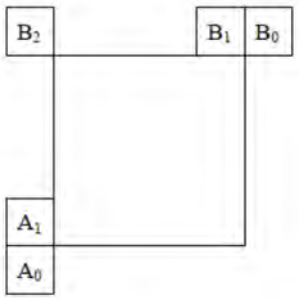


Figure 8-3 – Spatial motion vector neighbours (informative)

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 156.

428. When decoding Amazon Prime Video contents, on information and belief, Amazon performs a method of decoding that performs selecting a first spatial motion vector prediction candidate from the set of spatial motion vector prediction candidates as a potential spatial motion vector prediction candidate to be included in a merge list for the prediction unit, where the merge list is constructed based on the motion information of the spatial motion vector prediction candidates, corresponding to the decoding process specified by the H.265 standard.

8.5.3.2.3 Derivation process for spatial merging candidates

. . .

For the derivation of $availableFlagB_1$, $refIdxLXB_1$, $predFlagLXB_1$ and $mvLXB_1$ the following applies:

– The luma location ($xNbB_1$, $yNbB_1$) inside the neighbouring luma coding block is set equal to (xPb + nPbW − 1, yPb − 1).

– The availability derivation process for a prediction block as specified in clause 6.4.2 is invoked with the luma location (xCb, yCb), the current luma coding block size nCbS, the luma prediction block location (xPb, yPb), the luma prediction block width nPbW, the luma prediction block height nPbH, the luma location ($xNbB_1$, $yNbB_1$) and the partition index partIdx as inputs, and the output is assigned to the prediction block availability flag $availableB_1$.

. . .

For the derivation of $availableFlagB_0$, $refIdxLXB_0$, $predFlagLXB_0$ and $mvLXB_0$ the following applies:

– The luma location ($xNbB_0$, $yNbB_0$) inside the neighbouring luma coding block is set equal to (xPb + nPbW, yPb − 1).

– The availability derivation process for a prediction block as specified in clause 6.4.2 is invoked with the luma location (xCb, yCb), the current luma coding block size nCbS, the luma prediction block location (xPb, yPb), the luma prediction block width nPbW, the luma prediction block height nPbH, the luma location ($xNbB_0$, $yNbB_0$) and the partition index partIdx as inputs, and the output is assigned to the prediction block availability flag $availableB_0$.

. . .

For the derivation of $availableFlagA_0$, $refIdxLXA_0$, $predFlagLXA_0$ and $mvLXA_0$ the following applies:

– The luma location ($xNbA_0$, $yNbA_0$) inside the neighbouring luma coding block is set equal to (xPb − 1, yPb + nPbH).

– The availability derivation process for a prediction block as specified in clause 6.4.2 is invoked with the luma location (xCb, yCb), the current luma coding block size nCbS, the luma prediction block location (xPb, yPb), the luma prediction block width nPbW, the luma prediction block height nPbH, the luma location ($xNbA_0$, $yNbA_0$) and the partition index partIdx as inputs, and the output is assigned to the prediction block availability flag $availableA_0$.

. . .

For the derivation of $availableFlagB_2$, $refIdxLXB_2$, $predFlagLXB_2$ and $mvLXB_2$ the following applies:

– The luma location ($xNbB_2$, $yNbB_2$) inside the neighbouring luma coding block is set equal to (xPb − 1, yPb − 1).

– The availability derivation process for a prediction block as specified in clause 6.4.2 is invoked with the luma location (xCb, yCb), the current luma coding block size nCbS, the luma prediction block location (xPb, yPb), the luma prediction block width nPbW, the luma prediction block height nPbH, the luma location ($xNbB_2$, $yNbB_2$) and the partition index partIdx as inputs, and the output is assigned to the prediction block availability flag $availableB_2$.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 150-152.

8.5.3.2.7 Derivation process for motion vector predictor candidates

. . .

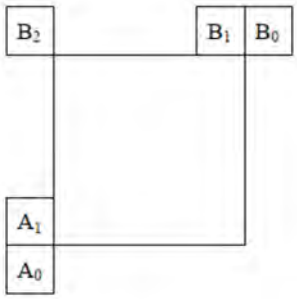


Figure 8-3 – Spatial motion vector neighbours (informative)

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 156.

429. When decoding Amazon Prime Video contents, on information and belief, Amazon performs a method of decoding that performs determining a subset of spatial motion vector prediction candidates based on the location of the block associated with the first spatial motion vector prediction candidate, corresponding to the decoding process specified by the H.265 standard.

8.5.3.2.3 Derivation process for spatial merging candidates

. . .

For the derivation of $availableFlagB_1$, $refIdxLXB_1$, $predFlagLXB_1$ and $mvLXB_1$ the following applies:

– The luma location ($xNbB_1$, $yNbB_1$) inside the neighbouring luma coding block is set equal to (xPb + nPbW − 1, yPb − 1).

. . .

2. $availableA_1$ is equal to TRUE and the prediction units covering the luma locations ($xNbA_1$, $yNbA_1$) and ($xNbB_1$, $yNbB_1$) have the same motion vectors and the same reference indices.

. . .

For the derivation of $availableFlagB_0$, $refIdxLXB_0$, $predFlagLXB_0$ and $mvLXB_0$ the following applies:

– The luma location ($xNbB_0$, $yNbB_0$) inside the neighbouring luma coding block is set equal to (xPb + nPbW, yPb − 1).

. . .

4. $availableB_1$ is equal to TRUE and the prediction units covering the luma locations ($xNbB_1$, $yNbB_1$) and ($xNbB_0$, $yNbB_0$) have the same motion vectors and the same reference indices.

. . .

For the derivation of $availableFlagA_0$, $refIdxLXA_0$, $predFlagLXA_0$ and $mvLXA_0$ the following applies:

– The luma location ($xNbA_0$, $yNbA_0$) inside the neighbouring luma coding block is set equal to (xPb − 1, yPb + nPbH).

. . .

6. $availableA_1$ is equal to TRUE and the prediction units covering the luma locations ($xNbA_1$, $yNbA_1$) and ($xNbA_0$, $yNbA_0$) have the same motion vectors and the same reference indices.

. . .

For the derivation of $availableFlagB_2$, $refIdxLXB_2$, $predFlagLXB_2$ and $mvLXB_2$ the following applies:

– The luma location ($xNbB_2$, $yNbB_2$) inside the neighbouring luma coding block is set equal to (xPb − 1, yPb − 1).

. . .

8. $availableA_1$ is equal to TRUE and prediction units covering the luma locations ($xNbA_1$, $yNbA_1$) and ($xNbB_2$, $yNbB_2$) have the same motion vectors and the same reference indices.

9. $availableB_1$ is equal to TRUE and the prediction units covering the luma locations ($xNbB_1$, $yNbB_1$) and ($xNbB_2$, $yNbB_2$) have the same motion vectors and the same reference indices.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 150-152.

8.5.3.2.7 Derivation process for motion vector predictor candidates

. . .

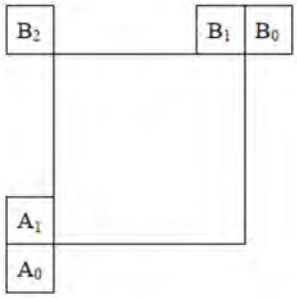


Figure 8-3 – Spatial motion vector neighbours (informative)

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 156

430. When decoding Amazon Prime Video contents, on information and belief, Amazon performs a method of decoding that performs comparing motion information of the first spatial motion vector prediction candidate with motion information of another spatial motion vector prediction candidate of the set of spatial motion vector prediction candidates without making a comparison of each possible candidate pair from the set of spatial motion vector prediction candidates, wherein comparing comprises performing an equivalence check or comparing a

difference in motion information to a threshold or other similarity metric, corresponding to the decoding process specified by the H.265 standard.

8.5.3.2.3 Derivation process for spatial merging candidates

. . .

For the derivation of $availableFlagB_1$, $refIdxLXB_1$, $predFlagLXB_1$ and $mvLXB_1$ the following applies:

– The luma location ($xNbB_1$, $yNbB_1$) inside the neighbouring luma coding block is set equal to (xPb + nPbW − 1, yPb − 1).

. . .

– The variables $availableFlagB_1$, $refIdxLXB_1$, $predFlagLXB_1$ and $mvLXB_1$ are derived as follows:

– If one or more of the following conditions are true, $availableFlagB_1$ is set equal to 0, both components of $mvLXB_1$ are set equal to 0, $refIdxLXB_1$ is set equal to −1 and $predFlagLXB_1$ is set equal to 0, with X being 0 or 1:

. . .

1. $availableB_1$ is equal to FALSE.
2. $availableA_1$ is equal to TRUE and the prediction units covering the luma locations ($xNbA_1$, $yNbA_1$) and ($xNbB_1$, $yNbB_1$) have the same motion vectors and the same reference indices.

– Otherwise, $availableFlagB_1$ is set equal to 1 and the following assignments are made:

$mvLXB_1$ = MvLX[$xNbB_1$][$yNbB_1$] (8-131)

$refIdxLXB_1$ = RefIdxLX[$xNbB_1$][$yNbB_1$] (8-132)

$predFlagLXB_1$ = PredFlagLX[$xNbB_1$][$yNbB_1$] (8-133)

For the derivation of $availableFlagB_0$, $refIdxLXB_0$, $predFlagLXB_0$ and $mvLXB_0$ the following applies:

– The luma location ($xNbB_0$, $yNbB_0$) inside the neighbouring luma coding block is set equal to (xPb + nPbW, yPb − 1).

. . .

– The variables $availableFlagB_0$, $refIdxLXB_0$, $predFlagLXB_0$ and $mvLXB_0$ are derived as follows:

– If one or more of the following conditions are true, $availableFlagB_0$ is set equal to 0, both components of $mvLXB_0$ are set equal to 0, $refIdxLXB_0$ is set equal to −1 and $predFlagLXB_0$ is set equal to 0, with X being 0 or 1:

. . .

3. $availableB_0$ is equal to FALSE.
4. $availableB_1$ is equal to TRUE and the prediction units covering the luma locations ($xNbB_1$, $yNbB_1$) and ($xNbB_0$, $yNbB_0$) have the same motion vectors and the same reference indices.

– Otherwise, $availableFlagB_0$ is set equal to 1 and the following assignments are made:

$mvLXB_0$ = MvLX[$xNbB_0$][$yNbB_0$] (8-134)

$refIdxLXB_0$ = RefIdxLX[$xNbB_0$][$yNbB_0$] (8-135)

$predFlagLXB_0$ = PredFlagLX[$xNbB_0$][$yNbB_0$] (8-136)

For the derivation of $availableFlagA_0$, $refIdxLXA_0$, $predFlagLXA_0$ and $mvLXA_0$ the following applies:

– The luma location ($xNbA_0$, $yNbA_0$) inside the neighbouring luma coding block is set equal to (xPb − 1, yPb + nPbH).

. . .

– The variables $availableFlagA_0$, $refIdxLXA_0$, $predFlagLXA_0$ and $mvLXA_0$ are derived as follows:

– If one or more of the following conditions are true, $availableFlagA_0$ is set equal to 0, both components of $mvLXA_0$ are set equal to 0, $refIdxLXA_0$ is set equal to −1 and $predFlagLXA_0$ is set equal to 0, with X being 0 or 1:

5. $availableA_0$ is equal to FALSE.

6. $availableA_1$ is equal to TRUE and the prediction units covering the luma locations ($xNbA_1$, $yNbA_1$) and ($xNbA_0$, $yNbA_0$) have the same motion vectors and the same reference indices.

– Otherwise, $availableFlagA_0$ is set equal to 1 and the following assignments are made:

$mvLXA_0$ = MvLX[$xNbA_0$][$yNbA_0$] (8-137)

$refIdxLXA_0$ = RefIdxLX[$xNbA_0$][$yNbA_0$] (8-138)

$predFlagLXA_0$ = PredFlagLX[$xNbA_0$][$yNbA_0$] (8-139)

For the derivation of $availableFlagB_2$, $refIdxLXB_2$, $predFlagLXB_2$ and $mvLXB_2$ the following applies:

– The luma location ($xNbB_2$, $yNbB_2$) inside the neighbouring luma coding block is set equal to (xPb − 1, yPb − 1).

. . .

– The variables $availableFlagB_2$, $refIdxLXB_2$, $predFlagLXB_2$ and $mvLXB_2$ are derived as follows:

– If one or more of the following conditions are true, $availableFlagB_2$ is set equal to 0, both components of $mvLXB_2$ are set equal to 0, $refIdxLXB_2$ is set equal to −1 and $predFlagLXB_2$ is set equal to 0, with X being 0 or 1:

7. $availableB_2$ is equal to FALSE.

8. $availableA_1$ is equal to TRUE and prediction units covering the luma locations ($xNbA_1$, $yNbA_1$) and ($xNbB_2$, $yNbB_2$) have the same motion vectors and the same reference indices.

9. $availableB_1$ is equal to TRUE and the prediction units covering the luma locations ($xNbB_1$, $yNbB_1$) and ($xNbB_2$, $yNbB_2$) have the same motion vectors and the same reference indices.

10. $availableFlagA_0$ + $availableFlagA_1$ + $availableFlagB_0$ + $availableFlagB_1$ is equal to 4.

– Otherwise, $availableFlagB_2$ is set equal to 1 and the following assignments are made:

$mvLXB_2$ = MvLX[$xNbB_2$][$yNbB_2$] (8-140)

$refIdxLXB_2$ = RefIdxLX[$xNbB_2$][$yNbB_2$] (8-141)

$predFlagLXB_2$ = PredFlagLX[$xNbB_2$][$yNbB_2$] (8-142)

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 150-152.

8.5.3.2.7 Derivation process for motion vector predictor candidates

. . .

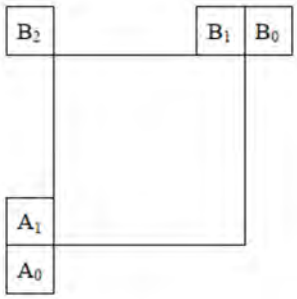


Figure 8-3 – Spatial motion vector neighbours (informative)

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 156.

431. When decoding Amazon Prime Video contents, on information and belief, Amazon performs a method of decoding that performs if at least one of the comparisons indicates that the motion vector information of the spatial motion vector prediction candidates correspond with each other, excluding the first spatial motion vector prediction candidate from the merge list, corresponding to the decoding process specified by the H.265 standard.

8.5.3.2.3 Derivation process for spatial merging candidates

. . .

For the derivation of $availableFlagB_1$, $refIdxLXB_1$, $predFlagLXB_1$ and $mvLXB_1$ the following applies:

– The luma location ($xNbB_1$, $yNbB_1$) inside the neighbouring luma coding block is set equal to (xPb + nPbW − 1, yPb − 1).

. . .

– The variables $availableFlagB_1$, $refIdxLXB_1$, $predFlagLXB_1$ and $mvLXB_1$ are derived as follows:

– If one or more of the following conditions are true, $availableFlagB_1$ is set equal to 0, both components of $mvLXB_1$ are set equal to 0, $refIdxLXB_1$ is set equal to −1 and $predFlagLXB_1$ is set equal to 0, with X being 0 or 1:

. . .

5. $availableB_1$ is equal to FALSE.
6. $availableA_1$ is equal to TRUE and the prediction units covering the luma locations ($xNbA_1$, $yNbA_1$) and ($xNbB_1$, $yNbB_1$) have the same motion vectors and the same reference indices.

– Otherwise, $availableFlagB_1$ is set equal to 1 and the following assignments are made:

$mvLXB_1 = MvLX[\ xNbB_1\ ][\ yNbB_1\ ]$ (8-131)

$refIdxLXB_1 = RefIdxLX[\ xNbB_1\ ][\ yNbB_1\ ]$ (8-132)

$predFlagLXB_1 = PredFlagLX[\ xNbB_1\ ][\ yNbB_1\ ]$ (8-133)

For the derivation of $availableFlagB_0$, $refIdxLXB_0$, $predFlagLXB_0$ and $mvLXB_0$ the following applies:

– The luma location ($xNbB_0$, $yNbB_0$) inside the neighbouring luma coding block is set equal to (xPb + nPbW, yPb − 1).

. . .

– The variables $availableFlagB_0$, $refIdxLXB_0$, $predFlagLXB_0$ and $mvLXB_0$ are derived as follows:

– If one or more of the following conditions are true, $availableFlagB_0$ is set equal to 0, both components of $mvLXB_0$ are set equal to 0, $refIdxLXB_0$ is set equal to −1 and $predFlagLXB_0$ is set equal to 0, with X being 0 or 1:

. . .

7. $availableB_0$ is equal to FALSE.

8. $availableB_1$ is equal to TRUE and the prediction units covering the luma locations ($xNbB_1$, $yNbB_1$) and ($xNbB_0$, $yNbB_0$) have the same motion vectors and the same reference indices.

– Otherwise, $availableFlagB_0$ is set equal to 1 and the following assignments are made:

$mvLXB_0$ = MvLX[$xNbB_0$][$yNbB_0$] (8-134)

$refIdxLXB_0$ = RefIdxLX[$xNbB_0$][$yNbB_0$] (8-135)

$predFlagLXB_0$ = PredFlagLX[$xNbB_0$][$yNbB_0$] (8-136)

For the derivation of $availableFlagA_0$, $refIdxLXA_0$, $predFlagLXA_0$ and $mvLXA_0$ the following applies:

– The luma location ($xNbA_0$, $yNbA_0$) inside the neighbouring luma coding block is set equal to (xPb − 1, yPb + nPbH).

. . .

– The variables $availableFlagA_0$, $refIdxLXA_0$, $predFlagLXA_0$ and $mvLXA_0$ are derived as follows:

– If one or more of the following conditions are true, $availableFlagA_0$ is set equal to 0, both components of $mvLXA_0$ are set equal to 0, $refIdxLXA_0$ is set equal to −1 and $predFlagLXA_0$ is set equal to 0, with X being 0 or 1:

5. $availableA_0$ is equal to FALSE.

6. $availableA_1$ is equal to TRUE and the prediction units covering the luma locations ($xNbA_1$, $yNbA_1$) and ($xNbA_0$, $yNbA_0$) have the same motion vectors and the same reference indices.

– Otherwise, $availableFlagA_0$ is set equal to 1 and the following assignments are made:

$mvLXA_0$ = MvLX[$xNbA_0$][$yNbA_0$] (8-137)

$refIdxLXA_0$ = RefIdxLX[$xNbA_0$][$yNbA_0$] (8-138)

$predFlagLXA_0$ = PredFlagLX[$xNbA_0$][$yNbA_0$] (8-139)

For the derivation of $availableFlagB_2$, $refIdxLXB_2$, $predFlagLXB_2$ and $mvLXB_2$ the following applies:

– The luma location ($xNbB_2$, $yNbB_2$) inside the neighbouring luma coding block is set equal to (xPb − 1, yPb − 1).

. . .

– The variables $availableFlagB_2$, $refIdxLXB_2$, $predFlagLXB_2$ and $mvLXB_2$ are derived as follows:

– If one or more of the following conditions are true, $availableFlagB_2$ is set equal to 0, both components of $mvLXB_2$ are set equal to 0, $refIdxLXB_2$ is set equal to −1 and $predFlagLXB_2$ is set equal to 0, with X being 0 or 1:

7. $availableB_2$ is equal to FALSE.

8. $availableA_1$ is equal to TRUE and prediction units covering the luma locations ($xNbA_1$, $yNbA_1$) and ($xNbB_2$, $yNbB_2$) have the same motion vectors and the same reference indices.

9. $availableB_1$ is equal to TRUE and the prediction units covering the luma locations ($xNbB_1$, $yNbB_1$) and ($xNbB_2$, $yNbB_2$) have the same motion vectors and the same reference indices.

10. $availableFlagA_0 + availableFlagA_1 + availableFlagB_0 + availableFlagB_1$ is equal to 4.

– Otherwise, $availableFlagB_2$ is set equal to 1 and the following assignments are made:

$mvLXB_2 = MvLX[\ xNbB_2\ ][\ yNbB_2\ ]$ (8-140)

$refIdxLXB_2 = RefIdxLX[\ xNbB_2\ ][\ yNbB_2\ ]$ (8-141)

$predFlagLXB_2 = PredFlagLX[\ xNbB_2\ ][\ yNbB_2\ ]$ (8-142)

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 150-152.

8.5.3.2.7 Derivation process for motion vector predictor candidates

. . .

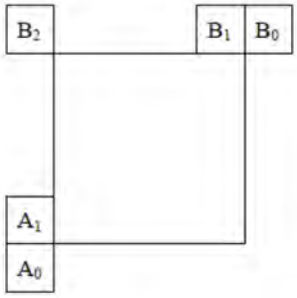


Figure 8-3 – Spatial motion vector neighbours (informative)

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 156.

432. When decoding Amazon Prime Video contents, on information and belief, Amazon performs a method of decoding that performs selecting a spatial motion vector prediction candidate from the merge list for use in decoding the encoded block of pixels, wherein the spatial motion vector prediction candidate is selected from the merge list using the information that was received identifying a respective spatial motion vector prediction candidate, corresponding to the decoding process specified by the H.265 standard.

3 Definitions

For the purposes of this Recommendation | International Standard, the following definitions apply.

. . .

3.153 syntax element: An element of data represented in the *bitstream*.

3.154 syntax structure: Zero or more *syntax elements* present together in the *bitstream* in a specified order.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 5, 7-12.

5.10 Variables, syntax elements and tables

Syntax elements in the bitstream are represented in **bold** type. Each syntax element is described by its name (all lower case letters with underscore characters), and one descriptor for its method of coded representation. The decoding process behaves according to the value of the syntax element and to the values of previously decoded syntax elements. When a value of a syntax element is used in the syntax tables or the text, it appears in regular (i.e., not bold) type.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 18.

7 Syntax and semantics

7.1 Method of specifying syntax in tabular form

The syntax tables specify a superset of the syntax of all allowed bitstreams **. . .**

. . .

. . . When **syntax_element** appears, it specifies that a syntax element is parsed from the bitstream **. . .**

. . .

7.2 Specification of syntax functions and descriptors

. . .

The following descriptors specify the parsing process of each syntax element:

– ae(v): context-adaptive arithmetic entropy-coded syntax element. The parsing process for this descriptor is specified in clause 9.3.

. . .

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 30-31.

7.3.8.6 Prediction unit syntax

prediction_unit(x0, y0, nPbW, nPbH) {	Descriptor
if(cu_skip_flag[x0][y0]) {	
if(MaxNumMergeCand > 1)	
merge_idx[x0][y0]	ae(v)
} else { /* MODE_INTER */	
merge_flag[x0][y0]	ae(v)
if(merge_flag[x0][y0]) {	
if(MaxNumMergeCand > 1)	
merge_idx[x0][y0]	ae(v)
} else {	
if(slice_type == B)	
inter_pred_idc[x0][y0]	ae(v)
if(inter_pred_idc[x0][y0] != PRED_L1) {	
if(num_ref_idx_l0_active_minus1 > 0)	
ref_idx_l0[x0][y0]	ae(v)
mvd_coding(x0, y0, 0)	
mvp_l0_flag[x0][y0]	ae(v)
}	
if(inter_pred_idc[x0][y0] != PRED_L0) {	
if(num_ref_idx_l1_active_minus1 > 0)	
ref_idx_l1[x0][y0]	ae(v)
if(mvd_l1_zero_flag && inter_pred_idc[x0][y0] == PRED_BI) {	
MvdL1[x0][y0][0] = 0	
MvdL1[x0][y0][1] = 0	
} else	
mvd_coding(x0, y0, 1)	
mvp_l1_flag[x0][y0]	ae(v)
}	
}	
}	
}	

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 55.

7.4.9.6 Prediction unit semantics

. . .

merge_idx[x0][y0] specifies the merging candidate index of the merging candidate list where x0, y0 specify the location (x0, y0) of the top-left luma sample of the considered prediction block relative to the top-left luma sample of the picture.

When merge_idx[x0][y0] is not present, it is inferred to be equal to 0.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 107.

8.5.3.2.2 Derivation process for luma motion vectors for merge mode

. . .

Outputs of this process are as follows, with X being 0 or 1:

. . .

- the reference indices $refIdxLXA_0$, $refIdxLXA_1$, $refIdxLXB_0$, $refIdxLXB_1$ and $refIdxLXB_2$ of the neighbouring prediction units,

. . .

- the motion vectors $mvLXA_0$, $mvLXA_1$, $mvLXB_0$, $mvLXB_1$ and $mvLXB_2$ of the neighbouring prediction units.

. . .

9. The following assignments are made with N being the candidate at position merge_idx[xOrigP][yOrigP] in the merging candidate list mergeCandList (N = mergeCandList[merge_idx[xOrigP][yOrigP]]) and X being replaced by 0 or 1:

refIdxLX = refIdxLXN (8-120)

predFlagLX = predFlagLXN (8-121)

. . .

mvLX[0] = mvLXN[0] (8-122)

mvLX[1] = mvLXN[1] (8-123)

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 147-149.

9 Parsing process

9.1 General

Inputs to this process are bits …

Outputs of this process are syntax element values.

This process is invoked when the descriptor of a syntax element in the syntax tables in clause 7.3 is equal to ue(v), se(v) (see clause 9.2), or ae(v) (see clause 9.3).

9.3 CABAC parsing process for slice segment data

9.3.1 General

This process is invoked when parsing syntax elements with descriptor ae(v) in clauses 7.3.8.1 through 7.3.8.11.

Inputs to this process are a request for a value of a syntax element and values of prior parsed syntax elements.

Output of this process is the value of the syntax element.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 201.

433. For another example, on information and belief, Amazon performs a method of decoding video in a manner that is covered by claim 9 of the ’833 Patent in the same or similar manner as described above, when Amazon.com videos are decoded by an H.265-compliant decoder, and as performed, for example, during internal testing of video quality and transcoding of videos on Amazon.com.

434. On information and belief, Amazon performs internal testing of video quality on Amazon.com and transcoding of video on Amazon.com in a manner substantially similar to that

which is performed on Amazon Prime Video. *See, e.g.*, https://www.amazon.science/blog/how-prime-video-uses-machine-learning-to-ensure-video-quality.

N. Amazon Makes, Imports, Uses, Sells, and/or Offers for Sale Products and Services that Infringe the ’267 Patent

435. The Accused Products infringe one or more claims of the ’267 Patent, including, for example, claim 19.

436. As just one example of infringement, on information and belief, Amazon performs a method of decoding a block of pixels in a video signal in a manner that is covered by claim 19 of the ’267 Patent for Amazon Prime Video contents, when Amazon Prime Video contents are decoded by an H.265-compliant decoder, and as performed, for example, during internal testing of video quality and transcoding. *See, e.g.*, https://www.amazon.science/blog/how-prime-video-uses-machine-learning-to-ensure-video-quality.

Screenshots of Amazon Prime Video indicating “Codec: hevc.”

437. When decoding Amazon Prime Video contents, on information and belief, Amazon performs a method of decoding that performs determining, for a current block, a first reference block based on a first motion vector and a second reference block based on a second motion

vector, wherein the pixels of the current block, the first reference block, and the second reference block have values with a first precision, corresponding to the decoding process specified by the H.265 standard.

3 Definitions

For the purposes of this Recommendation | International Standard, the following definitions apply:

3.11 **bi-predictive (B) slice**: A *slice* that is decoded using *intra prediction* or using *inter prediction* with at most two *motion vectors* and *reference indices* to *predict* the sample values of each *block*.

3.12 **bitstream**: A sequence of bits, in the form of a *NAL unit stream* or a *byte stream*, that forms the representation of *coded pictures* and associated data forming one or more coded video sequences *(CVSs)*.

3.41 **decoder**: An embodiment of a *decoding process*.

3.44 **decoding process**: The process specified in this Specification that reads a *bitstream* and derives *decoded pictures* from it.

3.63 **inter coding**: Coding of a *coding block*, *slice*, or *picture* that uses *inter prediction*.

3.64 **inter prediction**: A *prediction* derived in a manner that is dependent on data elements (e.g., sample values or motion vectors) of one or more *reference pictures*.

NOTE – A prediction from a reference picture that is the current picture itself is also inter prediction.

3.65 **intra coding**: Coding of a *coding block*, *slice*, or *picture* that uses *intra prediction*.

3.66 **intra prediction**: A *prediction* derived from only data elements (e.g., sample values) of the same decoded *slice* without referring to a *reference picture*.

3.69 **intra (I) slice**: A *slice* that is decoded using *intra prediction* only.

3.76 **list 0 (list 1) motion vector**: A *motion vector* associated with a *reference index* pointing into *reference picture list 0* (*list 1*).

3.77 **list 0 (list 1) prediction**: *Inter prediction* of the content of a *slice* using a *reference index* pointing into *reference picture list 0* (*list 1*).

3.82 **motion vector**: A two-dimensional vector used for *inter prediction* that provides an offset from the coordinates in the *decoded picture* to the coordinates in a *reference picture*.

3.100 **prediction**: An embodiment of the *prediction process*.

3.101 **prediction block**: A rectangular MxN *block* of samples on which the same *prediction* is applied.

3.102 **prediction process**: The use of a *predictor* to provide an estimate of the data element (e.g., sample value or motion vector) currently being decoded.

3.103 **prediction unit**: A *prediction block* of *luma* samples, two corresponding *prediction blocks* of *chroma* samples of a *picture* that has three sample arrays, or a *prediction block* of samples of a monochrome *picture* or a *picture* that is coded using three separate colour planes and *syntax structures* used to predict the *prediction block* samples.

3.104 **predictive (P) slice**: A *slice* that is decoded using *intra prediction* or using *inter prediction* with at most one *motion vector* and *reference index* to *predict* the sample values of each *block*.

3.121 **reference index**: An index into a *reference picture list*.

3.122 **reference picture**: A *picture* that is a *short-term reference picture* or a *long-term reference picture*.

NOTE – A reference picture contains samples that may be used for inter prediction in the decoding process of subsequent pictures in decoding order.

3.123 **reference picture list**: A list of *reference pictures* that is used for *inter prediction* of a *P* or *B slice*.

NOTE – For the decoding process of a P slice, there is one reference picture list – reference picture list 0. For the decoding process of a B slice, there are two reference picture lists – reference picture list 0 and reference picture list 1.

3.124 **reference picture list 0**: The *reference picture list* used for *inter prediction* of a *P* or the first *reference picture list* used for *inter prediction* of a *B slice*.

3.125 **reference picture list 1**: The second *reference picture list* used for *inter prediction* of a *B slice*.

3.136 **slice**: An integer number of *coding tree units* contained in one *independent slice segment* and all subsequent *dependent slice segments* (if any) that precede the next *independent slice segment* (if any) within the same *access unit*.

3.153 **syntax element**: An element of data represented in the *bitstream*.

3.154 **syntax structure**: Zero or more *syntax elements* present together in the *bitstream* in a specified order.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 4-12.

5.10 Variables, syntax elements and tables

Syntax elements in the bitstream are represented in **bold** type. Each syntax element is described by its name (all lower case letters with underscore characters), and one descriptor for its method of coded representation. The decoding process behaves according to the value of the syntax element and to the values of previously decoded syntax elements. When a value of a syntax element is used in the syntax tables or the text, it appears in regular (i.e., not bold) type.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 18.

6.3 Partitioning of pictures, slices, slice segments, tiles, coding tree units and coding tree blocks

6.3.1 Partitioning of pictures into slices, slice segments and tiles

This clause specifies how a picture is partitioned into slices, slice segments and tiles. Pictures are divided into slices and tiles. A slice is a sequence of one or more slice segments starting with an independent slice segment and containing all subsequent dependent slice segments (if any) that precede the next independent slice segment (if any) within the same picture. A slice segment is a sequence of coding tree units. Likewise, a tile is a sequence of coding tree units.

For example, a picture may be divided into two slices as shown in Figure 6-4. In this example, the first slice is composed of an independent slice segment containing 4 coding tree units, a dependent slice segment containing 32 coding tree units and another dependent slice segment containing 24 coding tree units; and the second slice consists of a single independent slice segment containing the remaining 39 coding tree units of the picture.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 23.

7 Syntax and semantics

7.1 Method of specifying syntax in tabular form

The syntax tables specify a superset of the syntax of all allowed bitstreams. Additional constraints on the syntax may be specified, either directly or indirectly, in other clauses.

NOTE An actual decoder should implement some means for identifying entry points into the bitstream and some means to identify and handle non-conforming bitstreams. The methods for identifying and handling errors and other such situations are not specified in this Specification.

The following table lists examples of the syntax specification format. When **syntax_element** appears, it specifies that a syntax element is parsed from the bitstream and the bitstream pointer is advanced to the next position beyond the syntax element in the bitstream parsing process.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 30.

7.3.6 Slice segment header syntax

7.3.6.1 General slice segment header syntax

slice_segment_header( ) {	Descriptor
first_slice_segment_in_pic_flag	u(1)
if(nal_unit_type >= BLA_W_LP && nal_unit_type <= RSV_IRAP_VCL23)	
no_output_of_prior_pics_flag	u(1)
slice_pic_parameter_set_id	ue(v)
if(!first_slice_segment_in_pic_flag) {	
if(dependent_slice_segments_enabled_flag)	
dependent_slice_segment_flag	u(1)
slice_segment_address	u(v)
}	
if(!dependent_slice_segment_flag) {	
for(i = 0; i < num_extra_slice_header_bits; i++)	
slice_reserved_flag[i]	u(1)
slice_type	ue(v)
if(output_flag_present_flag)	

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 44.

7.3.8.5 Coding unit syntax

coding_unit(x0, y0, log2CbSize) {	Descriptor
if(transquant_bypass_enabled_flag)	
cu_transquant_bypass_flag	ae(v)
if(slice_type != I)	
cu_skip_flag[x0][y0]	ae(v)
nCbS = (1 << log2CbSize)	
if(cu_skip_flag[x0][y0])	
prediction_unit(x0, y0, nCbS, nCbS)	
else {	
if(slice_type != I)	
pred_mode_flag	ae(v)
if(palette_mode_enabled_flag && CuPredMode[x0][y0] == MODE_INTRA && log2CbSize <= MaxTbLog2SizeY)	
palette_mode_flag[x0][y0]	ae(v)
if(palette_mode_flag[x0][y0])	
palette_coding(x0, y0, nCbS)	
else {	
if(CuPredMode[x0][y0] != MODE_INTRA || log2CbSize == MinCbLog2SizeY)	
part_mode	ae(v)
if(CuPredMode[x0][y0] == MODE_INTRA) {	

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 52-53.

7.3.8.6 Prediction unit syntax

prediction_unit(x0, y0, nPbW, nPbH) {	Descriptor
if(cu_skip_flag[x0][y0]) {	
if(MaxNumMergeCand > 1)	
merge_idx[x0][y0]	ae(v)
} else { /* MODE_INTER */	
merge_flag[x0][y0]	ae(v)
if(merge_flag[x0][y0]) {	
if(MaxNumMergeCand > 1)	
merge_idx[x0][y0]	ae(v)
} else {	
if(slice_type == B)	
inter_pred_idc[x0][y0]	ae(v)
if(inter_pred_idc[x0][y0] != PRED_L1) {	
if(num_ref_idx_l0_active_minus1 > 0)	
ref_idx_l0[x0][y0]	ae(v)
mvd_coding(x0, y0, 0)	
mvp_l0_flag[x0][y0]	ae(v)
}	
if(inter_pred_idc[x0][y0] != PRED_L0) {	
if(num_ref_idx_l1_active_minus1 > 0)	
ref_idx_l1[x0][y0]	ae(v)
if(mvd_l1_zero_flag && inter_pred_idc[x0][y0] == PRED_BI) {	
MvdL1[x0][y0][0] = 0	
MvdL1[x0][y0][1] = 0	
} else	
mvd_coding(x0, y0, 1)	
mvp_l1_flag[x0][y0]	ae(v)
}	
}	
}	
}	

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 55.

slice_type specifies the coding type of the slice according to Table 7-7.

Table 7-7 – Name association to slice_type

slice_type	Name of slice_type
0	B (B slice)
1	P (P slice)
2	I (I slice)

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 95.

pred_mode_flag equal to 0 specifies that the current coding unit is coded in inter prediction mode. pred_mode_flag equal to 1 specifies that the current coding unit is coded in intra prediction mode. The variable CuPredMode[x][y] is derived as follows for x = x0..x0 + nCbS − 1 and y = y0..y0 + nCbS − 1:

- If pred_mode_flag is equal to 0, CuPredMode[x][y] is set equal to MODE_INTER.
- Otherwise (pred_mode_flag is equal to 1), CuPredMode[x][y] is set equal to MODE_INTRA.

When pred_mode_flag is not present, the variable CuPredMode[x][y] is derived as follows for x = x0..x0 + nCbS − 1 and y = y0..y0 + nCbS − 1:

- If slice_type is equal to I, CuPredMode[x][y] is inferred to be equal to MODE_INTRA.
- Otherwise (slice_type is equal to P or B), when cu_skip_flag[x0][y0] is equal to 1, CuPredMode[x][y] is inferred to be equal to MODE_SKIP.

part_mode specifies the partitioning mode of the current coding unit. The semantics of part_mode depend on CuPredMode[x0][y0]. The variables PartMode and IntraSplitFlag are derived from the value of part_mode as defined in Table 7-10.

The value of part_mode is restricted as follows:

- If CuPredMode[x0][y0] is equal to MODE_INTRA, part_mode shall be equal to 0 or 1.
- Otherwise (CuPredMode[x0][y0] is equal to MODE_INTER), the following applies:
  - If log2CbSize is greater than MinCbLog2SizeY and amp_enabled_flag is equal to 1, part_mode shall be in the range of 0 to 2, inclusive, or in the range of 4 to 7, inclusive.
  - Otherwise, if log2CbSize is greater than MinCbLog2SizeY and amp_enabled_flag is equal to 0, or log2CbSize is equal to 3, part_mode shall be in the range of 0 to 2, inclusive.
  - Otherwise (log2CbSize is greater than 3 and equal to MinCbLog2SizeY), the value of part_mode shall be in the range of 0 to 3, inclusive.

When part_mode is not present, the variables PartMode and IntraSplitFlag are derived as follows:

- PartMode is set equal to PART_2Nx2N.
- IntraSplitFlag is set equal to 0.

Table 7-10 – Name association to prediction mode and partitioning type

CuPredMode[x0][y0]	part_mode	IntraSplitFlag	PartMode
MODE_INTRA	0	0	PART_2Nx2N
	1	1	PART_NxN
MODE_INTER	0	0	PART_2Nx2N
	1	0	PART_2NxN
	2	0	PART_Nx2N
	3	0	PART_NxN
	4	0	PART_2NxnU
	5	0	PART_2NxnD
	6	0	PART_nLx2N
	7	0	PART_nRx2N

inter_pred_idc[x0][y0] specifies whether list0, list1, or bi-prediction is used for the current prediction unit according to Table 7-11. The array indices x0, y0 specify the location (x0, y0) of the top-left luma sample of the considered prediction block relative to the top-left luma sample of the picture.

Table 7-11 – Name association to inter prediction mode

inter_pred_idc	Name of inter_pred_idc	
	(nPbW + nPbH) != 12	(nPbW + nPbH) == 12
0	PRED_L0	PRED_L0
1	PRED_L1	PRED_L1
2	PRED_BI	na

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 106-108.

6.2 Source, decoded and output picture formats

This clause specifies the relationship between source and decoded pictures that is given via the bitstream.

The video source that is represented by the bitstream is a sequence of pictures in decoding order.

The source and decoded pictures are each comprised of one or more sample arrays:

- Luma (Y) only (monochrome).
- Luma and two chroma (YCbCr or YCgCo).
- Green, Blue and Red (GBR, also known as RGB).
- Arrays representing other unspecified monochrome or tri-stimulus colour samplings (for example, YZX, also known as XYZ).

For convenience of notation and terminology in this Specification, the variables and terms associated with these arrays are referred to as luma (or L or Y) and chroma, where the two chroma arrays are referred to as Cb and Cr; regardless of the actual colour representation method in use. The actual colour representation method in use can be indicated in syntax that is specified in Annex E.

The number of bits necessary for the representation of each of the samples in the luma and chroma arrays in a video sequence is in the range of 8 to 16, inclusive, and the number of bits used in the luma array may differ from the number of bits used in the chroma arrays.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 21.

7.3.2.2.1 General sequence parameter set RBSP syntax

seq_parameter_set_rbsp( ) {	Descriptor
sps_video_parameter_set_id	u(4)
sps_max_sub_layers_minus1	u(3)
sps_temporal_id_nesting_flag	u(1)
profile_tier_level(1, sps_max_sub_layers_minus1)	
sps_seq_parameter_set_id	ue(v)
chroma_format_idc	ue(v)
if(chroma_format_idc == 3)	
separate_colour_plane_flag	u(1)
pic_width_in_luma_samples	ue(v)
pic_height_in_luma_samples	ue(v)
conformance_window_flag	u(1)
if(conformance_window_flag) {	
conf_win_left_offset	ue(v)
conf_win_right_offset	ue(v)
conf_win_top_offset	ue(v)
conf_win_bottom_offset	ue(v)
}	
bit_depth_luma_minus8	ue(v)
bit_depth_chroma_minus8	ue(v)
log2_max_pic_order_cnt_lsb_minus4	ue(v)

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 34.

bit_depth_luma_minus8 specifies the bit depth of the samples of the luma array $BitDepth_Y$ and the value of the luma quantization parameter range offset $QpBdOffset_Y$ as follows:

$$BitDepth_Y = 8 + bit_depth_luma_minus8 \quad (7\text{-}4)$$

$$QpBdOffset_Y = 6 * bit_depth_luma_minus8 \quad (7\text{-}5)$$

bit_depth_luma_minus8 shall be in the range of 0 to 8, inclusive.

bit_depth_chroma_minus8 specifies the bit depth of the samples of the chroma arrays $BitDepth_C$ and the value of the chroma quantization parameter range offset $QpBdOffset_C$ as follows:

$$BitDepth_C = 8 + bit_depth_chroma_minus8 \quad (7\text{-}6)$$

$$QpBdOffset_C = 6 * bit_depth_chroma_minus8 \quad (7\text{-}7)$$

bit_depth_chroma_minus8 shall be in the range of 0 to 8, inclusive.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 76.

8.5.3 Decoding process for prediction units in inter prediction mode

8.5.3.1 General

Inputs to this process are:

- a luma location (xCb, yCb) specifying the top-left sample of the current luma coding block relative to the top-left luma sample of the current picture,
- a luma location (xBl, yBl) specifying the top-left sample of the current luma prediction block relative to the top-left sample of the current luma coding block,
- a variable nCbS specifying the size of the current luma coding block,
- a variable nPbW specifying the width of the current luma prediction block,
- a variable nPbH specifying the height of the current luma prediction block,
- a variable partIdx specifying the index of the current prediction unit within the current coding unit.

Outputs of this process are:

- an $(nCbS_L)x(nCbS_L)$ array $predSamples_L$ of luma prediction samples, where $nCbS_L$ is derived as specified below,
- when ChromaArrayType is not equal to 0, an $(nCbSw_C)x(nCbSh_C)$ array $predSamples_{Cb}$ of chroma prediction samples for the component Cb, where $nCbSw_C$ and $nCbSh_C$ are derived as specified below,

  when ChromaArrayType is not equal to 0, an $(nCbSw_C)x(nCbSh_C)$ array $predSamples_{Cr}$ of chroma prediction samples for the component Cr, where $nCbSw_C$ and $nCbSh_C$ are derived as specified below.

The decoding process for prediction units in inter prediction mode consists of the following ordered steps:

1. The derivation process for motion vector components and reference indices as specified in clause 8.5.3.2 is invoked with the luma coding block location (xCb, yCb), the luma prediction block location (xBl, yBl), the luma coding block size block nCbS, the luma prediction block width nPbW, the luma prediction block height nPbH and the prediction unit index partIdx as inputs, and the luma motion vectors mvL0 and mvL1, when ChromaArrayType is not equal to 0, the chroma motion vectors mvCL0 and mvCL1, the reference indices refIdxL0 and refIdxL1 and the prediction list utilization flags predFlagL0 and predFlagL1 as outputs.
2. The decoding process for inter sample prediction as specified in clause 8.5.3.3 is invoked with the luma coding block location (xCb, yCb), the luma prediction block location (xBl, yBl), the luma coding block size block nCbS, the luma prediction block width nPbW, the luma prediction block height nPbH, the luma motion vectors mvL0 and mvL1, when ChromaArrayType is not equal to 0, the chroma motion vectors mvCL0 and mvCL1, the reference indices refIdxL0 and refIdxL1, and the prediction list utilization flags predFlagL0 and predFlagL1 as inputs, and the inter prediction samples (predSamples) that are an $(nCbS_L)x(nCbS_L)$ array $predSamples_L$ of prediction luma samples and, when ChromaArrayType is not equal to 0, two $(nCbSw_C)x(nCbSh_C)$ arrays $predSamples_{Cb}$ and $predSamples_{Cr}$ of prediction chroma samples, one for each of the chroma components Cb and Cr, as outputs.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 144-45.

8.5.3.2 Derivation process for motion vector components and reference indices

8.5.3.2.1 General

Inputs to this process are:

- a luma location (xCb, yCb) of the top-left sample of the current luma coding block relative to the top-left luma sample of the current picture,
- a luma location (xBl, yBl) of the top-left sample of the current luma prediction block relative to the top-left sample of the current luma coding block,
- a variable nCbS specifying the size of the current luma coding block,
- two variables nPbW and nPbH specifying the width and the height of the luma prediction block,
- a variable partIdx specifying the index of the current prediction unit within the current coding unit.

Outputs of this process are:

- the luma motion vectors mvL0 and mvL1,
- when ChromaArrayType is not equal to 0, the chroma motion vectors mvCL0 and mvCL1,
- the reference indices refIdxL0 and refIdxL1,
- the prediction list utilization flags predFlagL0 and predFlagL1.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 144-45.

For the derivation of the variables mvL0 and mvL1, refIdxL0 and refIdxL1, as well as predFlagL0 and predFlagL1, the following applies:

- If merge_flag[xPb][yPb] is equal to 1, the derivation process for luma motion vectors for merge mode as specified in clause 8.5.3.2.2 is invoked with the luma location (xCb, yCb), the luma location (xPb, yPb), the variables nCbS, nPbW, nPbH and the partition index partIdx as inputs, and the output being the luma motion vectors mvL0, mvL1, the reference indices refIdxL0, refIdxL1 and the prediction list utilization flags predFlagL0 and predFlagL1.
- Otherwise, for X being replaced by either 0 or 1 in the variables predFlagLX, mvLX and refIdxLX, in PRED_LX, and in the syntax elements ref_idx_lX and MvdLX, the following applies:
  1. The variables refIdxLX and predFlagLX are derived as follows:
     - If inter_pred_idc[xPb][yPb] is equal to PRED_LX or PRED_BI,

       refIdxLX = ref_idx_lX[xPb][yPb] (8-88)

       predFlagLX = 1 (8-89)

     - Otherwise, the variables refIdxLX and predFlagLX are specified by:

       refIdxLX = −1 (8-90)

       predFlagLX = 0 (8-91)
  2. The variable mvdLX is derived as follows:

     mvdLX[0] = MvdLX[xPb][yPb][0] (8-92)

     mvdLX[1] = MvdLX[xPb][yPb][1] (8-93)

4. When predFlagLX is equal to 1 and the picture with index refIdx from reference picture list LX of the slice is not the current picture, and use_integer_mv_flag is equal to 0, the luma motion vector mvLX is derived as follows:

$$uLX[\,0\,] = (\,mvpLX[\,0\,] + mvdLX[\,0\,] + 2^{16}\,)\ \%\ 2^{16} \quad (8\text{-}94)$$

$$mvLX[\,0\,] = (\,uLX[\,0\,] >= 2^{15}\,)\ ?\ (\,uLX[\,0\,] - 2^{16}\,) : uLX[\,0\,] \quad (8\text{-}95)$$

$$uLX[\,1\,] = (\,mvpLX[\,1\,] + mvdLX[\,1\,] + 2^{16}\,)\ \%\ 2^{16} \quad (8\text{-}96)$$

$$mvLX[\,1\,] = (\,uLX[\,1\,] >= 2^{15}\,)\ ?\ (\,uLX[\,1\,] - 2^{16}\,) : uLX[\,1\,] \quad (8\text{-}97)$$

NOTE 1 – The resulting values of mvLX[0] and mvLX[1] as specified above will always be in the range of -2^{15} to $2^{15} - 1$, inclusive.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 145-46.

8.5.3.3 Decoding process for inter prediction samples

8.5.3.3.1 General

Inputs to this process are:

- a luma location (xCb, yCb) specifying the top-left sample of the current luma coding block relative to the top-left luma sample of the current picture,
- a luma location (xBl, yBl) specifying the top-left sample of the current luma prediction block relative to the top-left sample of the current luma coding block,
- a variable nCbS specifying the size of the current luma coding block,
- two variables nPbW and nPbH specifying the width and the height of the luma prediction block,
- the luma motion vectors mvL0 and mvL1,

- when ChromaArrayType is not equal to 0, the chroma motion vectors mvCL0 and mvCL1,
- the reference indices refIdxL0 and refIdxL1,
- the prediction list utilization flags, predFlagL0, and predFlagL1.

Outputs of this process are:

- an $(nCbS_L)x(nCbS_L)$ array $predSamples_L$ of luma prediction samples, where $nCbS_L$ is derived as specified below,
- when ChromaArrayType is not equal to 0, an $(nCbSw_C)x(nCbSh_C)$ array $predSamples_{Cb}$ of chroma prediction samples for the component Cb, where $nCbSw_C$ and $nCbSh_C$ are derived as specified below,
- when ChromaArrayType is not equal to 0, an $(nCbSw_C)x(nCbSh_C)$ array $predSamples_{Cr}$ of chroma prediction samples for the component Cr, where $nCbSw_C$ and $nCbSh_C$ are derived as specified below.

The variable $nCbS_L$ is set equal to nCbS. When ChromaArrayType is not equal to 0, the variable $nCbSw_C$ is set equal to nCbS / SubWidthC and the variable $nCbSh_C$ is set equal to nCbS / SubHeightC.

Let $predSamplesL0_L$ and $predSamplesL1_L$ be (nPbW)x(nPbH) arrays of predicted luma sample values and, when ChromaArrayType is not equal to 0, $predSamplesL0_{Cb}$, $predSamplesL1_{Cb}$, $predSamplesL0_{Cr}$ and $predSamplesL1_{Cr}$ be (nPbW / SubWidthC)x(nPbH / SubHeightC) arrays of predicted chroma sample values.

For X being each of 0 and 1, when predFlagLX is equal to 1, the following applies:

- The reference picture consisting of an ordered two-dimensional array $refPicLX_L$ of luma samples and, when ChromaArrayType is not equal to 0, two ordered two-dimensional arrays $refPicLX_{Cb}$ and $refPicLX_{Cr}$ of chroma samples is derived by invoking the process specified in clause 8.5.3.3.2 with refIdxLX as input.
- The array $predSamplesLX_L$ and, when ChromaArrayType is not equal to 0, the arrays $predSamplesLX_{Cb}$ and $predSamplesLX_{Cr}$ are derived by invoking the fractional sample interpolation process specified in clause 8.5.3.3.3 with the luma locations (xCb, yCb) and (xBl, yBl), the luma prediction block width nPbW, the luma prediction block height nPbH, the motion vectors mvLX and, when ChromaArrayType is not equal to 0, mvCLX, and the reference arrays $refPicLX_L$, $refPicLX_{Cb}$, and $refPicLX_{Cr}$ as inputs.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 161-62.

8.5.3.3.3 Fractional sample interpolation process

8.5.3.3.3.1 General

Inputs to this process are:

- a luma location (xCb, yCb) specifying the top-left sample of the current luma coding block relative to the top-left luma sample of the current picture
- a luma location (xBl, yBl) specifying the top-left sample of the current luma prediction block relative to the top-left sample of the current luma coding block
- two variables nPbW and nPbH specifying the width and the height of the luma prediction block
- a luma motion vector mvLX given in quarter-luma-sample units
- when ChromaArrayType is not equal to 0, a chroma motion vector mvCLX given in eighth-chroma-sample units
- the selected reference picture sample array $refPicLX_L$ and, when ChromaArrayType is not equal to 0, the arrays $refPicLX_{Cb}$ and $refPicLX_{Cr}$.

Outputs of this process are:

- an (nPbW)x(nPbH) array $predSamplesLX_L$ of prediction luma sample values
- when ChromaArrayType is not equal to 0, two (nPbW / SubWidthC)x(nPbH / SubHeightC) arrays $predSamplesLX_{Cb}$ and $predSamplesLX_{Cr}$ of prediction chroma sample values.

The location (xPb, yPb) given in full-sample units of the upper-left luma samples of the current prediction block relative to the upper-left luma sample location of the given reference sample arrays is derived as follows:

$$xPb = xCb + xBl \quad (8\text{-}214)$$

$$yPb = yCb + yBl \quad (8\text{-}215)$$

Let ($xInt_L$, $yInt_L$) be a luma location given in full-sample units and ($xFrac_L$, $yFrac_L$) be an offset given in quarter-sample units. These variables are used only inside this clause for specifying fractional-sample locations inside the reference sample arrays $refPicLX_L$, $refPicLX_{Cb}$ and $refPicLX_{Cr}$.

For each luma sample location ($x_L = 0..nPbW - 1$, $y_L = 0..nPbH - 1$) inside the prediction luma sample array $predSamplesLX_L$, the corresponding prediction luma sample value $predSamplesLX_L[\, x_L \,][\, y_L \,]$ is derived as follows:

The variables $xInt_L$, $yInt_L$, $xFrac_L$ and $yFrac_L$ are derived as follows:

$$xInt_L = xPb + (mvLX[\, 0 \,] >> 2) + x_L \quad (8\text{-}216)$$

$$yInt_L = yPb + (mvLX[\, 1 \,] >> 2) + y_L \quad (8\text{-}217)$$

$$xFrac_L = mvLX[\, 0 \,] \,\&\, 3 \quad (8\text{-}218)$$

$$yFrac_L = mvLX[\, 1 \,] \,\&\, 3 \quad (8\text{-}219)$$

- The prediction luma sample value $predSamplesLX_L[\, x_L \,][\, y_L \,]$ is derived by invoking the process specified in clause 8.5.3.3.3.2 with ($xInt_L$, $yInt_L$), ($xFrac_L$, $yFrac_L$) and $refPicLX_L$ as inputs.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 163.

438. When decoding Amazon Prime Video contents, on information and belief, Amazon performs a method of decoding that performs using said first reference block to obtain a first

prediction, said first prediction having a second precision, which is higher than said first precision, corresponding to the decoding process specified by the H.265 standard.

8.5.3.3 Decoding process for inter prediction samples

8.5.3.3.1 General

Inputs to this process are:

- a luma location (xCb, yCb) specifying the top-left sample of the current luma coding block relative to the top-left luma sample of the current picture,
- a luma location (xBl, yBl) specifying the top-left sample of the current luma prediction block relative to the top-left sample of the current luma coding block,
- a variable nCbS specifying the size of the current luma coding block,
- two variables nPbW and nPbH specifying the width and the height of the luma prediction block,
- the luma motion vectors mvL0 and mvL1,
- when ChromaArrayType is not equal to 0, the chroma motion vectors mvCL0 and mvCL1,
- the reference indices refIdxL0 and refIdxL1,
- the prediction list utilization flags, predFlagL0, and predFlagL1.

Outputs of this process are:

- an $(nCbS_L)x(nCbS_L)$ array $predSamples_L$ of luma prediction samples, where $nCbS_L$ is derived as specified below,
- when ChromaArrayType is not equal to 0, an $(nCbSw_C)x(nCbSh_C)$ array $predSamples_{Cb}$ of chroma prediction samples for the component Cb, where $nCbSw_C$ and $nCbSh_C$ are derived as specified below,
- when ChromaArrayType is not equal to 0, an $(nCbSw_C)x(nCbSh_C)$ array $predSamples_{Cr}$ of chroma prediction samples for the component Cr, where $nCbSw_C$ and $nCbSh_C$ are derived as specified below.

The variable $nCbS_L$ is set equal to nCbS. When ChromaArrayType is not equal to 0, the variable $nCbSw_C$ is set equal to nCbS / SubWidthC and the variable $nCbSh_C$ is set equal to nCbS / SubHeightC.

Let $predSamplesL0_L$ and $predSamplesL1_L$ be (nPbW)x(nPbH) arrays of predicted luma sample values and, when ChromaArrayType is not equal to 0, $predSamplesL0_{Cb}$, $predSamplesL1_{Cb}$, $predSamplesL0_{Cr}$ and $predSamplesL1_{Cr}$ be (nPbW / SubWidthC)x(nPbH / SubHeightC) arrays of predicted chroma sample values.

For X being each of 0 and 1, when predFlagLX is equal to 1, the following applies:

- The reference picture consisting of an ordered two-dimensional array $refPicLX_L$ of luma samples and, when ChromaArrayType is not equal to 0, two ordered two-dimensional arrays $refPicLX_{Cb}$ and $refPicLX_{Cr}$ of chroma samples is derived by invoking the process specified in clause 8.5.3.3.2 with refIdxLX as input.
- The array $predSamplesLX_L$ and, when ChromaArrayType is not equal to 0, the arrays $predSamplesLX_{Cb}$ and $predSamplesLX_{Cr}$ are derived by invoking the fractional sample interpolation process specified in clause 8.5.3.3.3 with the luma locations (xCb, yCb) and (xBl, yBl), the luma prediction block width nPbW, the luma prediction block height nPbH, the motion vectors mvLX and, when ChromaArrayType is not equal to 0, mvCLX, and the reference arrays $refPicLX_L$, $refPicLX_{Cb}$, and $refPicLX_{Cr}$ as inputs.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 161-62.

8.5.3.3.3 Fractional sample interpolation process

8.5.3.3.3.1 General

Inputs to this process are:

- a luma location (xCb, yCb) specifying the top-left sample of the current luma coding block relative to the top-left luma sample of the current picture
- a luma location (xBl, yBl) specifying the top-left sample of the current luma prediction block relative to the top-left sample of the current luma coding block
- two variables nPbW and nPbH specifying the width and the height of the luma prediction block
- a luma motion vector mvLX given in quarter-luma-sample units
- when ChromaArrayType is not equal to 0, a chroma motion vector mvCLX given in eighth-chroma-sample units
- the selected reference picture sample array $refPicLX_L$ and, when ChromaArrayType is not equal to 0, the arrays $refPicLX_{Cb}$ and $refPicLX_{Cr}$.

Outputs of this process are:

- an (nPbW)x(nPbH) array $predSamplesLX_L$ of prediction luma sample values

  when ChromaArrayType is not equal to 0, two (nPbW / SubWidthC)x(nPbH / SubHeightC) arrays $predSamplesLX_{Cb}$ and $predSamplesLX_{Cr}$ of prediction chroma sample values.

The location (xPb, yPb) given in full-sample units of the upper-left luma samples of the current prediction block relative to the upper-left luma sample location of the given reference sample arrays is derived as follows:

$$xPb = xCb + xBl \quad (8\text{-}214)$$

$$yPb = yCb + yBl \quad (8\text{-}215)$$

Let ($xInt_L$, $yInt_L$) be a luma location given in full-sample units and ($xFrac_L$, $yFrac_L$) be an offset given in quarter-sample units. These variables are used only inside this clause for specifying fractional-sample locations inside the reference sample arrays $refPicLX_L$, $refPicLX_{Cb}$ and $refPicLX_{Cr}$.

For each luma sample location ($x_L = 0..nPbW - 1$, $y_L = 0..nPbH - 1$) inside the prediction luma sample array $predSamplesLX_L$, the corresponding prediction luma sample value $predSamplesLX_L[\, x_L \,][\, y_L \,]$ is derived as follows:

The variables $xInt_L$, $yInt_L$, $xFrac_L$ and $yFrac_L$ are derived as follows:

$$xInt_L = xPb + (mvLX[\, 0 \,] >> 2) + x_L \quad (8\text{-}216)$$

$$yInt_L = yPb + (mvLX[\, 1 \,] >> 2) + y_L \quad (8\text{-}217)$$

$$xFrac_L = mvLX[\, 0 \,] \,\&\, 3 \quad (8\text{-}218)$$

$$yFrac_L = mvLX[\, 1 \,] \,\&\, 3 \quad (8\text{-}219)$$

- The prediction luma sample value $predSamplesLX_L[\, x_L \,][\, y_L \,]$ is derived by invoking the process specified in clause 8.5.3.3.3.2 with ($xInt_L$, $yInt_L$), ($xFrac_L$, $yFrac_L$) and $refPicLX_L$ as inputs.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 163.

8.5.3.3.3.2 Luma sample interpolation process

Inputs to this process are:

- a luma location in full-sample units ($xInt_L$, $yInt_L$),
- a luma location in fractional-sample units ($xFrac_L$, $yFrac_L$),
- the luma reference sample array $refPicLX_L$.

Output of this process is a predicted luma sample value $predSampleLX_L$

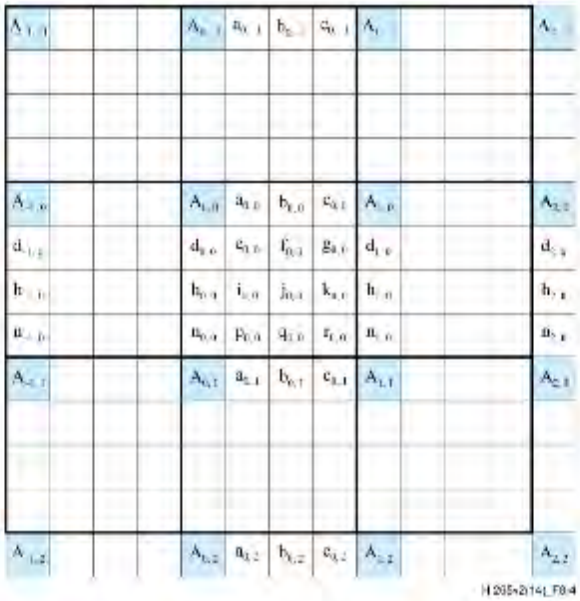


Figure 8-4 – Integer samples (shaded blocks with upper-case letters) and fractional sample positions (un-shaded blocks with lower-case letters) for quarter sample luma interpolation

In Figure 8-4, the positions labelled with upper-case letters $A_{i,j}$ within shaded blocks represent luma samples at full-sample locations inside the given two-dimensional array $refPicLX_L$ of luma samples. These samples may be used for generating the predicted luma sample value $predSampleLX_L$. The locations ($xA_{i,j}$, $yA_{i,j}$) for each of the corresponding luma samples $A_{i,j}$ inside the given array $refPicLX_L$ of luma samples are derived as follows:

$$xA_{i,j} = Clip3(0, pic_width_in_luma_samples - 1, xInt_L + i) \quad (8\text{-}224)$$

$$yA_{i,j} = Clip3(0, pic_height_in_luma_samples - 1, yInt_L + j) \quad (8\text{-}225)$$

The positions labelled with lower-case letters within un-shaded blocks represent luma samples at quarter-luma-sample fractional locations. The luma location offset in fractional-sample units ($xFrac_L$, $yFrac_L$) specifies which of the generated

luma samples at full-sample and fractional-sample locations is assigned to the predicted luma sample value $predSampleLX_L$. This assignment is as specified in Table 8-8. The value of $predSampleLX_L$ is the output.

The variables shift1, shift2 and shift3 are derived as follows:

- The variable shift1 is set equal to Min(4, $BitDepth_Y - 8$), the variable shift2 is set equal to 6 and the variable shift3 is set equal to Max(2, $14 - BitDepth_Y$).

Given the luma samples $A_{i,j}$ at full-sample locations ($xA_{i,j}$, $yA_{i,j}$), the luma samples $a_{0,0}$ to $r_{0,0}$ at fractional sample positions are derived as follows:

- The samples labelled $a_{0,0}$, $b_{0,0}$, $c_{0,0}$, $d_{0,0}$, $h_{0,0}$ and $n_{0,0}$ are derived by applying an 8-tap filter to the nearest integer position samples as follows:

$$a_{0,0} = (-A_{-3,0} + 4 * A_{-2,0} - 10 * A_{-1,0} + 58 * A_{0,0} + 17 * A_{1,0} - 5 * A_{2,0} + A_{3,0}) >> shift1 \quad (8\text{-}226)$$

$$b_{0,0} = (-A_{-3,0} + 4 * A_{-2,0} - 11 * A_{-1,0} + 40 * A_{0,0} + 40 * A_{1,0} - 11 * A_{2,0} + 4 * A_{3,0} - A_{4,0}) >> shift1 \quad (8\text{-}227)$$

$$c_{0,0} = (A_{-2,0} - 5 * A_{-1,0} + 17 * A_{0,0} + 58 * A_{1,0} - 10 * A_{2,0} + 4 * A_{3,0} - A_{4,0}) >> shift1 \quad (8\text{-}228)$$

$$d_{0,0} = (-A_{0,-3} + 4 * A_{0,-2} - 10 * A_{0,-1} + 58 * A_{0,0} + 17 * A_{0,1} - 5 * A_{0,2} + A_{0,3}) >> shift1 \quad (8\text{-}229)$$

$$h_{0,0} = (-A_{0,-3} + 4 * A_{0,-2} - 11 * A_{0,-1} + 40 * A_{0,0} + 40 * A_{0,1} - 11 * A_{0,2} + 4 * A_{0,3} - A_{0,4}) >> shift1 \quad (8\text{-}230)$$

$$n_{0,0} = (A_{0,-2} - 5 * A_{0,-1} + 17 * A_{0,0} + 58 * A_{0,1} - 10 * A_{0,2} + 4 * A_{0,3} - A_{0,4}) >> shift1 \quad (8\text{-}231)$$

- The samples labelled $e_{0,0}$, $i_{0,0}$, $p_{0,0}$, $f_{0,0}$, $j_{0,0}$, $q_{0,0}$, $g_{0,0}$, $k_{0,0}$ and $r_{0,0}$ are derived by applying an 8-tap filter to the samples $a_{0,i}$, $b_{0,i}$ and $c_{0,i}$ with i = −3..4 in the vertical direction as follows:

$$e_{0,0} = (-a_{0,-3} + 4 * a_{0,-2} - 10 * a_{0,-1} + 58 * a_{0,0} + 17 * a_{0,1} - 5 * a_{0,2} + a_{0,3}) >> shift2 \quad (8\text{-}232)$$

$$i_{0,0} = (-a_{0,-3} + 4 * a_{0,-2} - 11 * a_{0,-1} + 40 * a_{0,0} + 40 * a_{0,1} - 11 * a_{0,2} + 4 * a_{0,3} - a_{0,4}) >> shift2 \quad (8\text{-}233)$$

$$p_{0,0} = (a_{0,-2} - 5 * a_{0,-1} + 17 * a_{0,0} + 58 * a_{0,1} - 10 * a_{0,2} + 4 * a_{0,3} - a_{0,4}) >> shift2 \quad (8\text{-}234)$$

$$f_{0,0} = (-b_{0,-3} + 4 * b_{0,-2} - 10 * b_{0,-1} + 58 * b_{0,0} + 17 * b_{0,1} - 5 * b_{0,2} + b_{0,3}) >> shift2 \quad (8\text{-}235)$$

$$j_{0,0} = (-b_{0,-3} + 4 * b_{0,-2} - 11 * b_{0,-1} + 40 * b_{0,0} + 40 * b_{0,1} - 11 * b_{0,2} + 4 * b_{0,3} - b_{0,4}) >> shift2 \quad (8\text{-}236)$$

$$q_{0,0} = (b_{0,-2} - 5 * b_{0,-1} + 17 * b_{0,0} + 58 * b_{0,1} - 10 * b_{0,2} + 4 * b_{0,3} - b_{0,4}) >> shift2 \quad (8\text{-}237)$$

$$g_{0,0} = (-c_{0,-3} + 4 * c_{0,-2} - 10 * c_{0,-1} + 58 * c_{0,0} + 17 * c_{0,1} - 5 * c_{0,2} + c_{0,3}) >> shift2 \quad (8\text{-}238)$$

$$k_{0,0} = (-c_{0,-3} + 4 * c_{0,-2} - 11 * c_{0,-1} + 40 * c_{0,0} + 40 * c_{0,1} - 11 * c_{0,2} + 4 * c_{0,3} - c_{0,4}) >> shift2 \quad (8\text{-}239)$$

$$r_{0,0} = (c_{0,-2} - 5 * c_{0,-1} + 17 * c_{0,0} + 58 * c_{0,1} - 10 * c_{0,2} + 4 * c_{0,3} - c_{0,4}) >> shift2 \quad (8\text{-}240)$$

Table 8-8 – Assignment of the luma prediction sample predSampleLX$_L$

xFracL	0	0	0	0	1	1	1	1	2	2	2	2	3	3	3	3
yFracL	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3
predSampleLX$_L$	A << shift3	d	h	n	a	e	i	p	b	f	j	q	c	g	k	r

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 164-65.

8.5.3.3.3.3 Chroma sample interpolation process

This process is only invoked when ChromaArrayType is not equal to 0.

Inputs to this process are:

a chroma location in full-sample units ($xInt_C$, $yInt_C$),

– a chroma location in eighth fractional-sample units ($xFrac_C$, $yFrac_C$),

the chroma reference sample array $refPicLX_C$.

Output of this process is a predicted chroma sample value $predSampleLX_C$:

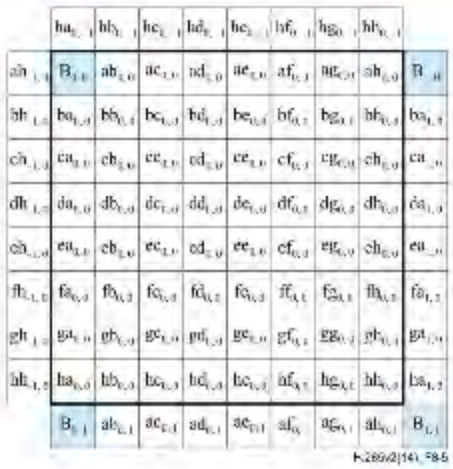


Figure 8-5 – Integer samples (shaded blocks with upper-case letters) and fractional sample positions (un-shaded blocks with lower-case letters) for eighth sample chroma interpolation

In Figure 8-5, the positions labelled with upper-case letters $B_{i,j}$ within shaded blocks represent chroma samples at full-sample locations inside the given two-dimensional array $refPicLX_C$ of chroma samples. These samples may be used for generating the predicted chroma sample value $predSampleLX_C$. The locations ($xB_{i,j}$, $yB_{i,j}$) for each of the corresponding chroma samples $B_{i,j}$ inside the given array $refPicLX_C$ of chroma samples are derived as follows:

$$xB_{i,j} = Clip3(0, (pic_width_in_luma_samples / SubWidthC) - 1, xInt_C + i) \quad (8\text{-}241)$$

$$yB_{i,j} = Clip3(0, (pic_height_in_luma_samples / SubHeightC) - 1, yInt_C + j) \quad (8\text{-}242)$$

The positions labelled with lower-case letters within un-shaded blocks represent chroma samples at eighth-pel sample fractional locations. The chroma location offset in fractional-sample units ($xFrac_C$, $yFrac_C$) specifies which of the generated chroma samples at full-sample and fractional-sample locations is assigned to the predicted chroma sample value $predSampleLX_C$. This assignment is as specified in Table 8-9. The output is the value of $predSampleLX_C$.

The variables shift1, shift2 and shift3 are derived as follows:

– The variable shift1 is set equal to Min(4, $BitDepth_C - 8$), the variable shift2 is set equal to 6 and the variable shift3 is set equal to Max(2, $14 - BitDepth_C$).

Given the chroma samples $B_{i,j}$ at full-sample locations ($xB_{i,j}$, $yB_{i,j}$), the chroma samples $ab_{0,0}$ to $hh_{0,0}$ at fractional sample positions are derived as follows:

– The samples labelled $ab_{0,0}$, $ac_{0,0}$, $ad_{0,0}$, $ae_{0,0}$, $af_{0,0}$, $ag_{0,0}$ and $ah_{0,0}$ are derived by applying a 4-tap filter to the nearest integer position samples as follows:

$$ab_{0,0} = (-2 * B_{-1,0} + 58 * B_{0,0} + 10 * B_{1,0} - 2 * B_{2,0}) >> shift1 \quad (8\text{-}243)$$

$$ac_{0,0} = (-4 * B_{-1,0} + 54 * B_{0,0} + 16 * B_{1,0} - 2 * B_{2,0}) >> shift1 \quad (8\text{-}244)$$

$$ad_{0,0} = (-6 * B_{-1,0} + 46 * B_{0,0} + 28 * B_{1,0} - 4 * B_{2,0}) >> shift1 \quad (8\text{-}245)$$

$$ae_{0,0} = (-4 * B_{-1,0} + 36 * B_{0,0} + 36 * B_{1,0} - 4 * B_{2,0}) >> shift1 \quad (8\text{-}246)$$

$$af_{0,0} = (-4 * B_{-1,0} + 28 * B_{0,0} + 46 * B_{1,0} - 6 * B_{2,0}) >> shift1 \quad (8\text{-}247)$$

$$ag_{0,0} = (-2 * B_{-1,0} + 16 * B_{0,0} + 54 * B_{1,0} - 4 * B_{2,0}) >> shift1 \quad (8\text{-}248)$$

$$ah_{0,0} = (-2 * B_{-1,0} + 10 * B_{0,0} + 58 * B_{1,0} - 2 * B_{2,0}) >> shift1 \quad (8\text{-}249)$$

- The samples labelled $ba_{0,0}$, $ca_{0,0}$, $da_{0,0}$, $ea_{0,0}$, $fa_{0,0}$, $ga_{0,0}$ and $ha_{0,0}$ are derived by applying a 4-tap filter to the nearest integer position samples as follows:

$$ba_{0,0} = (-2 * B_{0,-1} + 58 * B_{0,0} + 10 * B_{0,1} - 2 * B_{0,2}) >> shift1 \quad (8\text{-}250)$$

$$ca_{0,0} = (-4 * B_{0,-1} + 54 * B_{0,0} + 16 * B_{0,1} - 2 * B_{0,2}) >> shift1 \quad (8\text{-}251)$$

$$da_{0,0} = (-6 * B_{0,-1} + 46 * B_{0,0} + 28 * B_{0,1} - 4 * B_{0,2}) >> shift1 \quad (8\text{-}252)$$

$$ea_{0,0} = (-4 * B_{0,-1} + 36 * B_{0,0} + 36 * B_{0,1} - 4 * B_{0,2}) >> shift1 \quad (8\text{-}253)$$

$$fa_{0,0} = (-4 * B_{0,-1} + 28 * B_{0,0} + 46 * B_{0,1} - 6 * B_{0,2}) >> shift1 \quad (8\text{-}254)$$

$$ga_{0,0} = (-2 * B_{0,-1} + 16 * B_{0,0} + 54 * B_{0,1} - 4 * B_{0,2}) >> shift1 \quad (8\text{-}255)$$

$$ha_{0,0} = (-2 * B_{0,-1} + 10 * B_{0,0} + 58 * B_{0,1} - 2 * B_{0,2}) >> shift1 \quad (8\text{-}256)$$

- The samples labelled $bX_{0,0}$, $cX_{0,0}$, $dX_{0,0}$, $eX_{0,0}$, $fX_{0,0}$, $gX_{0,0}$ and $hX_{0,0}$ for X being replaced by b, c, d, e, f, g and h, respectively, are derived by applying a 4-tap filter to the intermediate values $aX_{0,i}$ with $i = -1..2$ in the vertical direction as follows:

$$bX_{0,0} = (-2 * aX_{0,-1} + 58 * aX_{0,0} + 10 * aX_{0,1} - 2 * aX_{0,2}) >> shift2 \quad (8\text{-}257)$$

$$cX_{0,0} = (-4 * aX_{0,-1} + 54 * aX_{0,0} + 16 * aX_{0,1} - 2 * aX_{0,2}) >> shift2 \quad (8\text{-}258)$$

$$dX_{0,0} = (-6 * aX_{0,-1} + 46 * aX_{0,0} + 28 * aX_{0,1} - 4 * aX_{0,2}) >> shift2 \quad (8\text{-}259)$$

$$eX_{0,0} = (-4 * aX_{0,-1} + 36 * aX_{0,0} + 36 * aX_{0,1} - 4 * aX_{0,2}) >> shift2 \quad (8\text{-}260)$$

$$fX_{0,0} = (-4 * aX_{0,-1} + 28 * aX_{0,0} + 46 * aX_{0,1} - 6 * aX_{0,2}) >> shift2 \quad (8\text{-}261)$$

$$gX_{0,0} = (-2 * aX_{0,-1} + 16 * aX_{0,0} + 54 * aX_{0,1} - 4 * aX_{0,2}) >> shift2 \quad (8\text{-}262)$$

$$hX_{0,0} = (-2 * aX_{0,-1} + 10 * aX_{0,0} + 58 * aX_{0,1} - 2 * aX_{0,2}) >> shift2 \quad (8\text{-}263)$$

Table 8-9 – Assignment of the chroma prediction sample predSampleLX$_C$ for (X, Y) being replaced by (1, b), (2, c), (3, d), (4, e), (5, f), (6, g) and (7, h), respectively

xFracC	0	0	0	0	0	0	0	0
yFracC	0	1	2	3	4	5	6	7
predSampleLX$_C$	B << shift3	ba	ca	da	ca	fa	ga	ha
xFracC	X	X	X	X	X	X	X	X
yFracC	0	1	2	3	4	5	6	7
predSampleLX$_C$	aY	bY	cY	dY	eY	fY	gY	hY

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 165-67.

439. When decoding Amazon Prime Video contents, on information and belief, Amazon performs a method of decoding that performs using said second reference block to obtain a second prediction, said second prediction having the second precision, corresponding to the decoding process specified by the H.265 standard.

8.5.3.3 Decoding process for inter prediction samples

8.5.3.3.1 General

Inputs to this process are:

- a luma location (xCb, yCb) specifying the top-left sample of the current luma coding block relative to the top-left luma sample of the current picture,
- a luma location (xBl, yBl) specifying the top-left sample of the current luma prediction block relative to the top-left sample of the current luma coding block,
- a variable nCbS specifying the size of the current luma coding block,
- two variables nPbW and nPbH specifying the width and the height of the luma prediction block,
- the luma motion vectors mvL0 and mvL1,

- when ChromaArrayType is not equal to 0, the chroma motion vectors mvCL0 and mvCL1,
- the reference indices refIdxL0 and refIdxL1,
- the prediction list utilization flags, predFlagL0, and predFlagL1.

Outputs of this process are:

- an $(nCbS_L)x(nCbS_L)$ array $predSamples_L$ of luma prediction samples, where $nCbS_L$ is derived as specified below,
- when ChromaArrayType is not equal to 0, an $(nCbSw_C)x(nCbSh_C)$ array $predSamples_{Cb}$ of chroma prediction samples for the component Cb, where $nCbSw_C$ and $nCbSh_C$ are derived as specified below,
- when ChromaArrayType is not equal to 0, an $(nCbSw_C)x(nCbSh_C)$ array $predSamples_{Cr}$ of chroma prediction samples for the component Cr, where $nCbSw_C$ and $nCbSh_C$ are derived as specified below.

The variable $nCbS_L$ is set equal to nCbS. When ChromaArrayType is not equal to 0, the variable $nCbSw_C$ is set equal to nCbS / SubWidthC and the variable $nCbSh_C$ is set equal to nCbS / SubHeightC.

Let $predSamplesL0_L$ and $predSamplesL1_L$ be (nPbW)x(nPbH) arrays of predicted luma sample values and, when ChromaArrayType is not equal to 0, $predSamplesL0_{Cb}$, $predSamplesL1_{Cb}$, $predSamplesL0_{Cr}$ and $predSamplesL1_{Cr}$ be (nPbW / SubWidthC)x(nPbH / SubHeightC) arrays of predicted chroma sample values.

For X being each of 0 and 1, when predFlagLX is equal to 1, the following applies:

- The reference picture consisting of an ordered two-dimensional array $refPicLX_L$ of luma samples and, when ChromaArrayType is not equal to 0, two ordered two-dimensional arrays $refPicLX_{Cb}$ and $refPicLX_{Cr}$ of chroma samples is derived by invoking the process specified in clause 8.5.3.3.2 with refIdxLX as input.
- The array $predSamplesLX_L$ and, when ChromaArrayType is not equal to 0, the arrays $predSamplesLX_{Cb}$ and $predSamplesLX_{Cr}$ are derived by invoking the fractional sample interpolation process specified in clause 8.5.3.3.3 with the luma locations (xCb, yCb) and (xBl, yBl), the luma prediction block width nPbW, the luma prediction block height nPbH, the motion vectors mvLX and, when ChromaArrayType is not equal to 0, mvCLX, and the reference arrays $refPicLX_L$, $refPicLX_{Cb}$, and $refPicLX_{Cr}$ as inputs.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 161-62.

8.5.3.3.3 Fractional sample interpolation process

8.5.3.3.3.1 General

Inputs to this process are:

- a luma location (xCb, yCb) specifying the top-left sample of the current luma coding block relative to the top-left luma sample of the current picture
- a luma location (xBl, yBl) specifying the top-left sample of the current luma prediction block relative to the top-left sample of the current luma coding block
- two variables nPbW and nPbH specifying the width and the height of the luma prediction block
- a luma motion vector mvLX given in quarter-luma-sample units
- when ChromaArrayType is not equal to 0, a chroma motion vector mvCLX given in eighth-chroma-sample units
- the selected reference picture sample array $refPicLX_L$ and, when ChromaArrayType is not equal to 0, the arrays $refPicLX_{Cb}$ and $refPicLX_{Cr}$.

Outputs of this process are:

- an (nPbW)x(nPbH) array $predSamplesLX_L$ of prediction luma sample values

  when ChromaArrayType is not equal to 0, two (nPbW / SubWidthC)x(nPbH / SubHeightC) arrays $predSamplesLX_{Cb}$ and $predSamplesLX_{Cr}$ of prediction chroma sample values.

The location (xPb, yPb) given in full-sample units of the upper-left luma samples of the current prediction block relative to the upper-left luma sample location of the given reference sample arrays is derived as follows:

$$xPb = xCb + xBl \quad (8\text{-}214)$$

$$yPb = yCb + yBl \quad (8\text{-}215)$$

Let ($xInt_L$, $yInt_L$) be a luma location given in full-sample units and ($xFrac_L$, $yFrac_L$) be an offset given in quarter-sample units. These variables are used only inside this clause for specifying fractional-sample locations inside the reference sample arrays $refPicLX_L$, $refPicLX_{Cb}$ and $refPicLX_{Cr}$.

For each luma sample location ($x_L = 0..nPbW - 1$, $y_L = 0..nPbH - 1$) inside the prediction luma sample array $predSamplesLX_L$, the corresponding prediction luma sample value $predSamplesLX_L[x_L][y_L]$ is derived as follows:

- The variables $xInt_L$, $yInt_L$, $xFrac_L$ and $yFrac_L$ are derived as follows:

$$xInt_L = xPb + (mvLX[0] >> 2) + x_L \quad (8\text{-}216)$$

$$yInt_L = yPb + (mvLX[1] >> 2) + y_L \quad (8\text{-}217)$$

$$xFrac_L = mvLX[0] \& 3 \quad (8\text{-}218)$$

$$yFrac_L = mvLX[1] \& 3 \quad (8\text{-}219)$$

- The prediction luma sample value $predSamplesLX_L[x_L][y_L]$ is derived by invoking the process specified in clause 8.5.3.3.3.2 with ($xInt_L$, $yInt_L$), ($xFrac_L$, $yFrac_L$) and $refPicLX_L$ as inputs.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 163.

8.5.3.3.3.2 Luma sample interpolation process

Inputs to this process are:

- a luma location in full-sample units ($xInt_L$, $yInt_L$),
- a luma location in fractional-sample units ($xFrac_L$, $yFrac_L$),
- the luma reference sample array $refPicLX_L$.

Output of this process is a predicted luma sample value $predSampleLX_L$

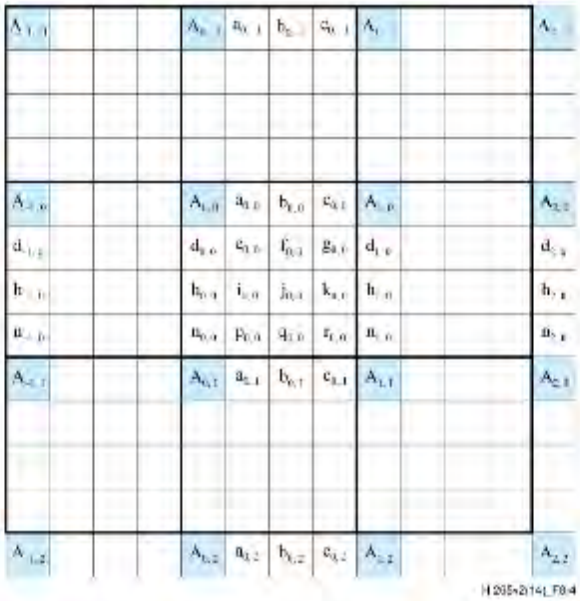


Figure 8-4 – Integer samples (shaded blocks with upper-case letters) and fractional sample positions (un-shaded blocks with lower-case letters) for quarter sample luma interpolation

In Figure 8-4, the positions labelled with upper-case letters $A_{i,j}$ within shaded blocks represent luma samples at full-sample locations inside the given two-dimensional array $refPicLX_L$ of luma samples. These samples may be used for generating the predicted luma sample value $predSampleLX_L$. The locations ($xA_{i,j}$, $yA_{i,j}$) for each of the corresponding luma samples $A_{i,j}$ inside the given array $refPicLX_L$ of luma samples are derived as follows:

$$xA_{i,j} = Clip3(0, pic_width_in_luma_samples - 1, xInt_L + i) \quad (8\text{-}224)$$

$$yA_{i,j} = Clip3(0, pic_height_in_luma_samples - 1, yInt_L + j) \quad (8\text{-}225)$$

The positions labelled with lower-case letters within un-shaded blocks represent luma samples at quarter-luma-sample fractional locations. The luma location offset in fractional-sample units ($xFrac_L$, $yFrac_L$) specifies which of the generated

luma samples at full-sample and fractional-sample locations is assigned to the predicted luma sample value $predSampleLX_L$. This assignment is as specified in Table 8-8. The value of $predSampleLX_L$ is the output.

The variables shift1, shift2 and shift3 are derived as follows:

- The variable shift1 is set equal to Min(4, $BitDepth_Y - 8$), the variable shift2 is set equal to 6 and the variable shift3 is set equal to Max(2, $14 - BitDepth_Y$).

Given the luma samples $A_{i,j}$ at full-sample locations ($xA_{i,j}$, $yA_{i,j}$), the luma samples $a_{0,0}$ to $r_{0,0}$ at fractional sample positions are derived as follows:

- The samples labelled $a_{0,0}$, $b_{0,0}$, $c_{0,0}$, $d_{0,0}$, $h_{0,0}$ and $n_{0,0}$ are derived by applying an 8-tap filter to the nearest integer position samples as follows:

$$a_{0,0} = (-A_{-3,0} + 4 * A_{-2,0} - 10 * A_{-1,0} + 58 * A_{0,0} + 17 * A_{1,0} - 5 * A_{2,0} + A_{3,0}) >> shift1 \quad (8\text{-}226)$$

$$b_{0,0} = (-A_{-3,0} + 4 * A_{-2,0} - 11 * A_{-1,0} + 40 * A_{0,0} + 40 * A_{1,0} - 11 * A_{2,0} + 4 * A_{3,0} - A_{4,0}) >> shift1 \quad (8\text{-}227)$$

$$c_{0,0} = (A_{-2,0} - 5 * A_{-1,0} + 17 * A_{0,0} + 58 * A_{1,0} - 10 * A_{2,0} + 4 * A_{3,0} - A_{4,0}) >> shift1 \quad (8\text{-}228)$$

$$d_{0,0} = (-A_{0,-3} + 4 * A_{0,-2} - 10 * A_{0,-1} + 58 * A_{0,0} + 17 * A_{0,1} - 5 * A_{0,2} + A_{0,3}) >> shift1 \quad (8\text{-}229)$$

$$h_{0,0} = (-A_{0,-3} + 4 * A_{0,-2} - 11 * A_{0,-1} + 40 * A_{0,0} + 40 * A_{0,1} - 11 * A_{0,2} + 4 * A_{0,3} - A_{0,4}) >> shift1 \quad (8\text{-}230)$$

$$n_{0,0} = (A_{0,-2} - 5 * A_{0,-1} + 17 * A_{0,0} + 58 * A_{0,1} - 10 * A_{0,2} + 4 * A_{0,3} - A_{0,4}) >> shift1 \quad (8\text{-}231)$$

- The samples labelled $e_{0,0}$, $i_{0,0}$, $p_{0,0}$, $f_{0,0}$, $j_{0,0}$, $q_{0,0}$, $g_{0,0}$, $k_{0,0}$ and $r_{0,0}$ are derived by applying an 8-tap filter to the samples $a_{0,i}$, $b_{0,i}$ and $c_{0,i}$ with i = −3..4 in the vertical direction as follows:

$$e_{0,0} = (-a_{0,-3} + 4 * a_{0,-2} - 10 * a_{0,-1} + 58 * a_{0,0} + 17 * a_{0,1} - 5 * a_{0,2} + a_{0,3}) >> shift2 \quad (8\text{-}232)$$

$$i_{0,0} = (-a_{0,-3} + 4 * a_{0,-2} - 11 * a_{0,-1} + 40 * a_{0,0} + 40 * a_{0,1} - 11 * a_{0,2} + 4 * a_{0,3} - a_{0,4}) >> shift2 \quad (8\text{-}233)$$

$$p_{0,0} = (a_{0,-2} - 5 * a_{0,-1} + 17 * a_{0,0} + 58 * a_{0,1} - 10 * a_{0,2} + 4 * a_{0,3} - a_{0,4}) >> shift2 \quad (8\text{-}234)$$

$$f_{0,0} = (-b_{0,-3} + 4 * b_{0,-2} - 10 * b_{0,-1} + 58 * b_{0,0} + 17 * b_{0,1} - 5 * b_{0,2} + b_{0,3}) >> shift2 \quad (8\text{-}235)$$

$$j_{0,0} = (-b_{0,-3} + 4 * b_{0,-2} - 11 * b_{0,-1} + 40 * b_{0,0} + 40 * b_{0,1} - 11 * b_{0,2} + 4 * b_{0,3} - b_{0,4}) >> shift2 \quad (8\text{-}236)$$

$$q_{0,0} = (b_{0,-2} - 5 * b_{0,-1} + 17 * b_{0,0} + 58 * b_{0,1} - 10 * b_{0,2} + 4 * b_{0,3} - b_{0,4}) >> shift2 \quad (8\text{-}237)$$

$$g_{0,0} = (-c_{0,-3} + 4 * c_{0,-2} - 10 * c_{0,-1} + 58 * c_{0,0} + 17 * c_{0,1} - 5 * c_{0,2} + c_{0,3}) >> shift2 \quad (8\text{-}238)$$

$$k_{0,0} = (-c_{0,-3} + 4 * c_{0,-2} - 11 * c_{0,-1} + 40 * c_{0,0} + 40 * c_{0,1} - 11 * c_{0,2} + 4 * c_{0,3} - c_{0,4}) >> shift2 \quad (8\text{-}239)$$

$$r_{0,0} = (c_{0,-2} - 5 * c_{0,-1} + 17 * c_{0,0} + 58 * c_{0,1} - 10 * c_{0,2} + 4 * c_{0,3} - c_{0,4}) >> shift2 \quad (8\text{-}240)$$

Table 8-8 – Assignment of the luma prediction sample predSampleLX$_L$

xFracL	0	0	0	0	1	1	1	1	2	2	2	2	3	3	3	3
yFracL	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3
predSampleLX$_L$	A << shift3	d	h	n	a	e	i	p	b	f	j	q	c	g	k	r

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 164-65.

8.5.3.3.3.3 Chroma sample interpolation process

This process is only invoked when ChromaArrayType is not equal to 0.

Inputs to this process are:

- a chroma location in full-sample units ($xInt_C$, $yInt_C$).

- a chroma location in eighth fractional-sample units ($xFrac_C$, $yFrac_C$),
- the chroma reference sample array $refPicLX_C$.

Output of this process is a predicted chroma sample value $predSampleLX_C$

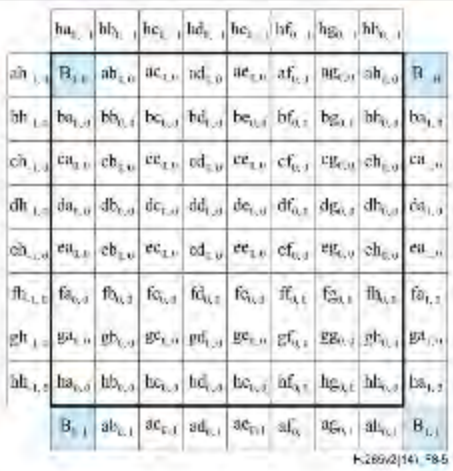


Figure 8-5 – Integer samples (shaded blocks with upper-case letters) and fractional sample positions (un-shaded blocks with lower-case letters) for eighth sample chroma interpolation

In Figure 8-5, the positions labelled with upper-case letters $B_{i,j}$ within shaded blocks represent chroma samples at full-sample locations inside the given two-dimensional array $refPicLX_C$ of chroma samples. These samples may be used for generating the predicted chroma sample value $predSampleLX_C$. The locations ($xB_{i,j}$, $yB_{i,j}$) for each of the corresponding chroma samples $B_{i,j}$ inside the given array $refPicLX_C$ of chroma samples are derived as follows:

$$xB_{i,j} = Clip3(0, (pic_width_in_luma_samples / SubWidthC) - 1, xInt_C + i) \quad (8\text{-}241)$$

$$yB_{i,j} = Clip3(0, (pic_height_in_luma_samples / SubHeightC) - 1, yInt_C + j) \quad (8\text{-}242)$$

The positions labelled with lower-case letters within un-shaded blocks represent chroma samples at eighth-pel sample fractional locations. The chroma location offset in fractional-sample units ($xFrac_C$, $yFrac_C$) specifies which of the generated chroma samples at full-sample and fractional-sample locations is assigned to the predicted chroma sample value $predSampleLX_C$. This assignment is as specified in Table 8-9. The output is the value of $predSampleLX_C$.

The variables shift1, shift2 and shift3 are derived as follows:

– The variable shift1 is set equal to Min(4, $BitDepth_C$ − 8), the variable shift2 is set equal to 6 and the variable shift3 is set equal to Max(2, 14 − $BitDepth_C$).

Given the chroma samples $B_{i,j}$ at full-sample locations ($xB_{i,j}$, $yB_{i,j}$), the chroma samples $ab_{0,0}$ to $hh_{0,0}$ at fractional sample positions are derived as follows:

– The samples labelled $ab_{0,0}$, $ac_{0,0}$, $ad_{0,0}$, $ae_{0,0}$, $af_{0,0}$, $ag_{0,0}$ and $ah_{0,0}$ are derived by applying a 4-tap filter to the nearest integer position samples as follows:

$$ab_{0,0} = (-2 * B_{-1,0} + 58 * B_{0,0} + 10 * B_{1,0} - 2 * B_{2,0}) >> shift1 \quad (8\text{-}243)$$

$$ac_{0,0} = (-4 * B_{-1,0} + 54 * B_{0,0} + 16 * B_{1,0} - 2 * B_{2,0}) >> shift1 \quad (8\text{-}244)$$

$$ad_{0,0} = (-6 * B_{-1,0} + 46 * B_{0,0} + 28 * B_{1,0} - 4 * B_{2,0}) >> shift1 \quad (8\text{-}245)$$

$$ae_{0,0} = (-4 * B_{-1,0} + 36 * B_{0,0} + 36 * B_{1,0} - 4 * B_{2,0}) >> shift1 \quad (8\text{-}246)$$

$$af_{0,0} = (-4 * B_{-1,0} + 28 * B_{0,0} + 46 * B_{1,0} - 6 * B_{2,0}) >> shift1 \quad (8\text{-}247)$$

$$ag_{0,0} = (-2 * B_{-1,0} + 16 * B_{0,0} + 54 * B_{1,0} - 4 * B_{2,0}) >> shift1 \quad (8\text{-}248)$$

$$ah_{0,0} = (-2 * B_{-1,0} + 10 * B_{0,0} + 58 * B_{1,0} - 2 * B_{2,0}) >> shift1 \quad (8\text{-}249)$$

– The samples labelled $ba_{0,0}$, $ca_{0,0}$, $da_{0,0}$, $ea_{0,0}$, $fa_{0,0}$, $ga_{0,0}$ and $ha_{0,0}$ are derived by applying a 4-tap filter to the nearest integer position samples as follows:

$$ba_{0,0} = (-2 * B_{0,-1} + 58 * B_{0,0} + 10 * B_{0,1} - 2 * B_{0,2}) >> shift1 \quad (8\text{-}250)$$

$$ca_{0,0} = (-4 * B_{0,-1} + 54 * B_{0,0} + 16 * B_{0,1} - 2 * B_{0,2}) >> shift1 \quad (8\text{-}251)$$

$$da_{0,0} = (-6 * B_{0,-1} + 46 * B_{0,0} + 28 * B_{0,1} - 4 * B_{0,2}) >> shift1 \quad (8\text{-}252)$$

$$ea_{0,0} = (-4 * B_{0,-1} + 36 * B_{0,0} + 36 * B_{0,1} - 4 * B_{0,2}) >> shift1 \quad (8\text{-}253)$$

$$fa_{0,0} = (-4 * B_{0,-1} + 28 * B_{0,0} + 46 * B_{0,1} - 6 * B_{0,2}) >> shift1 \quad (8\text{-}254)$$

$$ga_{0,0} = (-2 * B_{0,-1} + 16 * B_{0,0} + 54 * B_{0,1} - 4 * B_{0,2}) >> shift1 \quad (8\text{-}255)$$

$$ha_{0,0} = (-2 * B_{0,-1} + 10 * B_{0,0} + 58 * B_{0,1} - 2 * B_{0,2}) >> shift1 \quad (8\text{-}256)$$

– The samples labelled $bX_{0,0}$, $cX_{0,0}$, $dX_{0,0}$, $eX_{0,0}$, $fX_{0,0}$, $gX_{0,0}$ and $hX_{0,0}$ for X being replaced by b, c, d, e, f, g and h, respectively, are derived by applying a 4-tap filter to the intermediate values $aX_{0,i}$ with $i = -1..2$ in the vertical direction as follows:

$$bX_{0,0} = (-2 * aX_{0,-1} + 58 * aX_{0,0} + 10 * aX_{0,1} - 2 * aX_{0,2}) >> shift2 \quad (8\text{-}257)$$

$$cX_{0,0} = (-4 * aX_{0,-1} + 54 * aX_{0,0} + 16 * aX_{0,1} - 2 * aX_{0,2}) >> shift2 \quad (8\text{-}258)$$

$$dX_{0,0} = (-6 * aX_{0,-1} + 46 * aX_{0,0} + 28 * aX_{0,1} - 4 * aX_{0,2}) >> shift2 \quad (8\text{-}259)$$

$$eX_{0,0} = (-4 * aX_{0,-1} + 36 * aX_{0,0} + 36 * aX_{0,1} - 4 * aX_{0,2}) >> shift2 \quad (8\text{-}260)$$

$$fX_{0,0} = (-4 * aX_{0,-1} + 28 * aX_{0,0} + 46 * aX_{0,1} - 6 * aX_{0,2}) >> shift2 \quad (8\text{-}261)$$

$$gX_{0,0} = (-2 * aX_{0,-1} + 16 * aX_{0,0} + 54 * aX_{0,1} - 4 * aX_{0,2}) >> shift2 \quad (8\text{-}262)$$

$$hX_{0,0} = (-2 * aX_{0,-1} + 10 * aX_{0,0} + 58 * aX_{0,1} - 2 * aX_{0,2}) >> shift2 \quad (8\text{-}263)$$

Table 8-9 – Assignment of the chroma prediction sample predSampleLX$_C$ for (X, Y) being replaced by (1, b), (2, c), (3, d), (4, e), (5, f), (6, g) and (7, h), respectively

xFracC	0	0	0	0	0	0	0	0
yFracC	0	1	2	3	4	5	6	7
predSampleLX$_C$	B << shift3	ba	ca	da	ea	fa	ga	ha
xFracC	X	X	X	X	X	X	X	X
yFracC	0	1	2	3	4	5	6	7
predSampleLX$_C$	aY	bY	cY	dY	eY	fY	gY	hY

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 165-67.

440. When decoding Amazon Prime Video contents, on information and belief, Amazon performs a method of decoding that performs obtaining a combined prediction based at least partly upon said first prediction and said second prediction, corresponding to the decoding process specified by the H.265 standard.

8.5.3.3.4 Weighted sample prediction process

8.5.3.3.4.1 General

Inputs to this process are:

- two variables nPbW and nPbH specifying the width and the height of the current prediction block,
- two (nPbW)x(nPbH) arrays predSamplesL0 and predSamplesL1,
- the prediction list utilization flags, predFlagL0 and predFlagL1,

the reference indices refIdxL0 and refIdxL1,

a variable cIdx specifying colour component index.

Output of this process is the (nPbW)x(nPbH) array pbSamples of prediction sample values.

The variable bitDepth is derived as follows:

- If cIdx is equal to 0, bitDepth is set equal to $BitDepth_Y$.
- Otherwise, bitDepth is set equal to $BitDepth_C$.

The variable weightedPredFlag is derived as follows:

- If slice_type is equal to P, weightedPredFlag is set equal to weighted_pred_flag.
- Otherwise (slice_type is equal to B), weightedPredFlag is set equal to weighted_bipred_flag.

The following applies:

- If weightedPredFlag is equal to 0, the array pbSamples of the prediction samples is derived by invoking the default weighted sample prediction process as specified in clause 8.5.3.3.4.2 with the prediction block width nPbW, the prediction block height nPbH, two (nPbW)x(nPbH) arrays predSamplesL0 and predSamplesL1, the prediction list utilization flags predFlagL0 and predFlagL1 and the bit depth bitDepth as inputs.
- Otherwise (weightedPredFlag is equal to 1), the array pbSamples of the prediction samples is derived by invoking the weighted sample prediction process as specified in clause 8.5.3.3.4.3 with the prediction block width nPbW, the prediction block height nPbH, two (nPbW)x(nPbH) arrays predSamplesL0 and predSamplesL1, the prediction list utilization flags predFlagL0 and predFlagL1, the reference indices refIdxL0 and refIdxL1, the colour component index cIdx and the bit depth bitDepth as inputs.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 167-68.

8.5.3.3.4.2 Default weighted sample prediction process

Inputs to this process are:

- two variables nPbW and nPbH specifying the width and the height of the current prediction block,
- two (nPbW)x(nPbH) arrays predSamplesL0 and predSamplesL1,
- the prediction list utilization flags, predFlagL0, and predFlagL1,
- a bit depth of samples, bitDepth.

Output of this process is the (nPbW)x(nPbH) array pbSamples of prediction sample values.

Variables shift1, shift2, offset1 and offset2 are derived as follows:

- The variable shift1 is set equal to Max(2, 14 − bitDepth) and the variable shift2 is set equal to Max(3, 15 − bitDepth).
- The variable offset1 is set equal to 1 << (shift1 − 1).
- The variable offset2 is set equal to 1 << (shift2 − 1).

Depending on the values of predFlagL0 and predFlagL1, the prediction samples pbSamples[x][y] with x = 0..nPbW − 1 and y = 0..nPbH − 1 are derived as follows:

- If predFlagL0 is equal to 1 and predFlagL1 is equal to 0, the prediction sample values are derived as follows:

  pbSamples[x][y] = Clip3(0, (1 << bitDepth) − 1, (predSamplesL0[x][y] + offset1) >> shift1) (8-264)

- Otherwise, if predFlagL0 is equal to 0 and predFlagL1 is equal to 1, the prediction sample values are derived as follows:

  pbSamples[x][y] = Clip3(0, (1 << bitDepth) − 1, (predSamplesL1[x][y] + offset1) >> shift1) (8-265)

- Otherwise (predFlagL0 is equal to 1 and predFlagL1 is equal to 1), the prediction sample values are derived as follows:

  pbSamples[x][y] = Clip3(0, (1 << bitDepth) − 1,
  (predSamplesL0[x][y] + predSamplesL1[x][y] + offset2) >> shift2) (8-266)

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 168.

441. When decoding Amazon Prime Video contents, on information and belief, Amazon performs a method of decoding that performs decreasing a precision of said combined prediction by shifting bits of the combined prediction to the right, corresponding to the decoding process specified by the H.265 standard.

5.5 Bit-wise operators

The following bit-wise operators are defined as follows:

x >> y	Arithmetic right shift of a two's complement integer representation of x by y binary digits. This function is defined only for non-negative integer values of y. Bits shifted into the most significant bits (MSBs) as a result of the right shift have a value equal to the MSB of x prior to the shift operation.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 16.

8.5.3.3.4.2 Default weighted sample prediction process

Inputs to this process are:

- two variables nPbW and nPbH specifying the width and the height of the current prediction block,
- two (nPbW)x(nPbH) arrays predSamplesL0 and predSamplesL1,
- the prediction list utilization flags, predFlagL0, and predFlagL1,
- a bit depth of samples, bitDepth.

Output of this process is the (nPbW)x(nPbH) array pbSamples of prediction sample values.

Variables shift1, shift2, offset1 and offset2 are derived as follows:

- The variable shift1 is set equal to Max(2, 14 − bitDepth) and the variable shift2 is set equal to Max(3, 15 − bitDepth).
- The variable offset1 is set equal to 1 << (shift1 − 1).
- The variable offset2 is set equal to 1 << (shift2 − 1).

Depending on the values of predFlagL0 and predFlagL1, the prediction samples pbSamples[x][y] with x = 0..nPbW − 1 and y = 0..nPbH − 1 are derived as follows:

- If predFlagL0 is equal to 1 and predFlagL1 is equal to 0, the prediction sample values are derived as follows:

  pbSamples[x][y] = Clip3(0, (1 << bitDepth) − 1, (predSamplesL0[x][y] + offset1) >> shift1) (8-264)

- Otherwise, if predFlagL0 is equal to 0 and predFlagL1 is equal to 1, the prediction sample values are derived as follows:

  pbSamples[x][y] = Clip3(0, (1 << bitDepth) − 1, (predSamplesL1[x][y] + offset1) >> shift1) (8-265)

- Otherwise (predFlagL0 is equal to 1 and predFlagL1 is equal to 1), the prediction sample values are derived as follows:

  pbSamples[x][y] = Clip3(0, (1 << bitDepth) − 1,
  (predSamplesL0[x][y] + predSamplesL1[x][y] + offset2) >> shift2) (8-266)

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 168.

442. When decoding Amazon Prime Video contents, on information and belief, Amazon performs a method of decoding that performs reconstructing the block of pixels based on the combined prediction, corresponding to the decoding process specified by the H.265 standard.

8.5 Decoding process for coding units coded in inter prediction mode

8.5.1 General decoding process for coding units coded in inter prediction mode

Inputs to this process are:

- a luma location (xCb, yCb) specifying the top-left sample of the current luma coding block relative to the top-left luma sample of the current picture,
- a variable log2CbSize specifying the size of the current coding block.

Output of this process is a modified reconstructed picture before deblocking filtering.

The derivation process for quantization parameters as specified in clause 8.6.1 is invoked with the luma location (xCb, yCb) as input.

The variable $nCbS_L$ is set equal to 1 << log2CbSize. When ChromaArrayType is not equal to 0, the variable $nCbSw_C$ is set equal to (1 << log2CbSize) / SubWidthC and the variable $nCbSh_C$ is set equal to (1 << log2CbSize) / SubHeightC.

The decoding process for coding units coded in inter prediction mode consists of the following ordered steps:

1. The inter prediction process as specified in clause 8.5.2 is invoked with the luma location (xCb, yCb) and the luma coding block size log2CbSize as inputs, and the outputs are the array $predSamples_L$ and, when ChromaArrayType is not equal to 0, the arrays $predSamples_{Cb}$ and $predSamples_{Cr}$.
2. The decoding process for the residual signal of coding units coded in inter prediction mode specified in clause 8.5.4 is invoked with the luma location (xCb, yCb) and the luma coding block size log2CbSize as inputs, and the outputs are the array $resSamples_L$ and, when ChromaArrayType is not equal to 0, the arrays $resSamples_{Cb}$ and $resSamples_{Cr}$.
3. The reconstructed samples of the current coding unit are derived as follows:
   - The picture construction process prior to in-loop filtering for a colour component as specified in clause 8.6.7 is invoked with the luma coding block location (xCb, yCb), the variable nCurrSw set equal to $nCbS_L$, the variable nCurrSh set equal to $nCbS_L$, the variable cIdx set equal to 0, the $(nCbS_L)x(nCbS_L)$ array predSamples set equal to $predSamples_L$ and the $(nCbS_L)x(nCbS_L)$ array resSamples set equal to $resSamples_L$ as inputs.
   - When ChromaArrayType is not equal to 0, the picture construction process prior to in-loop filtering for a colour component as specified in clause 8.6.7 is invoked with the chroma coding block location (xCb / SubWidthC, yCb / SubHeightC), the variable nCurrSw set equal to $nCbSw_C$, the variable nCurrSh set equal to $nCbSh_C$, the variable cIdx set equal to 1, the $(nCbSw_C)x(nCbSh_C)$ array predSamples set equal to $predSamples_{Cb}$ and the $(nCbSw_C)x(nCbSh_C)$ array resSamples set equal to $resSamples_{Cb}$ as inputs.
   - When ChromaArrayType is not equal to 0, the picture construction process prior to in-loop filtering for a colour component as specified in clause 8.6.7 is invoked with the chroma coding block location (xCb / SubWidthC, yCb / SubHeightC), the variable nCurrSw set equal to $nCbSw_C$, the variable nCurrSh set equal to $nCbSh_C$, the variable cIdx set equal to 2, the $(nCbSw_C)x(nCbSh_C)$ array predSamples set equal to $predSamples_{Cr}$ and the $(nCbSw_C)x(nCbSh_C)$ array resSamples set equal to $resSamples_{Cr}$ as inputs.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 141-42.

8.5.2 Inter prediction process

This process is invoked when decoding coding unit whose CuPredMode[xCb][yCb] is not equal to MODE_INTRA.

Inputs to this process are:

- a luma location (xCb, yCb) specifying the top-left sample of the current luma coding block relative to the top-left luma sample of the current picture,
- a variable log2CbSize specifying the size of the current luma coding block.

Outputs of this process are:

- an $(nCbS_L)x(nCbS_L)$ array $predSamples_L$ of luma prediction samples, where $nCbS_L$ is derived as specified below,
- when ChromaArrayType is not equal to 0, an $(nCbSw_C)x(nCbSh_C)$ array $predSamples_{Cb}$ of chroma prediction samples for the component Cb, where $nCbSw_C$ and $nCbSh_C$ are derived as specified below,
- when ChromaArrayType is not equal to 0, an $(nCbSw_C)x(nCbSh_C)$ array $predSamples_{Cr}$ of chroma prediction samples for the component Cr, where $nCbSw_C$ and $nCbSh_C$ are derived as specified below.

The variable $nCbS_L$ is set equal to 1 << log2CbSize. When ChromaArrayType is not equal to 0, the variable $nCbSw_C$ is set equal to $nCbS_L$ / SubWidthC and the variable $nCbSh_C$ is set equal to $nCbS_L$ / SubHeightC.

The variable $nCbS1_L$ is set equal to $nCbS_L$ >> 1.

Depending on the value of PartMode, the following applies:

- If PartMode is equal to PART_2Nx2N, the decoding process for prediction units in inter prediction mode as specified in clause 8.5.3 is invoked with the luma location (xCb, yCb), the luma location (xBl, yBl) set equal to (0, 0), the size of the luma coding block $nCbS_L$, the width of the luma prediction block nPbW set equal to $nCbS_L$, the height of the luma prediction block nPbH set equal to $nCbS_L$ and a partition index partIdx set equal to 0 as inputs, and the outputs are an $(nCbS_L)x(nCbS_L)$ array $predSamples_L$ and, when ChromaArrayType is not equal to 0, two $(nCbSw_C)x(nCbSh_C)$ arrays $predSamples_{Cb}$ and $predSamples_{Cr}$.
- Otherwise, if PartMode is equal to PART_2NxN, the following ordered steps apply:
  1. The decoding process for prediction units in inter prediction mode as specified in clause 8.5.3 is invoked with the luma location (xCb, yCb), the luma location (xBl, yBl) set equal to (0, 0), the size of the luma coding block $nCbS_L$, the width of the luma prediction block nPbW set equal to $nCbS_L$, the height of the luma prediction block nPbH set equal to $nCbS_L$ >> 1 and a partition index partIdx set equal to 0 as inputs, and the outputs are an $(nCbS_L)x(nCbS_L)$ array $predSamples_L$ and, when ChromaArrayType is not equal to 0, two $(nCbSw_C)x(nCbSh_C)$ arrays $predSamples_{Cb}$ and $predSamples_{Cr}$.
  2. The decoding process for prediction units in inter prediction mode as specified in clause 8.5.3 is invoked with the luma location (xCb, yCb), the luma location (xBl, yBl) set equal to (0, $nCbS_L$ >> 1), the size of the luma coding block $nCbS_L$, the width of the luma prediction block nPbW set equal to $nCbS_L$, the height of the luma prediction block nPbH set equal to $nCbS_L$ >> 1 and a partition index partIdx set equal to 1 as inputs, and the outputs are the modified $(nCbS_L)x(nCbS_L)$ array $predSamples_L$ and, when ChromaArrayType is not equal to 0, the two modified $(nCbSw_C)x(nCbSh_C)$ arrays $predSamples_{Cb}$ and $predSamples_{Cr}$.
- Otherwise, if PartMode is equal to PART_Nx2N, the following ordered steps apply:

1. The decoding process for prediction units in inter prediction mode as specified in clause 8.5.3 is invoked with the luma location (xCb, yCb), the luma location (xBl, yBl) set equal to (0, 0), the size of the luma coding block $nCbS_L$, the width of the luma prediction block nPbW set equal to $nCbS_L$ >> 1, the height of the luma prediction block nPbH set equal to $nCbS_L$ and a partition index partIdx set equal to 0 as inputs, and the outputs are an $(nCbS_L)x(nCbS_L)$ array $predSamples_L$ and, when ChromaArrayType is not equal to 0, two $(nCbSw_C)x(nCbSh_C)$ arrays $predSamples_{Cb}$ and $predSamples_{Cr}$.

2. The decoding process for prediction units in inter prediction mode as specified in clause 8.5.3 is invoked with the luma location (xCb, yCb), the luma location (xBl, yBl) set equal to ($nCbS_L$ >> 1, 0), the size of the luma coding block $nCbS_L$, the width of the luma prediction block nPbW set equal to $nCbS_L$ >> 1, the height of the luma prediction block nPbH set equal to $nCbS_L$ and a partition index partIdx set equal to 1 as inputs, and the outputs are the modified $(nCbS_L)x(nCbS_L)$ array $predSamples_L$ and, when ChromaArrayType is not equal to 0, the two modified $(nCbSw_C)x(nCbSh_C)$ arrays $predSamples_{Cb}$ and $predSamples_{Cr}$.

– Otherwise, if PartMode is equal to PART_2NxnU, the following ordered steps apply:

1. The decoding process for prediction units in inter prediction mode as specified in clause 8.5.3 is invoked with the luma location (xCb, yCb), the luma location (xBl, yBl) set equal to (0, 0), the size of the luma coding block $nCbS_L$, the width of the luma prediction block nPbW set equal to $nCbS_L$, the height of the luma prediction block nPbH set equal to $nCbS_L$ >> 2 and a partition index partIdx set equal to 0 as inputs, and the outputs are an $(nCbS_L)x(nCbS_L)$ array $predSamples_L$ and, when ChromaArrayType is not equal to 0, two $(nCbSw_C)x(nCbSh_C)$ arrays $predSamples_{Cb}$ and $predSamples_{Cr}$.

2. The decoding process for prediction units in inter prediction mode as specified in clause 8.5.3 is invoked with the luma location (xCb, yCb), the luma location (xBl, yBl) set equal to (0, $nCbS_L$ >> 2), the size of the luma coding block $nCbS_L$, the width of the luma prediction block nPbW set equal to $nCbS_L$, the height of the luma prediction block nPbH set equal to ($nCbS_L$ >> 1) + ($nCbS_L$ >> 2) and a partition index partIdx set equal to 1 as inputs, and the outputs are the modified $(nCbS_L)x(nCbS_L)$ array $predSamples_L$ and, when ChromaArrayType is not equal to 0, the two modified $(nCbSw_C)x(nCbSh_C)$ arrays $predSamples_{Cb}$ and $predSamples_{Cr}$.

– Otherwise, if PartMode is equal to PART_2NxnD, the following ordered steps apply:

1. The decoding process for prediction units in inter prediction mode as specified in clause 8.5.3 is invoked with the luma location (xCb, yCb), the luma location (xBl, yBl) set equal to (0, 0), the size of the luma coding block $nCbS_L$, the width of the luma prediction block nPbW set equal to $nCbS_L$, the height of the luma prediction block nPbH set equal to ($nCbS_L$ >> 1) + ($nCbS_L$ >> 2) and a partition index partIdx set equal to 0 as inputs, and the outputs are an $(nCbS_L)x(nCbS_L)$ array $predSamples_L$ and, when ChromaArrayType is not equal to 0, two $(nCbSw_C)x(nCbSh_C)$ arrays $predSamples_{Cb}$ and $predSamples_{Cr}$.

2. The decoding process for prediction units in inter prediction mode as specified in clause 8.5.3 is invoked with the luma location (xCb, yCb), the luma location (xBl, yBl) set equal to (0, ($nCbS_L$ >> 1) + ($nCbS_L$ >> 2)), the size of the luma coding block $nCbS_L$, the width of the luma prediction block nPbW set equal to $nCbS_L$, the height of the luma prediction block nPbH set equal to $nCbS_L$ >> 2 and a partition index partIdx set equal to 1 as inputs, and the outputs are the modified $(nCbS_L)x(nCbS_L)$ array $predSamples_L$ and, when ChromaArrayType is not equal to 0, the two modified $(nCbSw_C)x(nCbSh_C)$ arrays $predSamples_{Cb}$ and $predSamples_{Cr}$.

– Otherwise, if PartMode is equal to PART_nLx2N, the following ordered steps apply:

1. The decoding process for prediction units in inter prediction mode as specified in clause 8.5.3 is invoked with the luma location (xCb, yCb), the luma location (xBl, yBl) set equal to (0, 0), the size of the luma coding block $nCbS_L$, the width of the luma prediction block nPbW set equal to $nCbS_L$ >> 2, the height of the luma prediction block nPbH set equal to $nCbS_L$ and a partition index partIdx set equal to 0 as inputs, and the outputs are an $(nCbS_L)x(nCbS_L)$ array $predSamples_L$ and, when ChromaArrayType is not equal to 0, two $(nCbSw_C)x(nCbSh_C)$ arrays $predSamples_{Cb}$ and $predSamples_{Cr}$.

2. The decoding process for prediction units in inter prediction mode as specified in clause 8.5.3 is invoked with the luma location (xCb, yCb), the luma location (xBl, yBl) set equal to ($nCbS_L$ >> 2, 0), the size of the luma coding block $nCbS_L$, the width of the luma prediction block nPbW set equal to ($nCbS_L$ >> 1) + ($nCbS_L$ >> 2), the height of the luma prediction block nPbH set equal to $nCbS_L$ and a partition index partIdx set equal to 1 as inputs, and the outputs are the modified $(nCbS_L)x(nCbS_L)$ array $predSamples_L$ and, when ChromaArrayType is not equal to 0, the two modified $(nCbSw_C)x(nCbSh_C)$ arrays $predSamples_{Cb}$ and $predSamples_{Cr}$.

– Otherwise, if PartMode is equal to PART_nRx2N, the following ordered steps apply:

1. The decoding process for prediction units in inter prediction mode as specified in clause 8.5.3 is invoked with the luma location (xCb, yCb), the luma location (xBl, yBl) set equal to (0, 0), the size of the luma coding block $nCbS_L$, the width of the luma prediction block nPbW set equal to ($nCbS_L$ >> 1) + ($nCbS_L$ >> 2), the height of the luma prediction block nPbH set equal to $nCbS_L$ and a partition index partIdx set equal to 0 as inputs, and the outputs are an ($nCbS_L$)x($nCbS_L$) array $predSamples_L$ and, when ChromaArrayType is not equal to 0, two ($nCbSw_C$)x($nCbSh_C$) arrays $predSamples_{Cb}$ and $predSamples_{Cr}$.

2. The decoding process for prediction units in inter prediction mode as specified in clause 8.5.3 is invoked with the luma location (xCb, yCb), the luma location (xBl, yBl) set equal to ($nCS1_L$ + ($nCbS_L$ >> 2), 0), the size of the luma coding block $nCbS_L$, the width of the luma prediction block nPbW set equal to $nCbS_L$ >> 2, the height of the luma prediction block nPbH set equal to $nCbS_L$ and a partition index partIdx set equal to 1 as inputs, and the outputs are the modified ($nCbS_L$)x($nCbS_L$) array $predSamples_L$ and, when ChromaArrayType is not equal to 0, the two modified ($nCbSw_C$)x($nCbSh_C$) arrays $predSamples_{Cb}$ and $predSamples_{Cr}$.

– Otherwise (PartMode is equal to PART_NxN), the following ordered steps apply:

1. The decoding process for prediction units in inter prediction mode as specified in clause 8.5.3 is invoked with the luma location (xCb, yCb), the luma location (xBl, yBl) set equal to (0, 0), the size of the luma coding block $nCbS_L$, the width of the luma prediction block nPbW set equal to $nCbS_L$ >> 1, the height of the luma prediction block nPbH set equal to $nCbS_L$ >> 1 and a partition index partIdx set equal to 0 as inputs, and the outputs are an ($nCbS_L$)x($nCbS_L$) array $predSamples_L$ and, when ChromaArrayType is not equal to 0, two ($nCbSw_C$)x($nCbSh_C$) arrays $predSamples_{Cb}$ and $predSamples_{Cr}$.

2. The decoding process for prediction units in inter prediction mode as specified in clause 8.5.3 is invoked with the luma location (xCb, yCb), the luma location (xBl, yBl) set equal to ($nCbS_L$ >> 1, 0), the size of the luma coding block $nCbS_L$, the width of the luma prediction block nPbW set equal to $nCbS_L$ >> 1, the height of the luma prediction block nPbH set equal to $nCbS_L$ >> 1 and a partition index partIdx set equal to 1 as inputs, and the outputs are the modified ($nCbS_L$)x($nCbS_L$) array $predSamples_L$ and, when ChromaArrayType is not equal to 0, the two modified ($nCbSw_C$)x($nCbSh_C$) arrays $predSamples_{Cb}$ and $predSamples_{Cr}$.

3. The decoding process for prediction units in inter prediction mode as specified in clause 8.5.3 is invoked with the luma location (xCb, yCb), the luma location (xBl, yBl) set equal to (0, $nCbS_L$ >> 1), the size of the luma coding block $nCbS_L$, the width of the luma prediction block nPbW set equal to $nCbS_L$ >> 1, the height of the luma prediction block nPbH set equal to $nCbS_L$ >> 1 and a partition index partIdx set equal to 2 as inputs, and the outputs are the modified ($nCbS_L$)x($nCbS_L$) array $predSamples_L$ and, when ChromaArrayType is not equal to 0, the two modified ($nCbSw_C$)x($nCbSh_C$) arrays $predSamples_{Cb}$ and $predSamples_{Cr}$.

4. The decoding process for prediction units in inter prediction mode as specified in clause 8.5.3 is invoked with the luma location (xCb, yCb), the luma location (xBl, yBl) set equal to ($nCbS_L$ >> 1, $nCbS_L$ >> 1), the size of the luma coding block $nCbS_L$, the width of the luma prediction block nPbW set equal to $nCbS_L$ >> 1, the height of the luma prediction block nPbH set equal to $nCbS_L$ >> 1 and a partition index partIdx set equal to 3 as inputs, and the outputs are the modified ($nCbS_L$)x($nCbS_L$) array $predSamples_L$ and, when ChromaArrayType is not equal to 0, the two modified ($nCbSw_C$)x($nCbSh_C$) arrays $predSamples_{Cb}$ and $predSamples_{Cr}$.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 142-44.

8.5.3.3 Decoding process for inter prediction samples

8.5.3.3.1 General

Inputs to this process are:

- a luma location (xCb, yCb) specifying the top-left sample of the current luma coding block relative to the top-left luma sample of the current picture,
- a luma location (xBl, yBl) specifying the top-left sample of the current luma prediction block relative to the top-left sample of the current luma coding block,
- a variable nCbS specifying the size of the current luma coding block,
- two variables nPbW and nPbH specifying the width and the height of the luma prediction block,
- the luma motion vectors mvL0 and mvL1,
- when ChromaArrayType is not equal to 0, the chroma motion vectors mvCL0 and mvCL1,
- the reference indices refIdxL0 and refIdxL1,
- the prediction list utilization flags, predFlagL0, and predFlagL1.

Outputs of this process are:

- an $(nCbS_L)x(nCbS_L)$ array $predSamples_L$ of luma prediction samples, where $nCbS_L$ is derived as specified below,
- when ChromaArrayType is not equal to 0, an $(nCbSw_C)x(nCbSh_C)$ array $predSamples_{Cb}$ of chroma prediction samples for the component Cb, where $nCbSw_C$ and $nCbSh_C$ are derived as specified below,
- when ChromaArrayType is not equal to 0, an $(nCbSw_C)x(nCbSh_C)$ array $predSamples_{Cr}$ of chroma prediction samples for the component Cr, where $nCbSw_C$ and $nCbSh_C$ are derived as specified below.

The variable $nCbS_L$ is set equal to nCbS. When ChromaArrayType is not equal to 0, the variable $nCbSw_C$ is set equal to nCbS / SubWidthC and the variable $nCbSh_C$ is set equal to nCbS / SubHeightC.

Let $predSamplesL0_L$ and $predSamplesL1_L$ be (nPbW)x(nPbH) arrays of predicted luma sample values and, when ChromaArrayType is not equal to 0, $predSamplesL0_{Cb}$, $predSamplesL1_{Cb}$, $predSamplesL0_{Cr}$ and $predSamplesL1_{Cr}$ be (nPbW / SubWidthC)x(nPbH / SubHeightC) arrays of predicted chroma sample values.

For X being each of 0 and 1, when predFlagLX is equal to 1, the following applies:

- The reference picture consisting of an ordered two-dimensional array $refPicLX_L$ of luma samples and, when ChromaArrayType is not equal to 0, two ordered two-dimensional arrays $refPicLX_{Cb}$ and $refPicLX_{Cr}$ of chroma samples is derived by invoking the process specified in clause 8.5.3.3.2 with refIdxLX as input.
- The array $predSamplesLX_L$ and, when ChromaArrayType is not equal to 0, the arrays $predSamplesLX_{Cb}$ and $predSamplesLX_{Cr}$ are derived by invoking the fractional sample interpolation process specified in clause 8.5.3.3.3 with the luma locations (xCb, yCb) and (xBl, yBl), the luma prediction block width nPbW, the luma prediction block height nPbH, the motion vectors mvLX and, when ChromaArrayType is not equal to 0, mvCLX, and the reference arrays $refPicLX_L$, $refPicLX_{Cb}$, and $refPicLX_{Cr}$ as inputs.

ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 161-62.

8.5.3.3.4.2 Default weighted sample prediction process

Inputs to this process are:

- two variables nPbW and nPbH specifying the width and the height of the current prediction block,
- two (nPbW)x(nPbH) arrays predSamplesL0 and predSamplesL1,
- the prediction list utilization flags, predFlagL0, and predFlagL1,
- a bit depth of samples, bitDepth.

Output of this process is the (nPbW)x(nPbH) array pbSamples of prediction sample values.

Variables shift1, shift2, offset1 and offset2 are derived as follows:

- The variable shift1 is set equal to Max(2, 14 − bitDepth) and the variable shift2 is set equal to Max(3, 15 − bitDepth).
- The variable offset1 is set equal to 1 << (shift1 − 1).
- The variable offset2 is set equal to 1 << (shift2 − 1).

Depending on the values of predFlagL0 and predFlagL1, the prediction samples pbSamples[x][y] with x = 0..nPbW − 1 and y = 0..nPbH − 1 are derived as follows:

- If predFlagL0 is equal to 1 and predFlagL1 is equal to 0, the prediction sample values are derived as follows:

  pbSamples[x][y] = Clip3(0, (1 << bitDepth) − 1, (predSamplesL0[x][y] + offset1) >> shift1) (8-264)

  Otherwise, if predFlagL0 is equal to 0 and predFlagL1 is equal to 1, the prediction sample values are derived as follows:

  pbSamples[x][y] = Clip3(0, (1 << bitDepth) − 1, (predSamplesL1[x][y] + offset1) >> shift1) (8-265)

- Otherwise (predFlagL0 is equal to 1 and predFlagL1 is equal to 1), the prediction sample values are derived as follows:

  pbSamples[x][y] = Clip3(0, (1 << bitDepth) − 1,
  (predSamplesL0[x][y] + predSamplesL1[x][y] + offset2) >> shift2) (8-266)

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 168.

8.6.7 Picture construction process prior to in-loop filter process

Inputs to this process are:

- a location (xCurr, yCurr) specifying the top-left sample of the current block relative to the top-left sample of the current picture component,
- the variables nCurrSw and nCurrSh specifying the width and height, respectively, of the current block,
- a variable cIdx specifying the colour component of the current block,
- an (nCurrSw)x(nCurrSh) array predSamples specifying the predicted samples of the current block,
- an (nCurrSw)x(nCurrSh) array resSamples specifying the residual samples of the current block.

Depending on the value of the colour component cIdx, the following assignments are made:

- If cIdx is equal to 0, recSamples corresponds to the reconstructed picture sample array S_L and the function clipCidx1 corresponds to $Clip1_Y$.
- Otherwise, if cIdx is equal to 1, recSamples corresponds to the reconstructed chroma sample array S_{Cb} and the function clipCidx1 corresponds to $Clip1_C$.
- Otherwise (cIdx is equal to 2), recSamples corresponds to the reconstructed chroma sample array S_{Cr} and the function clipCidx1 corresponds to $Clip1_C$.

The (nCurrSw)x(nCurrSh) block of the reconstructed sample array recSamples at location (xCurr, yCurr) is derived as follows:

recSamples[xCurr + i][yCurr + j] = clipCidx1(predSamples[i][j] + resSamples[i][j]) (8-327)
with i = 0..nCurrSw − 1, j = 0..nCurrSh − 1

ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 180.

443. For another example, on information and belief, Amazon performs a method of decoding a block of pixels in a video signal in a manner that is covered by claim 19 of the ’267 Patent in the same or similar manner as described above when Amazon.com videos are decoded by an H.265-compliant decoder, and as performed, for example, during internal testing of video quality or transcoding.

444. On information and belief, Amazon performs internal testing of video quality on Amazon.com and transcoding of video on Amazon.com in a manner substantially similar to that which is performed on Amazon Prime Video. *See, e.g.*, https://www.amazon.science/blog/how-prime-video-uses-machine-learning-to-ensure-video-quality.

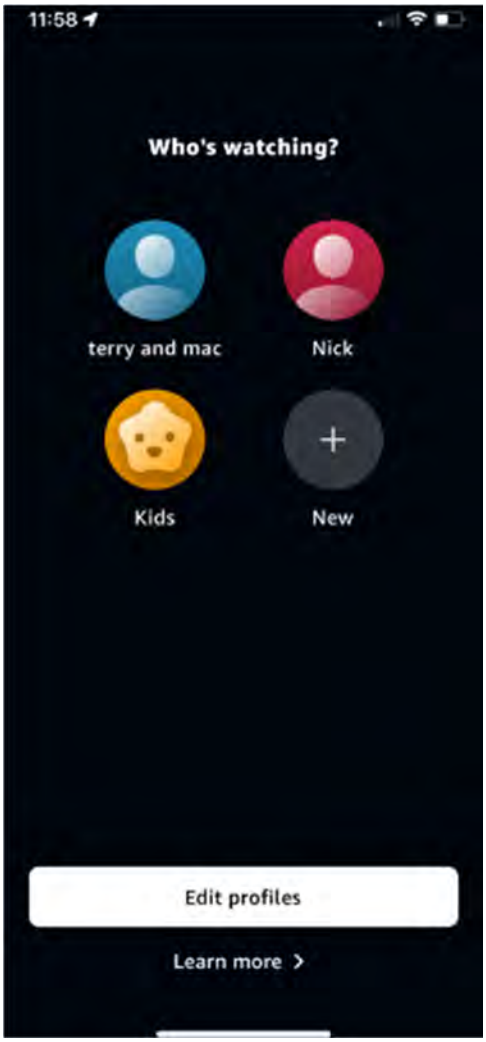
O. Amazon Makes, Imports, Uses, Sells, and/or Offers for Sale Products and Services that Infringe the ’137 Patent

445. The Accused Products infringe one or more claims of the ’137 Patent, including, for example, claim 1.

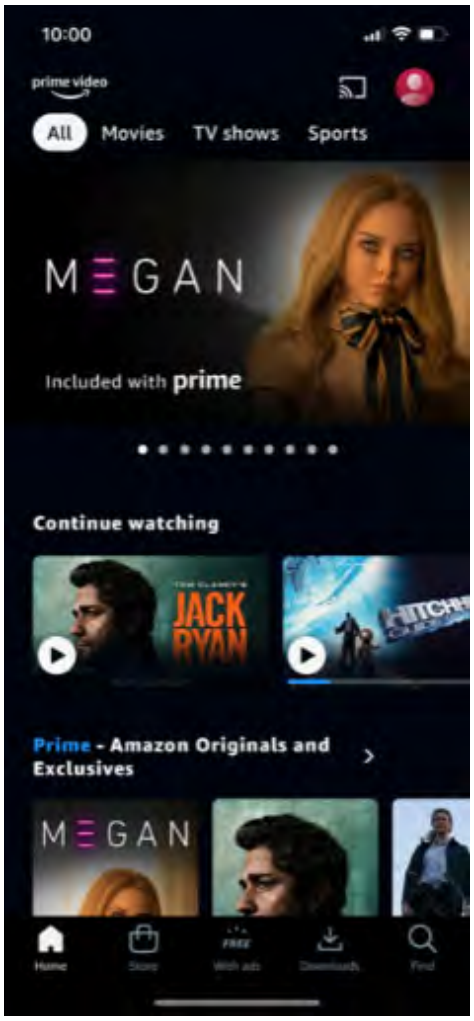
446. As just one example of infringement, on information and belief, Amazon performs a method in a manner that is covered by claim 1 of the ’137 Patent when users access and use Amazon Prime Video.

447. As another example of infringement, Amazon performs claim 1 of the ’137 Patent during internal testing of the Amazon Prime Video user interface. *See*, *e.g.*, https://www.vulture.com/2022/07/amazon-prime-video-new-design-home-screen.html (usability testing); https://www.forbes.com/sites/dbloom/2022/07/19/as-amazon-overhauls-prime-video-interface-keeping-it-simple-is-getting-more-complex/?sh=14d7505bf4bd (usability testing); https://www.geekwire.com/2023/amazon-starts-testing-new-mobile-app-layout-moving-search-box-to-bottom-of-the-screen/.

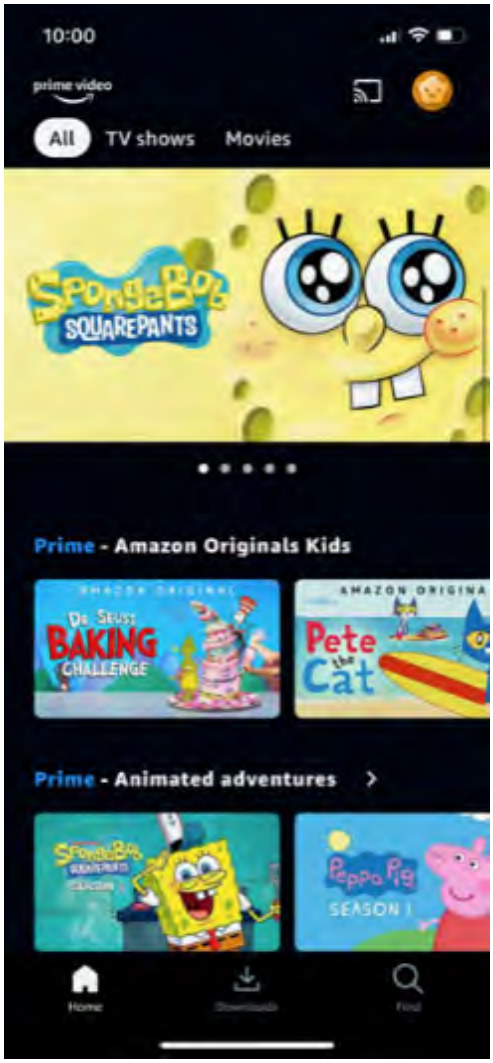
448. On information and belief, when accessing and searching for content on Amazon Prime Video, Amazon performs a method comprising facilitating a processing of and/or processing (1) data and/or (2) information and/or (3) at least one signal, the (1) data and/or (2) information and/or (3) at least one signal based, at least in part, on the following: a determination of a multi-dimensional query associated with at least one user device, wherein the multi-dimensional query specifies, at least in part, one or more personas, based, at least in part, on more than one person, associated with the at least one user device.



Screenshot of Amazon Prime Video application on the iPhone.



Screenshot of Amazon Prime Video application on the iPhone.



Screenshot of Amazon Prime Video application on the iPhone.

Digital Services & Content › Prime Video › Troubleshooting ›

Can I Watch Prime Video If I Travel Abroad?

Amazon Prime members are able to stream selected Amazon Originals titles while outside of their home country.

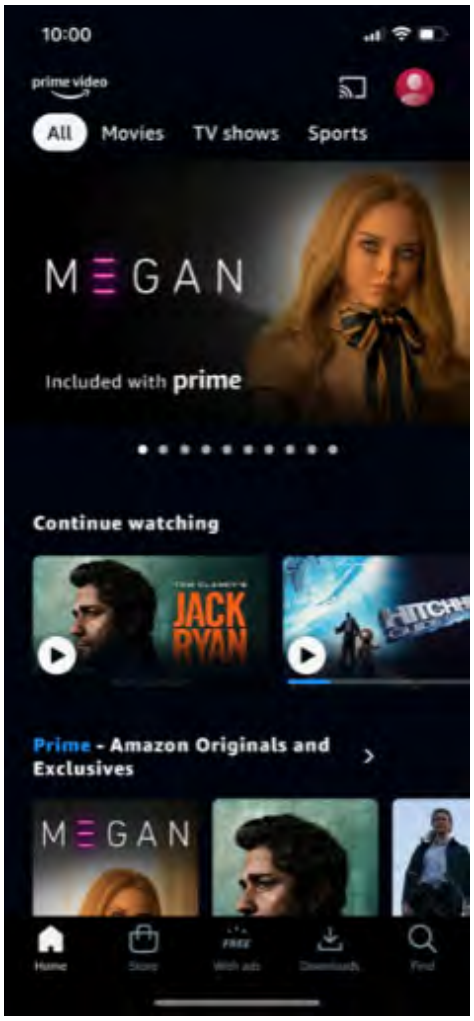
Outside of your home country, a reduced selection of Prime Video titles is available to stream. A selection marked "Watch While Abroad" shows the available titles.

If you have a compatible device, you're able to download titles before you travel in order to watch offline anywhere in the world.

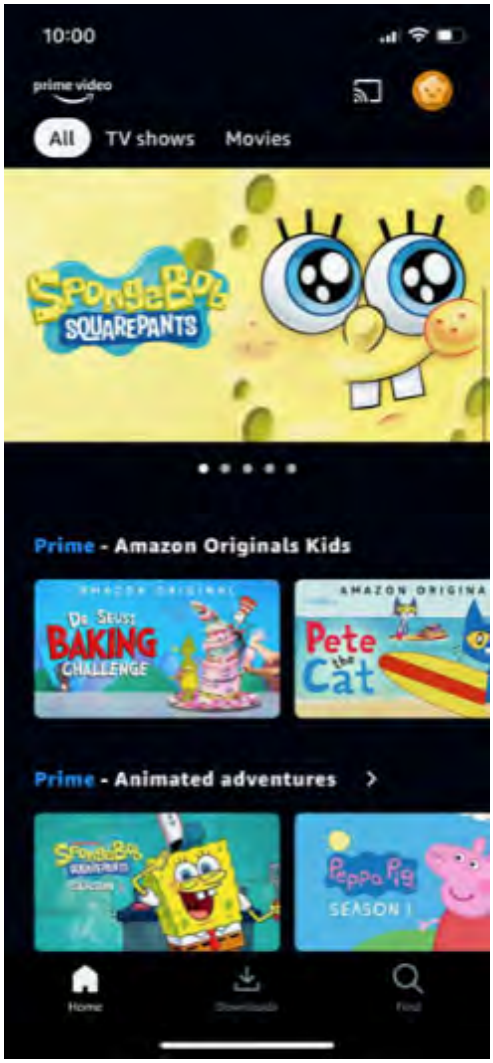
https://www.amazon.com/gp/help/customer/display.html?nodeId=GTBPYKSYKCXSPRKP#:~:text=Amazon%20Prime%20members%20are%20able,Abroad%22%20shows%20the%20available%20titles.

449. On information and belief, when accessing and searching for content on Amazon Prime Video, Amazon performs an execution of the multi-dimensional query on at least one context-sensitive database to generate one or more results.

Screenshot of Amazon Prime Video application on the iPhone.



Screenshot of Amazon Prime Video application on the iPhone.



Screenshot of Amazon Prime Video application on the iPhone.

Digital Services & Content › Prime Video › Troubleshooting ›

Can I Watch Prime Video If I Travel Abroad?

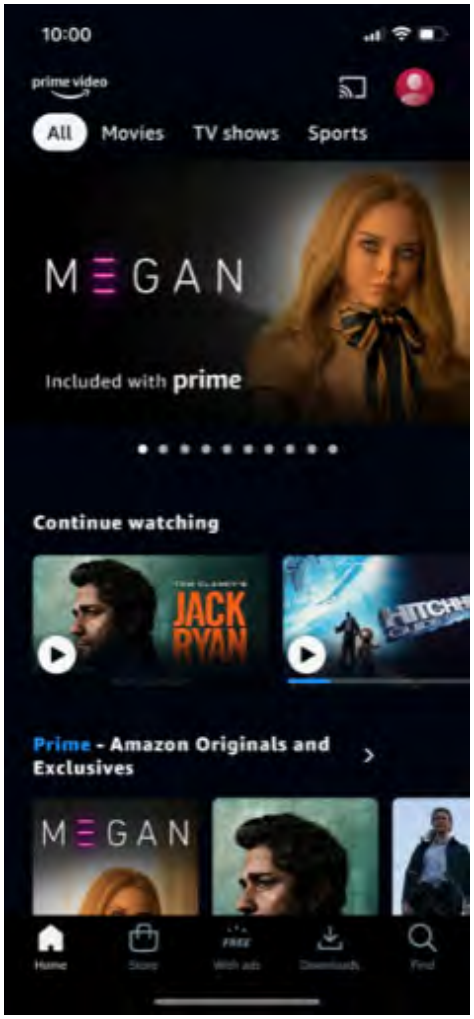
Amazon Prime members are able to stream selected Amazon Originals titles while outside of their home country.

Outside of your home country, a reduced selection of Prime Video titles is available to stream. A selection marked "Watch While Abroad" shows the available titles.

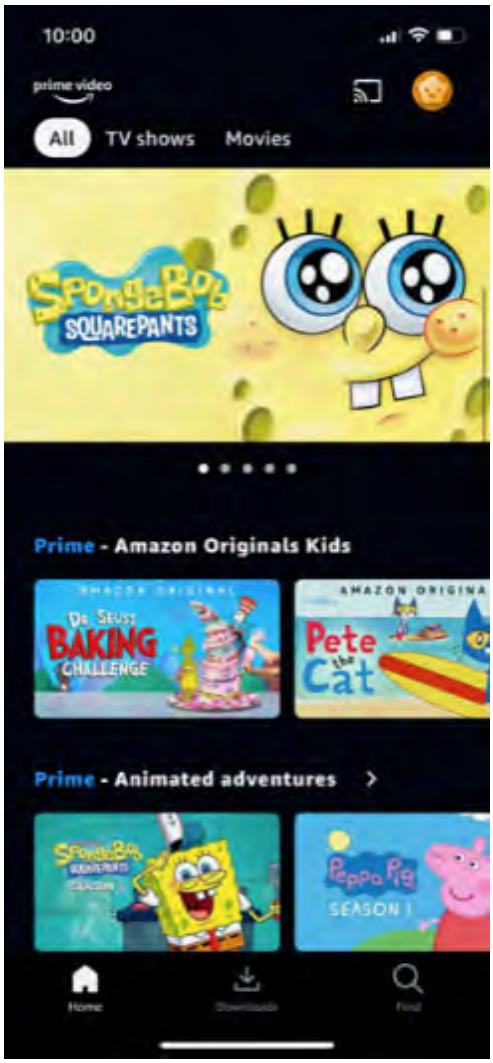
If you have a compatible device, you're able to download titles before you travel in order to watch offline anywhere in the world.

https://www.amazon.com/gp/help/customer/display.html?nodeId=GTBPYKSYKCXSPRKP#:~:text=Amazon%20Prime%20members%20are%20able,Abroad%22%20shows%20the%20available%20titles.

450. On information and belief, when accessing and searching for content on Amazon Prime Video, Amazon performs a determination of at least one ordering metric for the one or more results based, at least in part, on one or more user contextual attributes of the at least one user device.



Screenshot of Amazon Prime Video application on the iPhone.



Screenshot of Amazon Prime Video application on the iPhone.

COUNT I: PATENT INFRINGEMENT OF THE '808 PATENT

451. Nokia incorporates by reference the preceding paragraphs as though fully set forth herein.

452. Amazon infringes the '808 Patent by making, using, selling, offering for sale, and/or importing into the United States products and/or methods covered by one or more claims of the '808 Patent.

453. Amazon has been and is now infringing the claims of the '808 Patent, literally and/or under the Doctrine of Equivalents, in violation of 35 U.S.C. § 271, including by making,

using, testing, selling, consigning, importing into the United States, distributing within the United States, and/or exporting infringing products.

454. Amazon makes, uses, sells, offers for sale, and/or imports the Accused Products in this District and elsewhere in the United States, and thus directly infringes the ’808 Patent.

455. Amazon has had knowledge and notice of the ’808 Patent and its infringement thereof since at least April 2015, when Nokia notified Amazon that Amazon’s devices, products, and services, if made, used, sold, or offered for sale without a license, would infringe. Amazon has been involved in licensing discussions with Nokia regarding Nokia’s patent portfolios, which include the ’808 Patent. Amazon has also received actual notice of the ’808 Patent as of the date this lawsuit was filed and/or the date this Original Complaint was served upon Amazon.

456. Amazon has known of the existence of the ’808 Patent, and its acts of infringement have been willful and in disregard for the ’808 Patent, without any reasonable basis for believing that it had a right to engage in the infringing conduct. Amazon was aware of the ’808 Patent at least as a result of the licensing negotiations between the parties and continued to infringe the ’808 Patent.

457. Upon information and belief, Amazon derives revenue, directly and indirectly, from the activities relating to the Accused Products, including their importation, testing, manufacture, use, sale, and offer for sale.

458. Amazon’s infringement of the ’808 Patent has damaged and will continue to damage Nokia.

COUNT II: PATENT INFRINGEMENT OF THE ’321 PATENT

459. Nokia incorporates by reference the preceding paragraphs as though fully set forth herein.

460. Amazon infringes the ’321 Patent by making, using, selling, offering for sale, and/or importing into the United States products and/or methods covered by one or more claims of the ’321 Patent.

461. Amazon has been and is now infringing the claims of the ’321 Patent, literally and/or under the Doctrine of Equivalents, in violation of 35 U.S.C. § 271, including by making, using, testing, selling, consigning, importing into the United States, distributing within the United States, and/or exporting infringing products.

462. Amazon makes, uses, sells, offers for sale, and/or imports the Accused Products in this District and elsewhere in the United States, and thus directly infringes the ’321 Patent.

463. Amazon has had knowledge and notice of the ’321 Patent and its infringement thereof since at least April 2015, when Nokia notified Amazon that Amazon’s devices, products, and services, if made, used, sold, or offered for sale without a license, would infringe. Amazon has been involved in licensing discussions with Nokia regarding Nokia’s patent portfolios, which include the ’321 Patent. Amazon has also received actual notice of the ’321 Patent as of the date this lawsuit was filed and/or the date this Original Complaint was served upon Amazon.

464. Amazon has known of the existence of the ’321 Patent, and its acts of infringement have been willful and in disregard for the ’321 Patent, without any reasonable basis for believing that it had a right to engage in the infringing conduct. Amazon was aware of the ’321 Patent at least as a result of the licensing negotiations between the parties and continued to infringe the ’321 Patent.

465. Upon information and belief, Amazon derives revenue, directly and indirectly, from the activities relating to the Accused Products, including their importation, testing, manufacture, use, sale, and offer for sale.

466. Amazon's infringement of the '321 Patent has damaged and will continue to damage Nokia.

COUNT III: PATENT INFRINGEMENT OF THE '818 PATENT

467. Nokia incorporates by reference the preceding paragraphs as though fully set forth herein.

468. Amazon infringes the '818 Patent by making, using, selling, offering for sale, and/or importing into the United States products and/or methods covered by one or more claims of the '818 Patent.

469. Amazon has been and is now infringing the claims of the '818 Patent, literally and/or under the Doctrine of Equivalents, in violation of 35 U.S.C. § 271, including by making, using, testing, selling, consigning, importing into the United States, distributing within the United States, and/or exporting infringing products.

470. Amazon makes, uses, sells, offers for sale, and/or imports the Accused Products in this District and elsewhere in the United States, and thus directly infringes the '818 Patent.

471. Amazon has had knowledge and notice of the '818 Patent and its infringement thereof since at least April 2015, when Nokia notified Amazon that Amazon's devices, products, and services, if made, used, sold, or offered for sale without a license, would infringe. Amazon has been involved in licensing discussions with Nokia regarding Nokia's patent portfolios, which include the '818 Patent. Amazon has also received actual notice of the '818 Patent as of the date this lawsuit was filed and/or the date this Original Complaint was served upon Amazon.

472. Amazon has known of the existence of the '818 Patent, and its acts of infringement have been willful and in disregard for the '818 Patent, without any reasonable basis for believing that it had a right to engage in the infringing conduct. Amazon was aware of

the ’818 Patent at least as a result of the licensing negotiations between the parties and continued to infringe the ’818 Patent.

473. Upon information and belief, Amazon derives revenue, directly and indirectly, from the activities relating to the Accused Products, including their importation, testing, manufacture, use, sale, and offer for sale.

474. Amazon’s infringement of the ’818 Patent has damaged and will continue to damage Nokia.

COUNT IV: PATENT INFRINGEMENT OF THE ’469 PATENT

475. Nokia incorporates by reference the preceding paragraphs as though fully set forth herein.

476. Amazon infringes the ’469 Patent by making, using, selling, offering for sale, and/or importing into the United States products and/or methods covered by one or more claims of the ’469 Patent.

477. Amazon has been and is now infringing the claims of the ’469 Patent, literally and/or under the Doctrine of Equivalents, in violation of 35 U.S.C. § 271, including by making, using, testing, selling, consigning, importing into the United States, distributing within the United States, and/or exporting infringing products.

478. Amazon makes, uses, sells, offers for sale, and/or imports the Accused Products in this District and elsewhere in the United States, and thus directly infringes the ’469 Patent.

479. Amazon has had knowledge and notice of the ’469 Patent and its infringement thereof since at least October 25, 2023, when Nokia notified Amazon that Amazon’s devices, products, and services, if made, used, sold, or offered for sale without a license, would infringe. Amazon has been involved in licensing discussions with Nokia regarding Nokia’s patent portfolios, which include the ’469 Patent. Amazon has also received actual notice of the ’469

Patent as of the date this lawsuit was filed and/or the date this Original Complaint was served upon Amazon.

480. Amazon has known of the existence of the ’469 Patent, and its acts of infringement have been willful and in disregard for the ’469 Patent, without any reasonable basis for believing that it had a right to engage in the infringing conduct. Amazon was aware of the ’469 Patent at least as a result of the licensing negotiations between the parties and continued to infringe the ’469 Patent.

481. Upon information and belief, Amazon derives revenue, directly and indirectly, from the activities relating to the Accused Products, including their importation, testing, manufacture, use, sale, and offer for sale.

482. Amazon’s infringement of the ’469 Patent has damaged and will continue to damage Nokia.

COUNT V: PATENT INFRINGEMENT OF THE ’599 PATENT

483. Nokia incorporates by reference the preceding paragraphs as though fully set forth herein.

484. Amazon infringes the ’599 Patent by making, using, selling, offering for sale, and/or importing into the United States products and/or methods covered by one or more claims of the ’599 Patent.

485. Amazon has been and is now infringing the claims of the ’599 Patent, literally and/or under the Doctrine of Equivalents, in violation of 35 U.S.C. § 271, including by making, using, testing, selling, consigning, importing into the United States, distributing within the United States, and/or exporting infringing products.

486. Amazon makes, uses, sells, offers for sale, and/or imports the Accused Products in this District and elsewhere in the United States, and thus directly infringes the ’599 Patent.

487. Amazon has had knowledge and notice of the ’599 Patent and its infringement thereof since at least October 25, 2023, when Nokia notified Amazon that Amazon’s devices, products, and services, if made, used, sold, or offered for sale without a license, would infringe. Amazon has been involved in licensing discussions with Nokia regarding Nokia’s patent portfolios, which include the ’599 Patent. Amazon has also received actual notice of the ’599 Patent as of the date this lawsuit was filed and/or the date this Original Complaint was served upon Amazon.

488. Amazon has known of the existence of the ’599 Patent, and its acts of infringement have been willful and in disregard for the ’599 Patent, without any reasonable basis for believing that it had a right to engage in the infringing conduct. Amazon was aware of the ’599 Patent at least as a result of the licensing negotiations between the parties and continued to infringe the ’599 Patent.

489. Upon information and belief, Amazon derives revenue, directly and indirectly, from the activities relating to the Accused Products, including their importation, testing, manufacture, use, sale, and offer for sale.

490. Amazon’s infringement of the ’599 Patent has damaged and will continue to damage Nokia.

COUNT VI: PATENT INFRINGEMENT OF THE ’273 PATENT

491. Nokia incorporates by reference the preceding paragraphs as though fully set forth herein.

492. Amazon infringes the ’273 Patent by making, using, selling, offering for sale, and/or importing into the United States products and/or methods covered by one or more claims of the ’273 Patent.

493. Amazon has been and is now infringing the claims of the ’273 Patent, literally and/or under the Doctrine of Equivalents, in violation of 35 U.S.C. § 271, including by making, using, testing, selling, consigning, importing into the United States, distributing within the United States, and/or exporting infringing products.

494. Amazon makes, uses, sells, offers for sale, and/or imports the Accused Products in this District and elsewhere in the United States, and thus directly infringes the ’273 Patent.

495. Amazon has had knowledge and notice of the ’273 Patent and its infringement thereof since at least October 25, 2023, when Nokia notified Amazon that Amazon’s devices, products, and services, if made, used, sold, or offered for sale without a license, would infringe. Amazon has been involved in licensing discussions with Nokia regarding Nokia’s patent portfolios, which include the ’273 Patent. Amazon has also received actual notice of the ’273 Patent as of the date this lawsuit was filed and/or the date this Original Complaint was served upon Amazon.

496. Amazon has known of the existence of the ’273 Patent, and its acts of infringement have been willful and in disregard for the ’273 Patent, without any reasonable basis for believing that it had a right to engage in the infringing conduct. Amazon was aware of the ’273 Patent at least as a result of the licensing negotiations between the parties and continued to infringe the ’273 Patent.

497. Upon information and belief, Amazon derives revenue, directly and indirectly, from the activities relating to the Accused Products, including their importation, testing, manufacture, use, sale, and offer for sale.

498. Amazon’s infringement of the ’273 Patent has damaged and will continue to damage Nokia.

COUNT VII: PATENT INFRINGEMENT OF THE ’701 PATENT

499. Nokia incorporates by reference the preceding paragraphs as though fully set forth herein.

500. Amazon infringes the ’701 Patent by making, using, selling, offering for sale, and/or importing into the United States products and/or methods covered by one or more claims of the ’701 Patent.

501. Amazon has been and is now infringing the claims of the ’701 Patent, literally and/or under the Doctrine of Equivalents, in violation of 35 U.S.C. § 271, including by making, using, testing, selling, consigning, importing into the United States, distributing within the United States, and/or exporting infringing products.

502. Amazon makes, uses, sells, offers for sale, and/or imports the Accused Products in this District and elsewhere in the United States, and thus directly infringes the ’701 Patent.

503. Amazon has had knowledge and notice of the ’701 Patent and its infringement thereof since at least October 25, 2023, when Nokia notified Amazon that Amazon’s devices, products, and services, if made, used, sold, or offered for sale without a license, would infringe. Amazon has been involved in licensing discussions with Nokia regarding Nokia’s patent portfolios, which include the ’701 Patent. Amazon has also received actual notice of the ’701 Patent as of the date this lawsuit was filed and/or the date this Original Complaint was served upon Amazon.

504. Amazon has known of the existence of the ’701 Patent, and its acts of infringement have been willful and in disregard for the ’701 Patent, without any reasonable basis for believing that it had a right to engage in the infringing conduct. Amazon was aware of the ’701 Patent at least as a result of the licensing negotiations between the parties and continued to infringe the ’701 Patent.

505. Upon information and belief, Amazon derives revenue, directly and indirectly, from the activities relating to the Accused Products, including their importation, testing, manufacture, use, sale, and offer for sale.

506. Amazon's infringement of the '701 Patent has damaged and will continue to damage Nokia.

COUNT VIII: PATENT INFRINGEMENT OF THE '891 PATENT

507. Nokia incorporates by reference the preceding paragraphs as though fully set forth herein.

508. Amazon infringes the '891 Patent by making, using, selling, offering for sale, and/or importing into the United States products and/or methods covered by one or more claims of the '891 Patent.

509. Amazon has been and is now infringing the claims of the '891 Patent, literally and/or under the Doctrine of Equivalents, in violation of 35 U.S.C. § 271, including by making, using, testing, selling, consigning, importing into the United States, distributing within the United States, and/or exporting infringing products.

510. Amazon makes, uses, sells, offers for sale, and/or imports the Accused Products in this District and elsewhere in the United States, and thus directly infringes the '891 Patent.

511. Amazon has had knowledge and notice of the '891 Patent and its infringement thereof since at least October 25, 2023, when Nokia notified Amazon that Amazon's devices, products, and services, if made, used, sold, or offered for sale without a license, would infringe. Amazon has been involved in licensing discussions with Nokia regarding Nokia's patent portfolios, which include the '891 Patent. Amazon has also received actual notice of the '891 Patent as of the date this lawsuit was filed and/or the date this Original Complaint was served upon Amazon.

512. Amazon has known of the existence of the ’891 Patent, and its acts of infringement have been willful and in disregard for the ’891 Patent, without any reasonable basis for believing that it had a right to engage in the infringing conduct. Amazon was aware of the ’891 Patent at least as a result of the licensing negotiations between the parties and continued to infringe the ’891 Patent.

513. Upon information and belief, Amazon derives revenue, directly and indirectly, from the activities relating to the Accused Products, including their importation, testing, manufacture, use, sale, and offer for sale.

514. Amazon’s infringement of the ’891 Patent has damaged and will continue to damage Nokia.

COUNT IX: PATENT INFRINGEMENT OF THE ’005 PATENT

515. Nokia incorporates by reference the preceding paragraphs as though fully set forth herein.

516. Amazon infringes the ’005 Patent by making, using, selling, offering for sale, and/or importing into the United States products and/or methods covered by one or more claims of the ’005 Patent.

517. Amazon has been and is now infringing the claims of the ’005 Patent, literally and/or under the Doctrine of Equivalents, in violation of 35 U.S.C. § 271, including by making, using, testing, selling, consigning, importing into the United States, distributing within the United States, and/or exporting infringing products.

518. Amazon makes, uses, sells, offers for sale, and/or imports the Accused Products in this District and elsewhere in the United States, and thus directly infringes the ’005 Patent.

519. Amazon has had knowledge and notice of the ’005 Patent and its infringement thereof since at least October 25, 2023, when Nokia notified Amazon that Amazon’s devices,

products, and services, if made, used, sold, or offered for sale without a license, would infringe. Amazon has been involved in licensing discussions with Nokia regarding Nokia's patent portfolios, which include the '005 Patent. Amazon has also received actual notice of the '005 Patent as of the date this lawsuit was filed and/or the date this Original Complaint was served upon Amazon.

520. Amazon has known of the existence of the '005 Patent, and its acts of infringement have been willful and in disregard for the '005 Patent, without any reasonable basis for believing that it had a right to engage in the infringing conduct. Amazon was aware of the '005 Patent at least as a result of the licensing negotiations between the parties and continued to infringe the '005 Patent.

521. Upon information and belief, Amazon derives revenue, directly and indirectly, from the activities relating to the Accused Products, including their importation, testing, manufacture, use, sale, and offer for sale.

522. Amazon's infringement of the '005 Patent has damaged and will continue to damage Nokia.

COUNT X: PATENT INFRINGEMENT OF THE '764 PATENT

523. Nokia incorporates by reference the preceding paragraphs as though fully set forth herein.

524. Amazon infringes the '764 Patent by making, using, selling, offering for sale, and/or importing into the United States products and/or methods covered by one or more claims of the '764 Patent.

525. Amazon has been and is now infringing the claims of the '764 Patent, literally and/or under the Doctrine of Equivalents, in violation of 35 U.S.C. § 271, including by making,

using, testing, selling, consigning, importing into the United States, distributing within the United States, and/or exporting infringing products.

526. Amazon makes, uses, sells, offers for sale, and/or imports the Accused Products in this District and elsewhere in the United States, and thus directly infringes the ’764 Patent.

527. Amazon has had knowledge and notice of the ’764 Patent and its infringement thereof since at least October 25, 2023, when Nokia notified Amazon that Amazon’s devices, products, and services, if made, used, sold, or offered for sale without a license, would infringe. Amazon has been involved in licensing discussions with Nokia regarding Nokia’s patent portfolios, which include the ’764 Patent. Amazon has also received actual notice of the ’764 Patent as of the date this lawsuit was filed and/or the date this Original Complaint was served upon Amazon.

528. Amazon has known of the existence of the ’764 Patent, and its acts of infringement have been willful and in disregard for the ’764 Patent, without any reasonable basis for believing that it had a right to engage in the infringing conduct. Amazon was aware of the ’764 Patent at least as a result of the licensing negotiations between the parties and continued to infringe the ’764 Patent.

529. Upon information and belief, Amazon derives revenue, directly and indirectly, from the activities relating to the Accused Products, including their importation, testing, manufacture, use, sale, and offer for sale.

530. Amazon’s infringement of the ’764 Patent has damaged and will continue to damage Nokia.

COUNT XI: PATENT INFRINGEMENT OF THE ’148 PATENT

531. Nokia incorporates by reference the preceding paragraphs as though fully set forth herein.

532. Amazon infringes the ’148 Patent by making, using, selling, offering for sale, and/or importing into the United States products and/or methods covered by one or more claims of the ’148 Patent.

533. Amazon has been and is now infringing the claims of the ’148 Patent, literally and/or under the Doctrine of Equivalents, in violation of 35 U.S.C. § 271, including by making, using, testing, selling, consigning, importing into the United States, distributing within the United States, and/or exporting infringing products.

534. Amazon makes, uses, sells, offers for sale, and/or imports the Accused Products in this District and elsewhere in the United States, and thus directly infringes the ’148 Patent.

535. Amazon has had knowledge and notice of the ’148 Patent and its infringement thereof since at least October 25, 2023, when Nokia notified Amazon that Amazon’s devices, products, and services, if made, used, sold, or offered for sale without a license, would infringe. Amazon has been involved in licensing discussions with Nokia regarding Nokia’s patent portfolios, which include the ’148 Patent. Amazon has also received actual notice of the ’148 Patent as of the date this lawsuit was filed and/or the date this Original Complaint was served upon Amazon.

536. Amazon has known of the existence of the ’148 Patent, and its acts of infringement have been willful and in disregard for the ’148 Patent, without any reasonable basis for believing that it had a right to engage in the infringing conduct. Amazon was aware of the ’148 Patent at least as a result of the licensing negotiations between the parties and continued to infringe the ’148 Patent.

537. Upon information and belief, Amazon derives revenue, directly and indirectly, from the activities relating to the Accused Products, including their importation, testing, manufacture, use, sale, and offer for sale.

538. Amazon's infringement of the '148 Patent has damaged and will continue to damage Nokia.

COUNT XII: PATENT INFRINGEMENT OF THE '991 PATENT

539. Nokia incorporates by reference the preceding paragraphs as though fully set forth herein.

540. Amazon infringes the '991 Patent by making, using, selling, offering for sale, and/or importing into the United States products and/or methods covered by one or more claims of the '991 Patent.

541. Amazon has been and is now infringing the claims of the '991 Patent, literally and/or under the Doctrine of Equivalents, in violation of 35 U.S.C. § 271, including by making, using, testing, selling, consigning, importing into the United States, distributing within the United States, and/or exporting infringing products.

542. Amazon makes, uses, sells, offers for sale, and/or imports the Accused Products in this District and elsewhere in the United States, and thus directly infringes the '991 Patent.

543. Amazon has had knowledge and notice of the '991 Patent and its infringement thereof since at least October 25, 2023, when Nokia notified Amazon that Amazon's devices, products, and services, if made, used, sold, or offered for sale without a license, would infringe. Amazon has been involved in licensing discussions with Nokia regarding Nokia's patent portfolios, which include the '991 Patent. Amazon has also received actual notice of the '991 Patent as of the date this lawsuit was filed and/or the date this Original Complaint was served upon Amazon.

544. Amazon has known of the existence of the ’991 Patent, and its acts of infringement have been willful and in disregard for the ’991 Patent, without any reasonable basis for believing that it had a right to engage in the infringing conduct. Amazon was aware of the ’991 Patent at least as a result of the licensing negotiations between the parties and continued to infringe the ’991 Patent.

545. Upon information and belief, Amazon derives revenue, directly and indirectly, from the activities relating to the Accused Products, including their importation, testing, manufacture, use, sale, and offer for sale.

546. Amazon’s infringement of the ’991 Patent has damaged and will continue to damage Nokia.

COUNT XIII: PATENT INFRINGEMENT OF THE ’833 PATENT

547. Nokia incorporates by reference the preceding paragraphs as though fully set forth herein.

548. Amazon infringes the ’833 Patent by making, using, selling, offering for sale, and/or importing into the United States products and/or methods covered by one or more claims of the ’833 Patent.

549. Amazon has been and is now infringing the claims of the ’833 Patent, literally and/or under the Doctrine of Equivalents, in violation of 35 U.S.C. § 271, including by making, using, testing, selling, consigning, importing into the United States, distributing within the United States, and/or exporting infringing products.

550. Amazon makes, uses, sells, offers for sale, and/or imports the Accused Products in this District and elsewhere in the United States, and thus directly infringes the ’833 Patent.

551. Amazon has had knowledge and notice of the ’833 Patent and its infringement thereof since at least October 25, 2023, when Nokia notified Amazon that Amazon’s devices,

products, and services, if made, used, sold, or offered for sale without a license, would infringe. Amazon has been involved in licensing discussions with Nokia regarding Nokia's patent portfolios, which include the '833 Patent. Amazon has also received actual notice of the '833 Patent as of the date this lawsuit was filed and/or the date this Original Complaint was served upon Amazon.

552. Amazon has known of the existence of the '833 Patent, and its acts of infringement have been willful and in disregard for the '833 Patent, without any reasonable basis for believing that it had a right to engage in the infringing conduct. Amazon was aware of the '833 Patent at least as a result of the licensing negotiations between the parties and continued to infringe the '833 Patent.

553. Upon information and belief, Amazon derives revenue, directly and indirectly, from the activities relating to the Accused Products, including their importation, testing, manufacture, use, sale, and offer for sale.

554. Amazon's infringement of the '833 Patent has damaged and will continue to damage Nokia.

COUNT XIV: PATENT INFRINGEMENT OF THE '267 PATENT

555. Nokia incorporates by reference the preceding paragraphs as though fully set forth herein.

556. Amazon infringes the '267 Patent by making, using, selling, offering for sale, and/or importing into the United States products and/or methods covered by one or more claims of the '267 Patent.

557. Amazon has been and is now infringing the claims of the '267 Patent, literally and/or under the Doctrine of Equivalents, in violation of 35 U.S.C. § 271, including by making,

using, testing, selling, consigning, importing into the United States, distributing within the United States, and/or exporting infringing products.

558. Amazon makes, uses, sells, offers for sale, and/or imports the Accused Products in this District and elsewhere in the United States, and thus directly infringes the ’267 Patent.

559. Amazon has had knowledge and notice of the ’267 Patent and its infringement thereof since at least as of the date this lawsuit was filed and/or the date this Original Complaint was served upon Amazon.

560. Amazon’s acts of infringement are willful and in disregard for the ’267 Patent, without any reasonable basis for believing that it has a right to engage in infringing conduct. Amazon continues to infringe the ’267 Patent.

561. Upon information and belief, Amazon derives revenue, directly and indirectly, from the activities relating to the Accused Products, including their importation, testing, manufacture, use, sale, and offer for sale.

562. Amazon’s infringement of the ’267 Patent has damaged and will continue to damage Nokia.

COUNT XV: PATENT INFRINGEMENT OF THE ’137 PATENT

563. Nokia incorporates by reference the preceding paragraphs as though fully set forth herein.

564. Amazon infringes the ’137 Patent by making, using, selling, offering for sale, and/or importing into the United States products and/or methods covered by one or more claims of the ’137 Patent.

565. Amazon has been and is now infringing the claims of the ’137 Patent, literally and/or under the Doctrine of Equivalents, in violation of 35 U.S.C. § 271, including by making,

using, testing, selling, consigning, importing into the United States, distributing within the United States, and/or exporting infringing products.

566. Amazon makes, uses, sells, offers for sale, and/or imports the Accused Products in this District and elsewhere in the United States, and thus directly infringes the ’137 Patent.

567. Amazon has had knowledge and notice of the ’137 Patent and its infringement thereof since at least as of the date this lawsuit was filed and/or the date this Original Complaint was served upon Amazon.

568. Amazon’s acts of infringement are willful and in disregard for the ’137 Patent, without any reasonable basis for believing that it has a right to engage in the infringing conduct. Amazon continues to infringe the ’137 Patent.

569. Upon information and belief, Amazon derives revenue, directly and indirectly, from the activities relating to the Accused Products, including their importation, testing, manufacture, use, sale, and offer for sale.

570. Amazon’s infringement of the ’137 Patent has damaged and will continue to damage Nokia.

ATTORNEYS’ FEES

571. Nokia is entitled to recover reasonable and necessary attorneys’ fees under applicable law.

PRAYER FOR RELIEF

WHEREFORE, Nokia respectfully requests that this Court enter judgment in its favor as follows and afford Nokia the following relief:

I. adjudge and declare that Amazon infringes claims of the Asserted Patents;

II. adjudge and declare that Amazon’s infringement of claims of the Asserted Patents was willful, and that Amazon’s continued infringement of the claims is willful;

III. award Nokia actual damages;

IV. award Nokia enhanced damages pursuant to 35 U.S.C. § 284;

V. award Nokia pre-judgment and post-judgment interest to the full extent allowed under the law, as well as its costs;

VI. as to claims that are not essential to the H.264 or H.265 Standards, enter an injunction precluding Amazon and any entities in active concert with it from future acts of infringement;

VII. adjudge and declare that this is an exceptional case and award Nokia its reasonable attorneys' fees pursuant to 35 U.S.C. § 285;

VIII. order an accounting of damages for acts of infringement;

IX. award Nokia its costs of suit; and

X. award such other equitable relief which may be requested and to which Nokia is entitled.

Dated: October 31, 2023

Respectfully submitted,

FARNAN LLP

/s/ Michael J. Farnan

Brian E. Farnan (Bar No. 4089)
Michael J. Farnan (Bar No. 5165)
919 N. Market St. 12th Floor
Wilmington DE 19801
Tel.: (302) 777-0300
bfarnan@farnanlaw.com
mfarnan@farnanlaw.com

MCKOOL SMITH, P.C.

Warren H. Lipschitz*
Alexandra F. Easley*
300 Crescent Ct. Ste. 1200
Dallas, TX 75224
Tel.: (214) 978-4000

wlipschitz@mckoolsmith.com
aeasley@mckoolsmith.com

Joshua Budwin*
R. Mitch Verboncoeur*
303 Colorado St Suite 2100
Austin, TX 78701
Tel.: (512) 692-8700
jbudwin@mckoolsmith.com
mverboncoeur@mckoolsmith.com

Josh Newcomer*
600 Travis St., Suite 7000
Houston, Texas 77002
Tel.: (713) 485-7300
jnewcomer@mckoolsmith.com

Kevin Burgess*
104 East Houston St., Suite 300
Marshall, Texas 75670
Tel.: (903) 923-9000
kburgess@mckoolsmith.com

ALSTON & BIRD LLP

Theodore Stevenson, III*
2200 Ross Ave. #2300
Dallas, TX 75201
Tel.: (214) 922-3400
ted.stevenson@alston.com

John D. Haynes*
Nicholas T. Tsui*
Shawn Gannon*
1201 West Peachtree Street
Atlanta, GA 30309
Tel.: (404) 881-7000
john.haynes@alston.com
nick.tsui@alston.com
shawn.gannon@alston.com

***Local Civil Rule 83.5 motions for pro hac vice admission forthcoming**

Counsel for Plaintiff
Nokia Technologies Oy