

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

INTEL CORPORATION,
Petitioner

v.

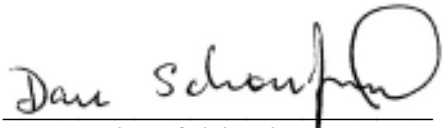
TELEFONAKTIEBOLAGET LM ERICSSON,
Patent Owner.

IPR2024-01009
U.S. Patent No. 10,142,659

DECLARATION OF DAN SCHONFELD, PH.D.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true, and further, that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code.

Date: 06/14/2024

By: 
Dan Schonfeld, Ph.D.

INTEL-1003
10,142,659

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	QUALIFICATIONS	2
III.	MATERIALS CONSIDERED	9
IV.	LEGAL STANDARDS	9
V.	TECHNOLOGY BACKGROUND.....	15
A.	Video Encoding and Decoding	15
B.	Deblocking Filters	20
1.	Pixel Values	22
2.	Filters	28
3.	Deblocking Filter Development.....	34
a.	H.263.....	35
b.	H.264.....	38
c.	Draft H.265 (B205).....	46
VI.	'659 PATENT	48
A.	Specification.....	49
B.	Prosecution History	59
VII.	PRIOR ART.....	61
A.	Fu (Ex-1021 (original); Ex-1022 (translation))	61
B.	List D/ Applicant Admitted Prior Art (Ex-1024).....	68
C.	H.241 (Ex-1017)	74
D.	B205 / Applicant Admitted Prior Art (Ex-1018)	77
E.	B118 (Ex-1023).....	78
F.	Bjontegaard (and Bjontegaard's Code) (Ex-1009; Exs-1010-1015).....	82
G.	Gonzalez (Ex-1020)	95
VIII.	LEVEL OF A PERSON OF ORDINARY SKILL	99

TABLE OF CONTENTS (continued)

	Page
IX. CLAIM CONSTRUCTION	100
X. GROUNDS	101
A. Ground 1: Obviousness over Fu in view of a POSITA's Understanding of the Art, Evidenced by List, Bjontegaard, B205, and B118.....	101
1. Claim 1	101
a. Element 1[pre]: A method of reducing blocking artifacts at a block boundary between a block of multiple pixels and a neighboring block of multiple pixels in a video frame, said method comprises:	101
b. Element 1[a]: calculating a first offset to be equal to $(9 \times (q_0 - p_0) - 3 \times (q_1 - p_1) + 8) \gg 4$, wherein $\gg$ denotes a right shift operation, p_0 denotes a pixel value of a pixel closest to said block boundary in a line of pixels in said block, wherein said line of pixels is perpendicular to said block boundary, p_1 denotes a pixel value of a pixel next closest to said block boundary in said line of pixels, q_0 denotes a pixel value of a pixel closest to said block boundary in a corresponding line of pixels in said neighboring block, wherein said corresponding line of pixels is perpendicular to said block boundary and q_1 denotes a pixel value of a pixel next closest to said block boundary in said corresponding line of pixels;.....	105
c. Element 1[b]: modifying said pixel value p_0 of said pixel closest to said block boundary in said line of pixels by adding said first offset to said pixel value p_0 of said pixel closest to said block boundary in said line of pixels to form a modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels; and	130
d. Element 1[c]: modifying said pixel value q_0 of said pixel closest to said block boundary in said corresponding line of pixels by subtracting said first	

TABLE OF CONTENTS (continued)

	Page
offset from said pixel value q_0 of said pixel closest to said block boundary in said corresponding line of pixels to form a modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels.....	132
2. Claim 2.....	133
a. Element 2[a]: The method according to claim 1, further comprising clipping said first offset to be within an interval of $-t_c$ and t_c by setting said first offset to be equal to $-t_c$ if said first offset is smaller than $-t_c$ and by setting said first offset to be equal to t_c if said first offset is larger than t_c ,.....	134
b. Element 2[b]: wherein t_c is a threshold value that is dependent on a quantization parameter value assigned to said block.	135
3. Claim 3.....	139
a. Element 3[a]: The method according to claim 1, further comprising: clipping said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels to be within an interval of zero and a defined maximum value by setting said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels to be equal to zero if said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels is smaller than zero and by setting said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels to be equal to said defined maximum value if said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels is larger than said defined maximum value; and	139
b. Element 3[b]: clipping said modified pixel value q_0' of said pixel closest to said block boundary in said	

TABLE OF CONTENTS (continued)

	Page
corresponding line of pixels to be within an interval of zero and said defined maximum value by setting said modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels to be equal to zero if said modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels is smaller than zero and by setting said modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels to be equal to said defined maximum if said modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels is larger than said defined maximum value.....	142
4. Claim 4.....	144
a. Element 4[a]: The method according to claim 1, further comprising: calculating a second offset to be based on $(p_2 + p_0[-]2 \times p_1 + 2 \times \Delta)/4$, wherein p_2 denotes a pixel value of a pixel second next closest to said block boundary in said line of pixels and Δ denotes said first offset;	144
b. Element 4[b]: calculating a third offset to be based on $(q_2 + q_0 - 2 \times q_1 - 2 \times \Delta)/4$, wherein q_2 denotes a pixel value of a pixel second next closest to said block boundary in said corresponding line of pixels;	158
c. Element 4[c]: modifying said pixel value p_1 of said pixel next closest to said block boundary in said line of pixels by adding said second offset to said pixel value p_1 of said pixel next closest to said block boundary in said line of pixels to form a modified pixel value p_1' of said pixel next closest to said block boundary in said line of pixels; and.....	164
d. Element 4[d]: modifying said pixel value q_1 of said pixel next closest to said block boundary in said	

TABLE OF CONTENTS (continued)

	Page
corresponding line of pixels by adding said third offset to said pixel value q_1 of said pixel next closest to said block boundary in said corresponding line of pixels to form a modified pixel value q_1' of said pixel next closest to said block boundary in said corresponding line of pixels.....	166
5. Claim 5.....	168
a. Element 5[a]: The method according to claim 4, wherein calculating said second offset comprises calculating said second offset to be equal to $((p_2+p_0+1)>>1)-p_1+\Delta>>1$, wherein $>>$ denotes a right shift operation; and.....	169
b. Element 5[b]: calculating said third offset comprises calculating said third offset to be equal to $((q_2+q_0+1)>>1)-q_1-\Delta>>1$	171
6. Claim 6.....	174
a. Element 6[a]: The method according to claim 4, further comprising: clipping said second offset to be within an interval of $-t_{C2}$ and t_{C2} by setting said second offset to be equal to $-t_{C2}$ if said second offset is smaller than $-t_{C2}$ and by setting said second offset to be equal to t_{C2} if said second offset is larger than t_{C2} , ...	174
b. Element 6[b]: wherein $-t_{C2}$ is a threshold value that is dependent on a quantization parameter value assigned to said block; and	175
c. Element 6[c]: clipping said third offset within an interval of $-t_{C2}$ and t_{C2} by setting said third offset to be equal to $-t_{C2}$ if said third offset is smaller than $-t_{C2}$ and by setting said third offset to be equal to t_{C2} if said third offset is larger than t_{C2}	176
7. Claim 7.....	177

TABLE OF CONTENTS (continued)

	Page
a. Element 7[a]: The method according to claim 4, further comprising: clipping said modified pixel value p1' of said pixel next closest to said block boundary in said line of pixels to be within an interval of zero and a defined maximum value by setting said modified pixel value p1' of said pixel next closest to said block boundary in said line of pixels to be equal to zero if said modified pixel value p1' of said pixel next closest to said block boundary in said line of pixels is smaller than zero and by setting said modified pixel value p1' of said pixel next closest to said block boundary in said line of pixels to be equal to said defined maximum value if said modified pixel value p1' of said pixel next closest to said block boundary in said line of pixels is larger than said defined maximum value; and	177
b. Element 7[b]: clipping said modified pixel value q1' of said pixel next closest to said block boundary in said corresponding line of pixels to be within an interval of zero and said defined maximum value by setting said modified pixel value q1' of said pixel next closest to said block boundary in said corresponding line of pixels to be equal to zero if said modified pixel value q1' of said pixel next closest to said block boundary in said corresponding line of pixels is smaller than zero and by setting said modified pixel value q1' of said pixel next closest to said block boundary in said corresponding line of pixels to be equal to said defined maximum if said modified pixel value q1' of said pixel next closest to said block boundary in said corresponding line of pixels is larger than said defined maximum value.	180
8. Claim 10	182
a. Element 10[pre]: A user equipment comprising a deblocking filter unit for reducing blocking artifacts at a block boundary between a block of multiple pixels and a	

TABLE OF CONTENTS (continued)

	Page
neighboring block of multiple pixels in a video frame, wherein the deblocking filter unit comprises:	182
b. Element 10[a]: a first offset calculator configured to calculate a first offset to be equal to $(9 \times (q_0 - p_0) - 3 \times q_1 - p_1) \gg 4$, wherein $\gg$ denotes a right shift operation, p_0 denotes a pixel value of a pixel closest to said block boundary in a line of pixels in said block, wherein said line of pixels is perpendicular to said block boundary, p_1 denotes a pixel value of a pixel next closest to said block boundary in said line of pixels, q_0 denotes a pixel value of a pixel closest to said block boundary in a corresponding line of pixels in said neighboring block, wherein said corresponding line of pixels is perpendicular to said block boundary, and q_1 denotes a pixel value of a pixel next closest to said block boundary in said corresponding line of pixels; and.....	189
c. Element 10[b]: a pixel value modifier configured to modify said pixel value p_0 of said pixel closest to said block boundary in said line of pixels by adding said first offset to said pixel value p_0 of said pixel closest to said block boundary in said line of pixels to form a modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels.....	189
d. Element 10[c]: and to modify said pixel value q_0 of said pixel value q_0 of said pixel closest to said block boundary in said corresponding line of pixels to form a modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels.....	190
9. Claim 11	190
a. Element 11[a]: The user equipment according to claim 10, wherein the deblocking filter unit further comprises a first clipping unit configured to clip said first offset within an interval of $-t_c$ and t_c by setting a first offset to	

TABLE OF CONTENTS (continued)

	Page
be equal to $-t_C$ if said first offset is smaller than $-t_C$ and by setting a first offset to be equal to t_C if said first offset is larger than t_C ,.....	190
b. Element 11[b]: wherein t_C is a threshold value that is dependent on a quantization parameter value assigned to said block.	191
10. Claim 12.....	191
a. Element 12[a]: The user equipment according to claim 10, wherein the deblocking filter unit further comprises a second clipping unit configured to clip said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels to be within an interval of zero and a defined maximum value by setting said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels to be equal to zero if said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels is smaller than zero and by setting said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels to be equal to said defined maximum value if said modified pixel value of p_0' of said pixel closest to said block boundary in said line of pixels is larger than said defined maximum value and.....	191
b. Element 12[b]: to clip said modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels to be within an interval of zero and said defined maximum value by setting said modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels to be equal to zero if said modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels is smaller than zero and by setting said modified pixel value q_0' of said pixel	

TABLE OF CONTENTS (continued)

	Page
closest to said block boundary in said corresponding line of pixels to be equal to said defined maximum if said modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels is larger than said defined maximum value.....	192
B. Ground 2: Obviousness over Bjontegaard in view of Bjontegaard's Code and a POSITA's Understanding of the Art, Evidenced by List, Fu, B205, and B118.....	192
1. Claim 1	192
a. Element 1[pre]: A method of reducing blocking artifacts at a block boundary between a block of multiple pixels and a neighboring block of multiple pixels in a video frame, said method comprises:	193
b. Element 1[a]: calculating a first offset to be equal to $(9 \times (q_0 - p_0) - 3 \times (q_1 - p_1) + 8) \gg 4$, wherein $\gg$ denotes a right shift operation, p_0 denotes a pixel value of a pixel closest to said block boundary in a line of pixels in said block, wherein said line of pixels is perpendicular to said block boundary, p_1 denotes a pixel value of a pixel next closest to said block boundary in said line of pixels, q_0 denotes a pixel value of a pixel closest to said block boundary in a corresponding line of pixels in said neighboring block, wherein said corresponding line of pixels is perpendicular to said block boundary and q_1 denotes a pixel value of a pixel next closest to said block boundary in said corresponding line of pixels;.....	195
c. Element 1[b]: modifying said pixel value p_0 of said pixel closest to said block boundary in said line of pixels by adding said first offset to said pixel value p_0 of said pixel closest to said block boundary in said line of pixels to form a modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels; and	204

TABLE OF CONTENTS (continued)

	Page
d. Element 1[c]: modifying said pixel value q_0 of said pixel closest to said block boundary in said corresponding line of pixels by subtracting said first offset from said pixel value q_0 of said pixel closest to said block boundary in said corresponding line of pixels to form a modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels.....	206
2. Claim 2.....	207
a. Element 2[a]: The method according to claim 1, further comprising clipping said first offset to be within an interval of $-t_c$ and t_c by setting said first offset to be equal to $-t_c$ if said first offset is smaller than $-t_c$ and by setting said first offset to be equal to t_c if said first offset is larger than t_c ,.....	208
b. Element 2[b]: wherein t_c is a threshold value that is dependent on a quantization parameter value assigned to said block.	211
3. Claim 3.....	213
a. Element 3[a]: The method according to claim 1, further comprising: clipping said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels to be within an interval of zero and a defined maximum value by setting said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels to be equal to zero if said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels is smaller than zero and by setting said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels to be equal to said defined maximum value if said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels is larger than said defined maximum value; and	213

TABLE OF CONTENTS **(continued)**

	Page
b. Element 3[b]: clipping said modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels to be within an interval of zero and said defined maximum value by setting said modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels to be equal to zero if said modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels is smaller than zero and by setting said modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels to be equal to said defined maximum if said modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels is larger than said defined maximum value.....	216
4. Claim 4.....	218
a. Element 4[a]: The method according to claim 1, further comprising: calculating a second offset to be based on $(p_2 + p_0[-]2 \times p_1 + 2 \times \Delta)/4$, wherein p_2 denotes a pixel value of a pixel second next closest to said block boundary in said line of pixels and Δ denotes said first offset;	218
b. Element 4[b]: calculating a third offset to be based on $(q_2 + q_0 - 2 \times q_1 - 2 \times \Delta)/4$, wherein q_2 denotes a pixel value of a pixel second next closest to said block boundary in said corresponding line of pixels;	227
c. Element 4[c]: modifying said pixel value p_1 of said pixel next closest to said block boundary in said line of pixels by adding said second offset to said pixel value p_1 of said pixel next closest to said block boundary in said line of pixels to form a modified pixel value p_1' of said pixel next closest to said block boundary in said line of pixels; and.....	232

TABLE OF CONTENTS (continued)

	Page
d. Element 4[d]: modifying said pixel value q_1 of said pixel next closest to said block boundary in said corresponding line of pixels by adding said third offset to said pixel value q_1 of said pixel next closest to said block boundary in said corresponding line of pixels to form a modified pixel value q_1' of said pixel next closest to said block boundary in said corresponding line of pixels.....	235
5. Claim 5	237
a. Element 5[a]: The method according to claim 4, wherein calculating said second offset comprises calculating said second offset to be equal to $((p_2 + p_0 + 1) \gg 1) - p_1 + \Delta \gg 1$, wherein $\gg$ denotes a right shift operation; and.....	238
b. Element 5[b]: calculating said third offset comprises calculating said third offset to be equal to $((q_2 + q_0 + 1) \gg 1) - q_1 - \Delta \gg 1$	240
6. Claim 6	243
a. Element 6[a]: The method according to claim 4, further comprising: clipping said second offset to be within an interval of $-t_{C2}$ and t_{C2} by setting said second offset to be equal to $-t_{C2}$ if said second offset is smaller than $-t_{C2}$ and by setting said second offset to be equal to t_{C2} if said second offset is larger than t_{C2} , ...	243
b. Element 6[b]: wherein $-t_{C2}$ is a threshold value that is dependent on a quantization parameter value assigned to said block; and	245
c. Element 6[c]: clipping said third offset within an interval of $-t_{C2}$ and t_{C2} by setting said third offset to be equal to $-t_{C2}$ if said third offset is smaller than $-t_{C2}$ and by setting said third offset to be equal to t_{C2} if said third offset is larger than t_{C2}	246

TABLE OF CONTENTS **(continued)**

	Page
7. Claim 7.....	250
a. Element 7[a]: The method according to claim 4, further comprising: clipping said modified pixel value p1' of said pixel next closest to said block boundary in said line of pixels to be within an interval of zero and a defined maximum value by setting said modified pixel value p1' of said pixel next closest to said block boundary in said line of pixels to be equal to zero if said modified pixel value p1' of said pixel next closest to said block boundary in said line of pixels is smaller than zero and by setting said modified pixel value p1' of said pixel next closest to said block boundary in said line of pixels to be equal to said defined maximum value if said modified pixel value p1' of said pixel next closest to said block boundary in said line of pixels is larger than said defined maximum value; and	251
b. Element 7[b]: clipping said modified pixel value q1' of said pixel next closest to said block boundary in said corresponding line of pixels to be within an interval of zero and said defined maximum value by setting said modified pixel value q1' of said pixel next closest to said block boundary in said corresponding line of pixels to be equal to zero if said modified pixel value q1' of said pixel next closest to said block boundary in said corresponding line of pixels is smaller than zero and by setting said modified pixel value q1' of said pixel next closest to said block boundary in said corresponding line of pixels to be equal to said defined maximum if said modified pixel value q1' of said pixel next closest to said block boundary in said corresponding line of pixels is larger than said defined maximum value.....	253
8. Claim 10.....	255
a. Element 10[pre]: A user equipment comprising a deblocking filter unit for reducing blocking artifacts at a	

TABLE OF CONTENTS (continued)

	Page
block boundary between a block of multiple pixels and a neighboring block of multiple pixels in a video frame, wherein the deblocking filter unit comprises:	255
b. Element 10[a]: a first offset calculator configured to calculate a first offset to be equal to $(9 \times (q_0 - p_0) - 3 \times q_1 - p_1) \gg 4$, wherein $\gg$ denotes a right shift operation, p_0 denotes a pixel value of a pixel closest to said block boundary in a line of pixels in said block, wherein said line of pixels is perpendicular to said block boundary, p_1 denotes a pixel value of a pixel next closest to said block boundary in said line of pixels, q_0 denotes a pixel value of a pixel closest to said block boundary in a corresponding line of pixels in said neighboring block, wherein said corresponding line of pixels is perpendicular to said block boundary, and q_1 denotes a pixel value of a pixel next closest to said block boundary in said corresponding line of pixels; and.....	257
c. Element 10[b]: a pixel value modifier configured to modify said pixel value p_0 of said pixel closest to said block boundary in said line of pixels by adding said first offset to said pixel value p_0 of said pixel closest to said block boundary in said line of pixels to form a modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels.....	258
d. Element 10[c]: and to modify said pixel value q_0 of said pixel value q_0 of said pixel closest to said block boundary in said corresponding line of pixels to form a modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels.....	258
9. Claim 11	258
a. Element 11[a]: The user equipment according to claim 10, wherein the deblocking filter unit further comprises a first clipping unit configured to clip said first offset	

TABLE OF CONTENTS (continued)

	Page
within an interval of $-t_C$ and t_C by setting a first offset to be equal to $-t_C$ if said first offset is smaller than $-t_C$ and by setting a first offset to be equal to t_C if said first offset is larger than t_C ,.....	259
b. Element 11[b]: wherein t_C is a threshold value that is dependent on a quantization parameter value assigned to said block.	259
10. Claim 12.....	259
a. Element 12[a]: The user equipment according to claim 10, wherein the deblocking filter unit further comprises a second clipping unit configured to clip said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels to be within an interval of zero and a defined maximum value by setting said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels to be equal to zero if said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels is smaller than zero and by setting said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels to be equal to said defined maximum value if said modified pixel value of p_0' of said pixel closest to said block boundary in said line of pixels is larger than said defined maximum value and.....	260
b. Element 12[b]: to clip said modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels to be within an interval of zero and said defined maximum value by setting said modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels to be equal to zero if said modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels is smaller than zero and by	

TABLE OF CONTENTS
(continued)

	Page
setting said modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels to be equal to said defined maximum if said modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels is larger than said defined maximum value.....	261
XI. SUMMARY AND ADDITIONAL REMARKS	261

I, Dr. Dan Schonfeld, do hereby declare:

I. INTRODUCTION

1. My name is Dan Schonfeld. I am currently a Professor in the Department of Electrical and Computer Engineering at the University of Illinois at Chicago.

2. I have been retained by Petitioner Intel Corporation (“Intel” or “Petitioner”) to submit a Declaration in support of Intel’s Petition for *Inter Partes* Review of claims 1-7 and 1-12 (“Challenged Claims”) of U.S. Patent No. 10,142,659 (“’659 Patent”) (Ex-1001).

3. I have been asked to opine on whether the Challenged Claims of the ’659 Patent are rendered obvious by the prior art. My opinions are based on my years of education, research, and experience, as well as my investigation and study of relevant materials. The materials that I evaluated in support of this Declaration include all exhibits cited in this Declaration and in the Petition.

4. I may rely upon these materials, my knowledge and experience, and/or additional materials to rebut arguments raised by the Patent Owner. Further, I may also consider additional documents and information in forming any necessary opinions, including documents that may not yet have been provided to me.

5. My analysis of the materials produced in this matter is ongoing, and I will continue to review any new material as it is provided. This Declaration

represents only those opinions I have formed to date. I reserve the right to revise, supplement, and/or amend my opinions stated herein based on new information and on my continuing analysis of the materials already provided.

6. I am being compensated for my services at my standard hourly rate and reimbursed for reasonable expenses that I may incur while working on this matter. I have no financial interest in the outcome of this matter or any other matter that may exist between Petitioners and Telefonaktiebolaget LM Ericsson (“Ericsson” or “Patent Owner”). This compensation does not depend in any way on the conclusions I reach.

II. QUALIFICATIONS

7. My qualifications for forming the opinions set forth in this Declaration are summarized here and include my educational background, career history, publications, and other relevant qualifications. My full *curriculum vitae* is available at Ex-1004, and includes my detailed employment background, professional experience, and list of publications.

8. I received my B.S. degree in Electrical Engineering and Computer Science from the University of California, Berkeley, California, in 1986 with a concentration on Computer Engineering/Systems Engineering. I received my M.S. degree in Electrical and Computer Engineering from The Johns Hopkins University, Baltimore, Maryland, in 1988 with a concentration on Speech

Processing/Biomedical Signal Processing. I received my Ph.D. degree in Electrical and Computer Engineering from The Johns Hopkins University, Baltimore, Maryland, in 1990 with a concentration on Nonlinear Signal Processing/Image Analysis.

9. In August 1990, I joined the Department of Electrical Engineering and Computer Science at the University of Illinois, Chicago, Illinois, where I am a tenured Professor in the Departments of Electrical and Computer Engineering, Computer Science, and Bioengineering. Before I joined the University of Illinois at Chicago, I served as an instructor in the Department of Electrical and Computer Engineering at The Johns Hopkins University, Baltimore, Maryland.

10. At the University of Illinois at Chicago, I have served as the Director of the University-Industry Engineering Research Center (UIERC), formerly the Manufacturing Research Center (MRC). I am also Co-Director of the Multimedia Communications Laboratory (MCL) and a member of the Signal and Image Research Laboratory (SIRL).

11. Over the past few decades, I have also served as a visiting professor in (a) the Advanced Analytics Institute (AAI) at the University of Technology, Sydney, Australia, (b) the Department of Information Engineering and Computer Science (DISI) at the University of Trento, Italy, (c) the School of Computer Engineering at

the Nanyang Technological University, Singapore, and (d) the Department of Electrical Engineering—Systems at Tel-Aviv University, Israel.

12. I have been elected Fellow of the Institute of Electrical and Electronics Engineers (IEEE) “for contributions to image and video analysis” as well as Fellow of the International Society for Optics and Photonics (“SPIE”) “for specific achievements in morphological image processing and video analysis.” I have also been elected University Scholar of the University of Illinois. A complete list of my publications, professional activities, and honors that I have received is fully set forth in my *curriculum vitae*, available at Ex-1004.

13. I have previously served as Editor-in-Chief and Deputy Editor-in-Chief of the IEEE Transactions on Circuits and Systems for Video Technology. I have also previously served as Area Editor for special issues of the IEEE Signal Processing Magazine. I have served as Associate Editor of the IEEE Transactions on Image Processing (on image and video storage, retrieval and analysis), Associate Editor of the IEEE Transactions on Circuits and Systems for Video Technology (on video analysis), Associate Editor of the IEEE Transactions on Signal Processing (on multidimensional signal processing and multimedia signal processing), and Associate Editor of the IEEE Transactions on Image Processing (on nonlinear filtering). I have also served on the editorial board of the IEEE Signal Processing Magazine, EURASIP Journal of Image and Video Processing, Research Letters in

Signal Processing, and Bentham Science Publishers, Ltd.'s "Recent Patents on Computer Science" and "Recent Patents on Electrical Engineering" publications. I have served as guest editor of numerous special issues in various journal publications in the area of multimedia systems.

14. I have previously served on the Conference Board of the IEEE Signal Processing Society. I have previously served as Technical Program Chair of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) 2018 as well as Program Chair of the IEEE Conference on Visual Communications and Image Processing (VCIP) 2015. I have also previously served as General Co-Chair of the Workshop on Big Data in 3D Computer Vision 2013 and the IEEE International Conference on Multimedia and Expo (ICME) 2012. I have served as Chair of the IEEE Workshop on Video Mining 2008 and the SPIE Conference on Visual Communications and Image Processing 2007. I have also served on the organizing committees of various conferences including the IEEE International Conference on Image Processing 1998, 2012, and 2020, IEEE/SPIE VCIP 2010, 2017, and IEEE Workshop on Nonlinear Signal and Image Processing (NSIP) 1997. I was an organizer of the Thematic Symposium on Multimedia Search and Retrieval at ICASSP 2009.

15. I have authored and co-authored over 250 technical papers for various journals and conferences. I was author of a book chapter, entitled: "Image and video

communication networks,” and later editions entitled: “Video communication networks.” I was co-author (with Carlo Giulietti and Rashid Ansari) of a paper that won the Best Paper Award at the ACM Multimedia Workshop on Advanced Video Streaming Techniques for Peer-to-Peer Networks and Social Networking 2010. I was also co-author (with Junlan Yang) of a paper that won the Best Student Paper Award at the IEEE International Conference on Image Processing 2007. I was also co-author (with Wei Qu) of a paper that won the Best Student Paper Award at the IEEE International Conference on Image Processing 2006. I was also co-author (with Nidhal Bouaynaya) of a paper that won the Best Student Paper Award in Visual Communications and Image Processing 2006. In addition, many of my publications relate to the broad topic of multimedia systems, which includes audio, image, and video processing. My publications in the area of multimedia systems dates back to 1988. A list of my publications within the past ten years is included in Ex-1004.

16. I have been selected to be the keynote speaker at the IEEE International Conference on Computer, Electrical and Communication Engineering (ICCECE), Kolkata, India, in 2023. I was also the keynote speaker at the N-Brain Alliance Launch & Product Release Conference and the Think for Education Conference (TEC) AI+Education, both held in Beijing, China, in 2018. I was also the keynote speaker at the International Conference on Wireless Communications and Signal

Processing (WCSP), Yangzhou, China, in 2016, and the International Conference on Intelligent Control and Information Processing (ICICIP) and International Conference on Brain Inspired Cognitive Systems (BICS), Beijing, China, in 2013. Further, I was a plenary speaker at the IEEE/IET International Conference on Audio, Language and Image Processing (ICALIP), Shanghai, China, in 2010, and at the IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Genoa, Italy, in 2009. I was also a plenary speaker at the INPT/ASME International Conference on Communications, Signals, and Systems (ICCSS), Rabat Morocco, in 1995 and 2001.

17. I have served as Representative of Regions 1-6 (North America) on the Chapters Committee of the IEEE Signal Processing Society. I have also served as Chairman of the IEEE Signal Processing Chicago Chapter. I have also served on the IEEE Image, Video, and Multidimensional Signal Processing (IVMSP) Technical Committee, formerly the IEEE Image and Multidimensional Signal Processing (IMDSP) Technical Committee, Visual Signal Processing and Communications (VSPC) Technical Committee, IEEE Signal and Image Processing in Medicine Technical Committee, and the IEEE Multimedia Communications Technical Committee. I currently serve on the American National Standards Institute (ANSI)/Underwriters Laboratory (UL) Standards Technical Panel (“STP”) on Multimedia Systems.

18. I have also taught various courses that relate to multimedia systems. For example, since the late 1990s, I have introduced and taught an advanced undergraduate-level/first-year graduate-level course on multimedia systems (originally called multimedia communication networks), which focuses on audio, image, and video processing and communications.

19. I have also served as a consultant in various engagements related to multimedia systems. For example, over the past decade, I have served as an expert witness in several cases related to multimedia systems. In 1997, I served as a consultant for Prairiecomm Corp. where, among other tasks, I developed and deployed multimedia systems. Since 2002, I have also served as a member of the American National Standards Institute (ANSI)/Underwriters Laboratory (UL) Standards Technical Panel (STP) on various standards related to multimedia systems.

20. Additional details of my education and work experience, professional activities, awards and honors, and publications that may be relevant to opinions I have formed are set forth in my *curriculum vitae*, which is available at Ex-1004. Additionally, I have consulted for several companies in the area of signal processing and multimedia systems.

III. MATERIALS CONSIDERED

21. The materials that I considered and relied upon in preparing my Declaration and forming my opinions include all exhibits cited in this Declaration and the Petition, including the '659 Patent, the '659 file history, and all of the relevant prior art.

22. I also have relied on my academic and professional experience in reaching the opinions expressed in this Declaration.

IV. LEGAL STANDARDS

23. Counsel has informed me of the legal principles that apply in determining whether patent claims are unpatentable for obviousness. Paragraphs in this section beginning with “I understand” or “I have been informed” capture my understanding of the law as provided to me by counsel. The other paragraphs are my opinion.

24. I understand that a claim may be unpatentable as obvious under § 103(a) if the subject matter described in the claim as a whole would have been obvious to a hypothetical “person of ordinary skill in the art” (“POSITA”) in view of a prior-art reference, or combination of prior-art references at the time the claimed invention was made.

25. I understand that “the time the claimed invention was made” refers to the “priority” or “critical” date of the invention.

26. I understand that, before considering the prior art, the claims are first construed to determine what subject matter the claims encompass. I understand that the claims are read from the perspective of POSITA at the time of the alleged invention. I understand that the words of a claim are generally given their ordinary and customary meaning as understood by a POSITA in light of the entire patent specification in which they appear, as well as the prosecution history of the patent.

27. I understand, however, that the words of a claim may be given a meaning other than their plain and ordinary meaning if: (1) a patentee expressly and unambiguously defined a term in a way that is different from its plain meaning; or (2) a patentee “disclaims” the full scope of a claim term’s plain meaning, for instance, by making clear and unambiguous statements during prosecution that a claim term does or does not encompass certain subject matter that may otherwise be encompassed by the plain words of the claim.

28. I understand that, after construing the claims, the teachings of the prior art are considered. I understand that the prior art includes references, among other things, that were known or used by others in this country, or patented or described in a printed publication in this or a foreign country, before the date the invention was allegedly made by the inventor(s). For a printed publication to qualify as prior art, I understand that it must be demonstrate that the publication was disseminated or otherwise sufficiently accessible to the public. However, I also understand that a

POSITA is assumed to be aware of all relevant prior art in the field of endeavor covered by the patent-at-issue and all analogous prior art.

29. I understand that although the ultimate question of the obviousness of a claimed invention is a legal determination, obviousness is based on several factual inquiries, including: the scope and content of the prior art, the differences between the claimed subject matter and the prior art, the level of ordinary skill in the art at the time of the invention, and any “objective indicia” or “secondary considerations” of non-obviousness, which must all be considered.

30. I understand that, in determining the scope and content of the prior art, a prior-art reference is considered relevant if it falls within the field of the inventor’s endeavor, or if it is reasonably pertinent to the particular problem that the inventor was trying to solve. A reference is reasonably pertinent if it logically would have commended itself to an inventor’s attention in considering the problem sought to be solved.

31. I understand that, to assess the differences between prior art and the claimed subject matter, the claimed invention must be considered as a whole. I understand that this involves showing that a POSITA, confronted by the same problems as the inventor and with no knowledge of the claimed invention, would have identified the elements in the prior art and combined them in the claimed manner.

32. I understand that there are several rationales for combining prior-art references. For instance, I understand that it is considered obvious to:

- combine prior-art elements according to known methods to yield predictable results;
- substitute one known element for another to obtain predictable results;
- use the prior-art elements in a predictable way according to their established functions;
- apply a known technique to a known device (method or product) ready for improvement to yield predictable results;
- choose from a finite number of identified, predictable solutions, with a reasonable expectation of success;
- or if there is some teaching, suggestion, or motivation in the prior art that would have led a POSITA to modify a prior-art reference or combine prior-art teachings to arrive at the claimed invention.

I understand, however, that a POSITA does not need to have the same motivations or reasons to combine the prior art as the inventors, and can arrive at the claimed invention for completely different reasons than the inventors.

33. I also understand that a claim may be considered obvious in light of the prior art, even if no one prior-art reference provided its exact teachings, if the general conditions of the claimed subject matter are disclosed in the prior art, and the claimed

invention is simply an optimization of the prior art that could be achieved through routine experimentation. I understand that, for a claim to be obvious on this basis, it must be demonstrated that a POSITA would have recognized that a particular variable in the prior art was “result-effective”—meaning that modifying the variable was known to affect a particular result—and optimizing that variable would have been within the skill of a POSITA.

34. However, I understand that even if the general working conditions of the claimed subject matter were known, the claims may nonetheless be non-obvious if: (1) the claimed invention produces a new and unexpected result that is different “in kind” and not merely “in degree” from that of the prior art; (2) the prior art taught away from the claimed invention; (3) the claimed variable was not recognized as “result-effective”; or (4) the prior art disclosed a broad range that would not invite routine optimization. I also understand that a difference “in kind” means some transformative difference beyond what was recognized by the prior art, and not merely a change of percentages in, *e.g.*, efficiency.

35. I understand that, when considering the prior art, the prior art cannot be modified or combined using “hindsight.” I understand that hindsight refers to situations in which the patent-at-issue is used as a framework to pick and choose pieces from the prior art to arrive at the claims, or adopting rationales identified by the patent-at-issue that those in the art would not have recognized. Instead, I

understand that the obviousness inquiry is evaluated from the perspective of a POSITA as of the critical date of the patent-at-issue, with all the prior art before them, to determine whether the POSITA would have independently arrived at the claimed invention without the benefit of the patent-at-issue for guidance.

36. I understand that “objective indicia” or “secondary considerations” of non-obviousness must also be considered in the obviousness analysis, and include whether: (1) there was a long-felt need for the claimed invention; (2) the claimed invention has achieved commercial success; (3) there was copying of the claimed invention by others; (4) there were failed attempts by others to make the alleged invention; (5) there was praise of the invention in the field; and (6) there was skepticism the claimed invention could be achieved, among a few others.

37. As of the time that I prepared this Declaration, the Patent Owner had not identified any secondary considerations of non-obviousness. Therefore, I reserve the right to elaborate on my understanding of these secondary considerations and to address any secondary considerations that Patent Owner may subsequently identify.

38. Finally, I understand that, in an *inter partes* review proceeding, the obviousness of a claim must be demonstrated by “a preponderance of the evidence,” and that this burden falls on the Petitioner. I understand that the preponderance of the evidence standard means that a reasonable factfinder would find a material fact more probable than the nonexistence of that fact. It does not allow for speculation

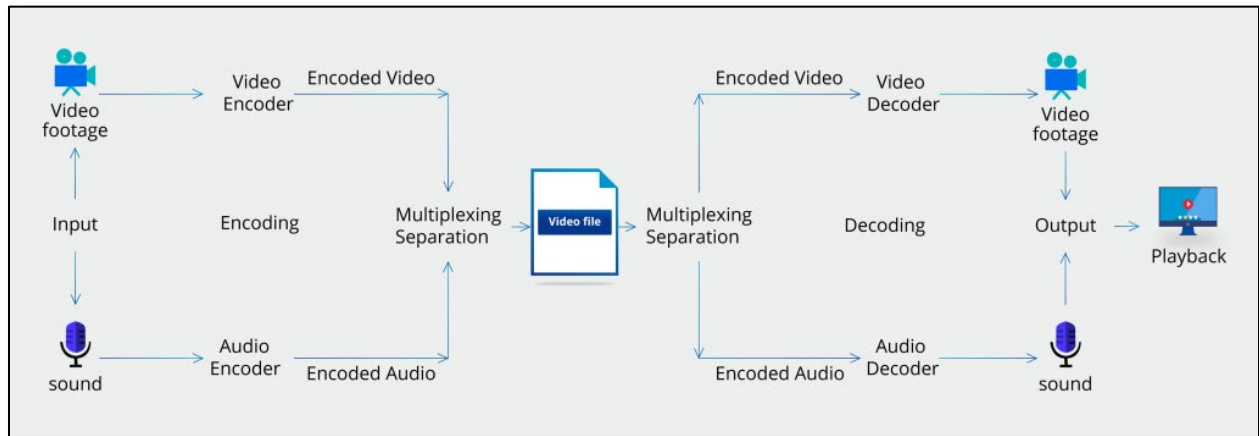
regarding specific facts and is instead focused on whether the evidence more likely than not demonstrates the existence or non-existence of specific material facts. I understand that “preponderance of the evidence” is a lower standard than “clear and convincing evidence” (which requires a fact to be substantially more likely to be true than untrue), or “beyond a reasonable doubt” (which is an exceedingly high standard that I understand is generally reserved for criminal matters).

39. I have applied the “preponderance of the evidence” standard throughout my Declaration.

V. TECHNOLOGY BACKGROUND

A. Video Encoding and Decoding

40. Digital videos must generally undergo an encoding process to convert the images (also called “pictures” or “frames”) that make up the video into a binary bitstream that can be processed and stored by computers. As part of the encoding process, videos are usually “compressed” to reduce the amount of data needed to represent the video. When the video is to be displayed on a computer screen, television, or some other display device, the binary bitstream undergoes a decoding process that essentially reverses the encoding process to reconstruct the video images from the bitstream.



Ex-1031, 1.¹

41. The encoder encodes video images by performing several steps. At a high level, the key steps involved in the encoding of a video image are provided below.

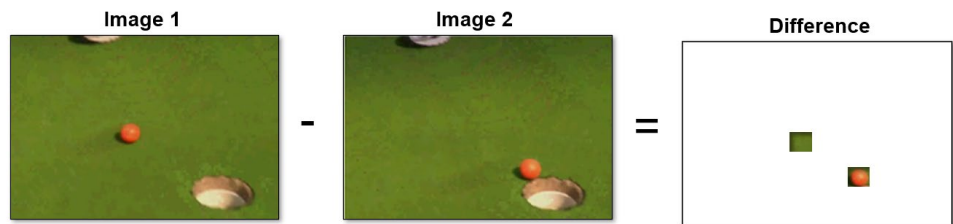
- a. **Blocking:** The encoder subdivides the image into blocks and processes each block individually to alleviate demands on processing resources. By way of example, an image may be divided into blocks of 16x16 pixels, 8x8 pixels, or even 4x4 pixels, although it was known in the art that different-sized blocks could be used.

¹ Although some of the images I identify in the technology background may post-date the earliest-possible priority date of the patent, they are provided only as useful demonstratives of certain concepts, and the aspects on which I am relying on them were known prior to the earliest-possible priority date of the patent.



Ex-1070, 4.

b. **Prediction and Residuals:** The encoder achieves much of the compression by characterizing, when possible, only the difference between different images in the video.



The encoder characterizes the differences between images in two ways: (1) the motion of objects between two images; and (2) any remaining “residual” differences between the images. First, the encoder characterizes the motion of objects by taking a block in the current image and searching a previously encoded “reference” image to find a set of pixels with a similar arrangement of pixels as the current block. The encoder then

characterizes how those pixels have moved between the two images in the form of a “motion vector.” The encoder will then apply the motion vector to identify pixels in the reference image to create a “prediction” block of the current image. This “prediction” block is then subtracted from the current block to capture any remaining differences between the current block and prediction block, which is called “residual” data. If the prediction is accurate, then the difference is small, which significantly reduces the amount of data that is output by the encoder. At least two types of prediction were commonly used prior to the ’659 Patent: (1) inter-picture prediction, which predicts a current block based on the corresponding block in a reference frame, and (2) intra-picture prediction, which predicts a current block based on adjacent blocks in the same frame.

c. **Transforms:** The encoder then applies an orthogonal transform (Discrete Cosine Transform, or “DCT”) to the residual data to transform them to another domain in order to achieve further compression. Specifically, the data is transformed from the block’s spatial data to frequency data. The human eye is more sensitive to the information contained in low frequencies (corresponding to larger, less-variable features in an image) than to the information contained in high frequencies (corresponding to smaller, rapidly changing features). The DCT separates

the more perceptually significant information from less perceptually significant information by evaluating the contribution of different frequency components to the residual data to produce a “DCT coefficient” characterizing each frequency component’s contribution. After the DCT, the compression algorithm encodes the low-frequency DCT coefficients with high precision, but uses fewer bits to encode the high-frequency DCT coefficients as part of a “quantization” process.

d. **Entropy encoding:** Finally, the encoder applies an entropy coding technique to convert the DCT coefficients into codewords to achieve further compression. An entropy coding technique maps frequently occurring DCT coefficients to shorter codewords and infrequently occurring DCT coefficients to longer code words to achieve compression. This is also known as “variable length coding.” One early example of this process is “Morse code,” in which more-often used letters (e.g., “e” and “t”) are assigned the shortest codewords (one dot or one dash, respectively), and less-used letters have progressively longer codes. The decoder performs the reverse and employs the same mapping to convert the codewords into the DCT coefficients.

42. In sum, the output of the encoder is a bitstream that contains a complete representation of a video image, or when an image is based on a “prediction” of

another fully encoded image, the motion vectors and residuals needed to reconstruct the predicted image. When the decoder receives the bitstream containing fully encoded images, or the motion vectors and residual data for predicted images, it will then reverse the encoding process. For instance, whereas the encoder calculates a motion vector by comparing pixels in the current block to a reference, the decoder will apply the motion vector to a reference image to create the prediction block. Then, the decoder will reverse the quantization and transform process performed by the encoder—recreating the “residual data” from its compressed, encoded form in the frequency domain back into spatial pixel data—and *add* the residual data back to the prediction block to create a reconstructed version of the current block.

43. An exemplary overview of the encoding process consistent with the one I have described above can be found at Ex-1040, 28-35 (motion compensation), 42-43, 45-47 (transform), 50-52 (quantization), 58 (entropy encoding); Ex-1030.

B. Deblocking Filters

44. One drawback of block-based encoding and decoding is that reconstructed images may contain perceptible pixel-color changes at the boundaries of neighboring blocks, called “blocking artifacts,” because of the way each block is processed and quantized separately. For instance, because the transform and quantization process employed during encoding may discard higher-frequency DCT coefficients so that the encoded video requires less data than the original video, when

the video image is subsequently decoded and reconstructed using predominantly lower-frequency DCT coefficients, the high-frequency details in the original image would not be present, which contributes to the generation of blocking artifacts. An example of a reconstructed image with noticeable “blocking artifacts” is shown below.



Ex-1052, 1372.

45. To reduce the apparent visibility of these blocking artifacts, reconstructed images can be processed using a “deblocking filter.” In essence, a deblocking filter will evaluate the boundaries between adjacent blocks and, when certain conditions are met, apply a filter to blend the pixels along the boundary between neighboring blocks. As shown in the example below, the lines that were visible between different blocks in the image above are less noticeable.

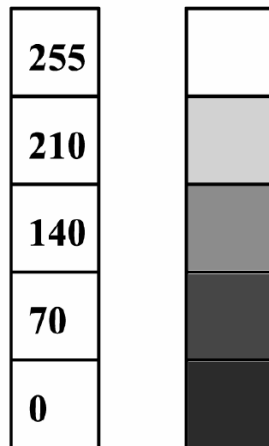


Ex-1052, 1372.

1. Pixel Values

46. Deblocking filters operate on the pixels that make up each video image. Digital video images are made up of pixels, which can be mapped to pixels on an output device (such as a computer display or television). For instance, each image in a 1080p video typically has 1920x1080 pixels, and each image in a 4K video typically has 3840x2160 pixels.

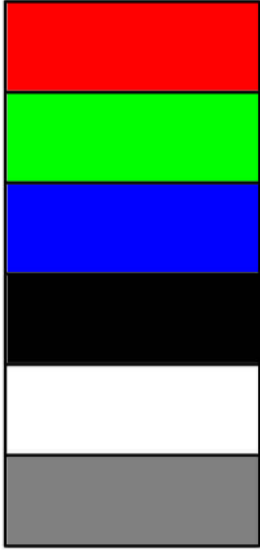
47. Each pixel in a video image can be represented by a pixel value. For instance, an 8-bit greyscale image means that each pixel can be represented using 8 bits of data (8 zeros or ones), which would result in 256 (or 2^8) possible pixel values. *See, e.g.,* Ex-1034, 2. Each pixel in an 8-bit greyscale image can therefore be assigned a value between 0 (pure black) to 255 (pure white), as shown below:



Id.

48. The pixels in a color image can be represented using one of several color-representation schemes. One such scheme is called the RGB scheme that relies upon Red, Green, and Blue—the three primary colors. *Id.*, 6. With this scheme, each color component (red, green, and blue) can be assigned, for instance, its own 8-bit value—*each* with a value between 0 and 255—such that each pixel is represented by 24 bits in total. *Id.* This produces approximately 16.7 million (or 2^{24}) possible color combinations. *Id.*

R / G / B		
255	0	0
0	255	0
0	0	255
0	0	0
255	255	255
128	128	128



When a pixel is displayed, the three color components are combined to realize the final color. *See id.*, 7.

49. Alternately, the same information regarding the color of a pixel can be represented by Luminance or Luma (Y) and three color differences. Luminance (Y) is a weighted average of the R, G and B components of a pixel. Ex-1029, 3; Ex-1040, 13-14. The color differences (Chrominance or Chroma) represent the differences between the three color components and the luminance:

$$CR = R - Y$$

$$CB = B - Y$$

$$CG = G - Y$$

Ex-1029, 3; Ex-1040, 13-14.

50. Because the sum of CR, CB and CG is constant, only two of the three values are used. The third value can be derived from the other two. Typically, CR and CB are used along with Y to represent colors. Ex-1029, 4; Ex-1040, 14. In the Y:CR:CB scheme, only the Luminance/Luma (Y), the red chroma (CR) and blue chroma (CB) are used. The human eye is more sensitive to luminance than to color. Ex-1040, 13. As a result, the CR and the CB components can be represented with a lower resolution than the Luminance component without affecting the visual quality of the image. *Id.* Thus, the number of bits required to represent an image in the Y:CR:CB scheme is fewer than the bits needed in the RGB scheme. Many video codecs—such as the H.263 and H.264 Standards—describe the data contained in macroblocks and blocks using the Y:CR:CB scheme. Ex-1037, 4 (“As in previous video coding Recommendations and International Standards, a macroblock, consisting of a 16x16 block of luma samples and two corresponding blocks of chroma samples, is used as the basic processing unit of the video decoding process.”).

51. Because pixels are represented as numeric values, they can also be operated on, such as added, subtracted, multiplied, and divided. When digital images are edited on a computer, for instance, to change the brightness or contrast, or apply pre-determined “filters,” behind the scenes, mathematical equations modify the pixel values.

52. Regardless of the pixel-value scheme used, a deblocking filter implements mathematics to alter the pixel values of pixels near block boundaries in an attempt to smooth out the differences between neighboring blocks. A simplistic example of how a deblocking filter might operate is provided below, showing two neighboring blocks (a “P” block and “Q” block) that each have particular pixel values (100 or 175, respectively), that produce a perceptible change at the boundary.

P Block			Q Block		
p_2	p_1	p_0	q_0	q_1	q_2
100	100	100	175	175	175
100	100	100	175	175	175
100	100	100	175	175	175
100	100	100	175	175	175
100	100	100	175	175	175

53. There are two primary ways to calculate new pixel values during deblocking. The first is to calculate the new pixel value directly using a set of original pixel values. For instance, using an averaging filter, a new pixel value for the pixel closest to the boundary (p_0 , which I label p_0') can be calculated using the following example equation:

$$p_0' = \frac{p_1 + p_0 + q_0}{3} = \frac{100 + 100 + 175}{3} = 125$$

$$q_0' = \frac{q_1 + q_0 + p_0}{3} = \frac{175 + 175 + 100}{3} = 150$$

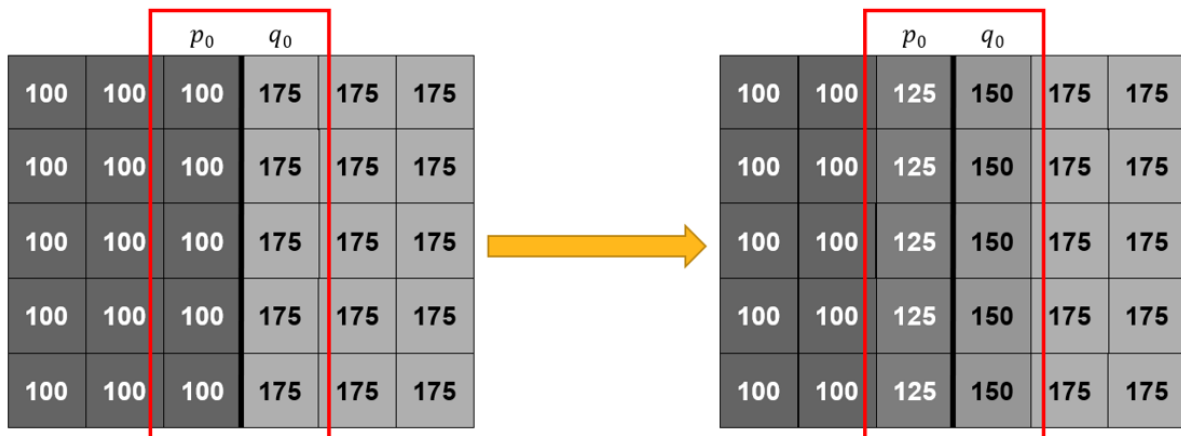
Alternatively, an intermediate “delta” or “offset” (Δ) value can be calculated, which is then *applied* to the original pixel values (*e.g.*, added to one side and subtracted from the other). For instance, one example offset equation could be:

$$\Delta = \frac{q_0 - p_0}{3} = \frac{175 - 100}{3} = 25$$

$$p_0' = p_0 + \Delta = 100 + 25 = 125$$

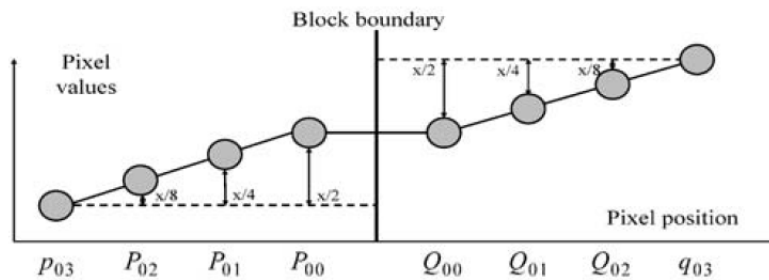
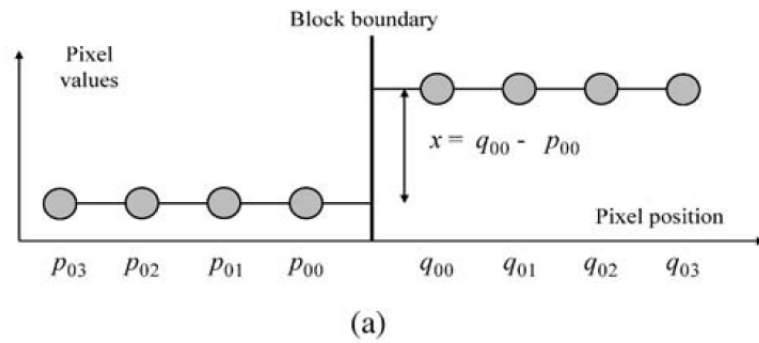
$$q_0' = q_0 - \Delta = 175 - 25 = 150$$

While these two different techniques produced the same result in this very simplified example, that is not always the case, because how much a pixel is modified depends on the specific equations and pixel values used. Regardless, modifying the pixel values in this example using these example equations produces a smoother transition between the two blocks:



This can also be depicted graphically, as the image below shows how multiple pixels are filtered to produce a smooth transition over a block boundary by calculating the

difference across the boundary (“x”) and dividing the difference by a larger integer (2, 4, 8) the further the pixel is from the block boundary to produce a smooth transition:



Ex-1039, 215.

2. Filters

54. Pixel values are often “filtered” based on the frequency characteristics of the image block, which refers to how pixel values fluctuate over a given space. For instance, below is a signal that consists of a set of pixels that are considered to have a relatively “low frequency”:

150	160	170	180	170	160	150	160
-----	-----	-----	-----	-----	-----	-----	-----

See Gonzalez, 258 (“low frequencies ... are related to slowly varying intensity components in an image”). As can be seen, the pixel values are not changing significantly over the 8-pixel space shown. Compare this, however, to a signal that consists of pixel values with a relatively “high frequency”:

0	150	255	150	0	255	0	150
---	-----	-----	-----	---	-----	---	-----

Id. (“high frequencies are caused by sharp transitions in intensity, such as edges and noise.”).

55. Although filters can serve many different purposes, at a high level, filters are often classified into one of three frequency-discriminator filters: “low-pass” filters, “high-pass” filters, and “band-pass” filters.

56. A low-pass filter means that the filter is intended to allow lower-frequency components to “pass” through from input to output, and filter (or suppress) high-frequency components. Gonzalez, 258. In image processing, a low-pass filter reduces rapid changes in pixel values. One use of a low-pass filter is to remove spurious data such as random noise. A simple way to achieve this is by averaging filters, in which the filter takes multiple pixel values as inputs and produces an average value as its output, thereby reducing rapid changes in pixel values. An artifact of low-pass filtering, including simple averaging, is the introduction of “blurring” to the filtered image. *Id.* (explaining how a “lowpass filter

... would blur an image”). For example, if we average each pixel value in the above low-frequency signal example based on the values of its two neighboring pixels (except for the left-most and right-most pixel, which have only one neighboring pixel), the following output would be provided:

Input (Low-Frequency Signal):

150	160	170	180	170	160	150	160
-----	-----	-----	-----	-----	-----	-----	-----

Output (Low-Pass Filtered, Low-Frequency Signal):

155	160	170	173	170	160	157	155
-----	-----	-----	-----	-----	-----	-----	-----

As can be seen, the pixel values of the low-pass filtered version of the low frequency signal do not fluctuate much, and in fact, only a few pixels were modified at all. But if this same filter were applied to the high-frequency example signal, the following output would be provided:

Input (High-Frequency Signal):

0	150	255	150	0	255	0	150
---	-----	-----	-----	---	-----	---	-----

Output (Low-Pass Filtered, High-Frequency Signal):

75	135	185	135	135	128	135	75
----	-----	-----	-----	-----	-----	-----	----

A low-pass filter therefore has a much more significant impact on high-frequency signals than low-frequency ones.

57. A “high-pass” filter does the opposite of a low-pass filter: it allows high-frequency components to “pass” through the filter, and therefore *accentuates* changes in pixel value. One use of a high-pass filter is to perform a “sharpening” operation, in which the filter takes multiple pixel values as inputs and amplifies differences in pixel values to thereby increase rapid changes in pixel values. Gonzalez, 258 (explaining how a “highpass filter ... would enhance sharp details”). For instance, if we apply a high-pass filter (*e.g.*, [-1, 3, -1]) to the low-frequency signal (except for the left-most and right-most pixel, which have only one neighboring pixel), the following output would be provided:

Input (Low-Frequency Signal):

150	160	170	180	170	160	150	160
-----	-----	-----	-----	-----	-----	-----	-----

Output (High-Pass Filtered, Low-Frequency Signal):

130	160	170	200	170	160	130	180
-----	-----	-----	-----	-----	-----	-----	-----

As can be seen, the differences in pixel values are accentuated with the high-pass filter. Because the example high-frequency signal I provide at the beginning of this section is already extreme and would not be as helpful in demonstrating the changes

to pixel values imparted by a high-pass filter, I have used a different, slightly lower high-frequency signal as the input example to demonstrate how a high-pass filter would further accentuate the high-frequencies in its output:

Input (High-Frequency Signal):

50	150	165	150	50	175	100	150
----	-----	-----	-----	----	-----	-----	-----

Output (High-Pass Filtered, High-Frequency Signal):

0	235	195	235	0	255	0	250
---	-----	-----	-----	---	-----	---	-----

58. Finally, a “band-pass” filter has an intermediate effect in that it removes both high-frequency and low-frequency components, while preserving the intermediary-frequency components of the signal. It is commonly used, for example, in radio and television communications systems where a message is transmitted over a specific channel (*i.e.*, frequency range) and a band-pass filter is used to extract the message.

59. There are several ways to evaluate the frequency characteristics of a signal. One way to identify the fluctuations in pixel values is by using a high-pass filter, not to modify the pixel values as shown above, but simply to evaluate their frequency attributes. For instance, a simple difference operation—otherwise known as an approximation of the “first-order derivative”—between adjacent pixels can be

performed. In physics terms, this is like measuring the slope or velocity between two pixels.

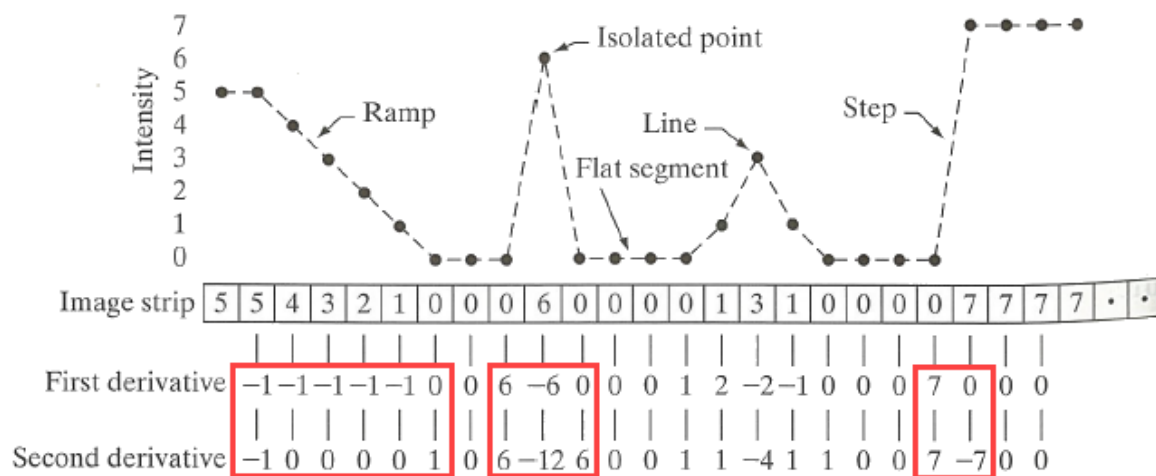
60. However, measuring the slope or velocity of change is not always the preferred approach to evaluating pixels, because it will respond to *any* changes in pixel values. For instance, if a series of pixels gradually change in value (like the low-frequency example signal above), the first-order derivative would indicate a change at every pixel. For this reason, in image processing, first-order derivatives were known to produce “thick” edges when evaluating an image because they would measure changes over a wider area. *See, e.g.,* Gonzalez, 694; Ex-1041, 154 (Fig. 5.8 noting first-order derivative will capture the entire range between points “a” and “b”).

61. Another preferred way to identify a change in pixel values is by calculating the “difference-of-difference”—otherwise known as an approximation of the “second-order derivative”—in pixel values. Such a calculation requires a minimum of three pixels (*e.g.,* p_2, p_1, p_0), and, as the name suggests, takes the difference of the differences: $((p_0 - p_1) - (p_1 - p_2)) = (p_0 - 2p_1 + p_2)$. In physics terms, this is like measuring the acceleration between pixels.

62. Because a second-order derivative responds only to *changes* in the change of pixel-values, it would not respond to gradual, consistent changes like the first-order derivative would, but would respond to rapid changes in pixel values. *See,*

e.g., Gonzalez, 693-94; Ex-1041, 154. For this reason, second-order derivatives were preferred for assessing the location of fine details in an image. *See* Gonzalez, 693-95.

63. An example of the different responses to a first- and second-order derivative is shown below:



Gonzalez, 694 (annotations added).

64. Typically, systems use either a first- or second-order derivative to assess pixel changes for simplicity. Gonzalez, 692.

3. Deblocking Filter Development

65. Prior to the '659 Patent, deblocking filters had been routinely used in video coding systems. Each typically followed the same two-step approach: (1) analyze the pixels near a block boundary to determine whether they need to be filtered; and if it is determined that filtering should be performed, (2) filter/modify the pixel values. Within this framework, prior-art deblocking filters largely differed

in three ways: (1) what conditions were used to decide whether deblocking filtering is performed; (2) how many pixels to filter as part of deblocking; and (3) the equations used to filter/modify the pixel values.

66. To illustrate, I have provided below a walkthrough of the International Telecommunication Union (“ITU”) standards that have been released over the last two decades, but many other deblocking filters existed in the prior art.

a. H.263

67. In 1998, the International Telecommunication Union’s Telecommunication Standardization Sector (“ITU-T”) released a revised version of the H.263 standard that, for the first time for the ITU, provided an optional deblocking filter as part of its standardized video decoder. Ex-1035, 4 (§ 3.4.8). The standard called for the deblocking filter to be implemented in-loop, meaning that the filter is applied during the image-reconstruction process so that the reference images used to predict current images would have already had the filter applied. *Id.*, 4, 81 (J.3); Ex-1039, 1.

68. Deblocking filters could also be implemented in “post-processing,” which means the deblocking filter is applied *after* all of the decoding operations have been performed but before the video image is displayed on screen. Ex-1033, 188; Ex-1039, 211. In other words, in post-processing deblocking, the reference images used to predict current images would maintain the blocking artifacts during

prediction, and the artifacts would only be eliminated later. However, it has been preferred to keep deblocking filters “in-loop” because that produced more accurate predictions during encoding and decoding. Ex-1033, 188.

69. In H.263, a decision is first made whether to apply the deblocking filter at all by checking the type of macroblock to which the block belonged. Ex-1035, 82. If the correct conditions are met, then a deblocking filter would be applied. *Id.* Using A and B to represent pixels in one block, and C and D to represent pixels in a neighboring block, the H.263 standard provided the equations that would be applied to perform deblocking. *Id.*

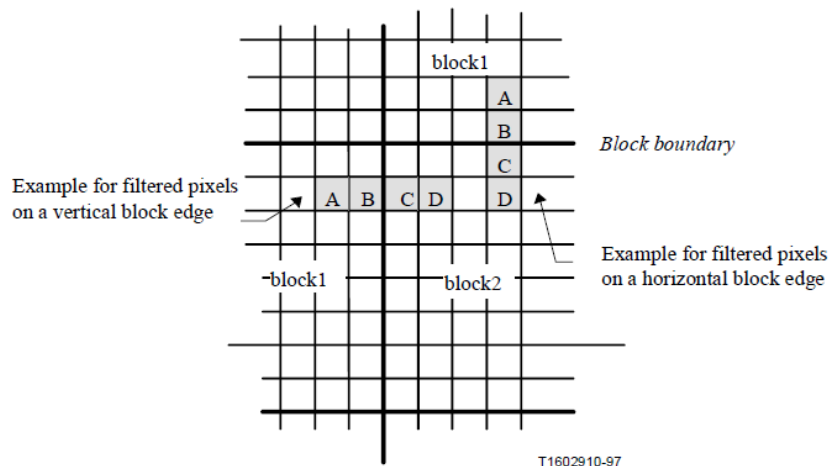


Figure J.1/H.263 – Examples of positions of filtered pixels

Id.

70. To perform deblocking, first, a preliminary “offset” value (d) is calculated by weighting the pixels near the block boundary using the equation:

$$d = \frac{A - 4B + 4C - D}{8}$$

Id. This preliminary offset value is then processed through a “ramp” function that linearly adjusts the offset depending on the intended “strength” of the ramp. *Id.* The “strength” of the ramp is, in turn, controlled by a “quantization parameter,” *id.*, 83, which characterizes how much the block was quantized (or compressed) during the encoding process. In other words, the more the block was quantized during encoding, the stronger the filter would be. The ramp function would then output a modified offset value “d1.” *Id.*, 82.

71. This “d1” offset value would then be used to modify the pixel values of the pixels closest to the block boundary. Specifically, the d1 value would be added to the value of pixel B to produce a new pixel value B1 (or $B1=B+d1$), and subtracted from the value of pixel C to produce a new pixel value C1 (or $C1=C-d1$). *Id.* Applying the offset to pixels B and C blends the transition between the two blocks. However, to ensure the new pixel values for B1 and C1 would stay within the range of 0-255 (the only valid pixel values when using 8 bits), the output of the operation $B+d1$ and $C-d1$ calculations would be “clipped.” *Id.*, 81-82.

72. The clipping function serves as a limit on an output, defining the minimum and maximum values that can be output. For instance, because 8-bit pixels must have a value between 0 and 255, if $B+d1$ produced a value of 258, the clipping

function would instead output the maximum value of 255. Or if C–d1 produced a pixel value of -4, the clipping function would output the minimum value of 0 instead. *See id.* The clipping function just ensures the final output stays within a specified window of possible pixel values.

73. Then, a second-offset value “d2” would be calculated for pixels A and D using the equation:

$$d2 = \frac{A - D}{4}$$

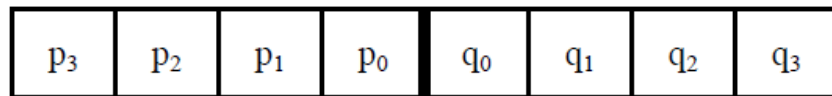
Id., 94. However, the value of d2 is also “clipped” to be, at most, one-half of d1. *Id.*; *see also id.*, 95 (“the function clipd1(x, lim) clips x to the range $\pm \text{abs}(\text{lim})$.”). The value of d2 would then be subtracted from A to produce a new pixel value A1 (or $A1 = A - d2$), and added to D to produce new pixel value D1 (or $D1 = D + d2$). Pixels A and D would therefore be modified less than pixels B and C, because they are further from the block boundary.

b. H.264

74. After the release of H.263, ITU-T formed a joint venture with the Motion Picture Expert Group (“MPEG”)—another standards-setting organization for multimedia coding—called the “Joint Video Team” (“JVT”) to develop its successor. When ITU-T/MPEG released the H.264 standard in 2003, the deblocking filter was no longer optional, and its operation was more complex.

75. First, the H.264 standard² outlines a process to evaluate the boundary strength (characterized as variable “bS”) between two neighboring blocks, and assign the boundary strength a number between 0 and 4. Ex-1037, 204-05. A bS of zero means no filtering is to be applied, and the higher the bS number, the more filtering that will be applied. *Id.*, 206 (Equation 8-468).

76. When the bS is greater than 0 but less than 4, the deblocking filter will evaluate additional conditions to determine whether to perform deblocking. Specifically, three calculations are performed— $|p_0 - q_0|$ (the pixel difference across the boundary), $|p_1 - p_0|$ (the pixel difference of the pixels nearest the boundary in block p), and $|q_1 - q_0|$ (the pixel difference of the pixels nearest the boundary in block q)—which are each compared to a specific “threshold” value that is based on the quantization parameters for the two blocks being evaluated. *Id.*, 205-06 (Equation 8-468).



Id. 202. In other words, the H.264 standard checks the difference in pixel values across the block boundary ($p_0 - q_0$) and the difference in pixel values of the pixels

² I will refer to the March 2010 version of the H.264 standard, which is prior to the earliest-possible priority date of the '659 Patent.

nearest to the block boundary on each respective side (p_1-p_0 and q_1-q_0), and compares each to a threshold to determine whether to perform deblocking.

77. Then, intermediate “threshold variables a_p and a_q are derived” using the following equations:

$$\begin{aligned}a_p &= \text{Abs}(p_2 - p_0) \\a_q &= \text{Abs}(q_2 - q_0)\end{aligned}$$

Id., 207 (equations 8-471, 8-472). In other words, H.264 calculates the difference between the pixel closest to the block boundary and third-closest pixel to the block boundary for each of the p and q blocks, separately.

78. The H.264 standard then provides the filtering that is to be applied. Specifically, the offset is calculated using the equation:

$$\Delta_{p_0,q_0} = \text{Clip3}(-t_C, t_C, (((q_0-p_0) \ll 2) + (p_1-q_1) + 4) \gg 3))$$

Id., 207.

79. This equation is written in what is informally called “bit-shift” form. This equation has two particular “bit-shift” operations: a left ($\ll$) shift, and a right ($\gg$) shift. These operations are approximations of the mathematical processes of multiplication and division, respectively. However, they are performed on the actual binary bits of data stored by the computer, which uses less computational resources than actually performing multiplication or division. And because the “bit-shift”

operations operate on binary bits of data, they can be implemented in hardware configurations as well as software.

80. Bit-shift operations take advantage of the way numbers are represented and related to one another in binary form. Below is a table of example numerical values and their binary counterparts:

Number	Binary Representation
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000

81. What is notable about this representation is that every number representing a power of 2 (e.g., $1=2^0$, $2=2^1$, $4=2^2$, $8=2^3$) has a single “1” in its binary form that shifts one binary unit to the left as the value increases, and the numbers between them follow a pattern of alternating between “0” and “1” to the right of the leading “1.” It is this relationship that bit-shift operations take advantage of.

82. A “left-shift” operation—which is effectively a multiplication operation—means that each binary value is shifted to the left, and a zero is appended to the right. Thus, a left shift of $\ll n$ would shift all bits “n” to the left and add “n” zeros to the right. This equates to multiplying the number by 2^n . Some examples are provided below.

Input Number	Input Binary	Shift Operation	Math Form	Output Binary	Output Number
1	0001	$1 \ll 1$	$1 * 2^1$	0010	2
1	0001	$1 \ll 2$	$1 * 2^2$	0100	4
1	0001	$1 \ll 3$	$1 * 2^3$	1000	8
2	0010	$2 \ll 1$	$2 * 2^1$	0100	4
2	0010	$2 \ll 2$	$2 * 2^2$	1000	8
3	0011	$3 \ll 1$	$3 * 2^1$	0110	6
4	0100	$4 \ll 1$	$4 * 2^1$	1000	8

83. A “right-shift” operation does something similar, but in the opposite direction, and represents an integer-approximation of division by 2^n . Some examples of right-shift operations are shown below:

Input Number	Input Binary	Shift Operation	Math Form	Output Binary	Output Number
8	1000	$8 \gg 3$	$8 / 2^3$	0001	1
8	1000	$8 \gg 2$	$8 / 2^2$	0010	2

8	1000	$8 \gg 1$	$8/2^1$	0100	4
7	0111	$7 \gg 2$	$7/2^2$	0001	1
7	0111	$7 \gg 1$	$7/2^1$	0011	3
6	0110	$6 \gg 1$	$6/2^1$	0011	3

There are a few additional noteworthy elements about the right-shift operation. First, the output of a right-shift operation is always an integer, even when division would produce a decimal or fractional remainder. Second, the output is always rounded down to the next-lowest integer value, not the integer nearest the decimal or fractional remainder. For instance, when taking $7 \gg 2$, the output is 1; it is not 1.75 (the true value of $7/2^2$ or $7/4$), or 2 (1.75 rounded to its nearest integer).

84. To round to the nearest integer, a rounding value is added to the shift operation equivalent to half the value of the denominator (or $2^n/2$). For instance, because “ $\gg 3$ ” equates to dividing by eight, to ensure each output is rounded to the nearest integer, a value of 4 should be added to each input number. And because “ $\gg 2$ ” equates to dividing by 4, a value of 2 should be added to each input number. I have rearranged the examples above to show how this would apply in practice.

Right-Shift Operation	Input Number	Rounding Value	New Input Number	Input Binary	Output Binary	Output Number
$\gg 3$	8	+4	12	1100	0001	1
$\gg 2$	8	+2	10	1010	0010	2

>>2	7	+2	9	1001	0010	2
>>1	7	+1	8	1000	0100	4

Even when the rounding value is added to, e.g., an input of “8,” the output remains the same as if the rounding value had not been added. That is because the extra right-facing bits that are added to the binary value of 8 (1000) for 12 (1100) and 10 (1010) are ultimately eliminated by the right-shift operation anyway. The output, however, *does* change when the rounding value is added to, e.g., an input of “7” (0111) because there is a new left-facing bit that is not eliminated by the right-shift operation (e.g., for 9, 1001; and 8, 1000) by the right-shift operation. Adding the rounding value to odd numbers or even numbers not divisible by a power of 2 ensures the output number is rounded to the nearest integer value, rather than always rounded down.

85. Returning to the H.264 standard, then, the offset equation $((q_0 - p_0) \ll 2) + (p_1 - q_1) + 4 \gg 3$ can be rewritten in division form by applying each bit-shift operation and dropping the rounding value (with the understanding that the output of the division form would be rounded to the nearest integer as well):

$$\frac{4(q_0 - p_0) + (p_1 - q_1)}{8}$$

86. According to the H.264 standard, this offset (subject to clipping) is then added to the pixel value for p0 and subtracted from the value q0 to produce new pixel values for the pixels closest to the block boundary. *Id.*, 207.

87. After filtering the pixels closest to the block boundary, H.264 discloses that a decision is made whether to filter the pixel second closest to the block boundary in each block, separately. *Id.*, 207-208. Specifically, using the block-specific evaluations previously calculated ($a_p = \text{Abs}(p_2 - p_0)$; $a_q = \text{Abs}(q_2 - q_0)$), if a specific flag (chromaStyleFilteringFlag) is set, and a_p is less than a threshold (β), pixel p1 is filtered; otherwise, pixel p1 is not filtered. *Id.* And if a_q is less than a threshold (β) (and the same flag is set), pixel q1 is filtered; otherwise, pixel q1 is not filtered. *Id.*, 208.

88. If a decision is made to filter the p1 and/or q1 pixels, p1 and q1 are modified using the following offset equations, which are added to p1 or q1, respectively:

$$\Delta_{p1} = \text{Clip3}(-tC_0, tC_0, (p_2 + ((p_0 + q_0 + 1) \gg 1) - (p_1 \ll 1)) \gg 1))$$

$$\Delta_{q1} = \text{Clip3}(-tC_0, tC_0, (q_2 + ((p_0 + q_0 + 1) \gg 1) - (q_1 \ll 1)) \gg 1))$$

Id., 208. Like $\Delta_{p0,q0}$, the offset equations can also be rewritten in division form (subject to rounding and clipping):

$$\Delta_{p1} = \frac{2(p_2) - 4(p_1) + p_0 + q_0}{4}$$

$$\Delta_{q1} = \frac{2(q2) - 4(q1) + q0 + p0}{4}$$

89. Finally, if the bS value is equal to 4, then a different set of offset equations are applied for deblocking, which can modify up to three pixels on each side of the boundary. *Id.*, 208-09.

c. Draft H.265 (B205)

90. In 2010, ITU and MPEG formed another joint-venture called the “Joint Collaborative Team on Video Coding” (“JCT-VC”) which was tasked with “develop[ing] a new generation video coding standard that will further reduce by 50% the data rate needed for high quality video coding.” Ex-1042. This new standard would be called the “High Efficiency Video Coding (HEVC)” standard. *Id.*

91. Prior to the earliest-possible effective filing date of the ’659 Patent (as the ’659 Patent explains), the JCT-VC released a document numbered JCTVC-B205, titled “Test Model under Consideration.” Ex-1018, 1. The draft was intended to capture “a coding efficiency close to the best performing proposals” and “a complexity point that is close to that of the lowest complexity submissions shown to provide substantial improvement of coding efficiency” that had been submitted to date. *Id.*

92. The test model at the time included a deblocking filter that worked a bit differently than the one in H.264. *See id.*, 127-32. Specifically, rather than calculate

an initial “boundary strength” value like H.264, B205 proposed a deblocking filter that would check one of a series of possible conditions *plus* a filter-decision equation to determine whether deblocking should be performed. The filter-decision equation is provided below:

$$|p_{22}-2*p_{12}+p_{02}| + |q_{22}-2*q_{12}+q_{02}| + |p_{25}-2*p_{15}+p_{05}| + |q_{25}-2*q_{15}+q_{05}| < \beta$$

Id., 130. In simpler terms, this equation took four second-order “difference-of-difference” measurements—two from each side of the block boundary using two different lines of pixels—summed them and compared them to a threshold value (β) that was based on the quantization parameters for the subject blocks.

p3 ₀	p2 ₀	p1 ₀	p0 ₀	q0 ₀	q1 ₀	q2 ₀	q3 ₀
p3 ₁	p2 ₁	p1 ₁	p0 ₁	q0 ₁	q1 ₁	q2 ₁	q3 ₁
p3 ₂	p2₂	p1₂	p0₂	q0₂	q1₂	q2₂	q3 ₂
A p3 ₃	p2 ₃	p1 ₃	p0 ₃	q0 ₃	q1 ₃	q2 ₃	q3 ₃
P3 ₄	p2 ₄	p1 ₄	p0 ₄	q0 ₄	q1 ₄	q2 ₄	q3 ₄
P3 ₅	p2₅	p1₅	p0₅	q0₅	q1₅	q2₅	q3 ₅
P3 ₆	p2 ₆	p1 ₆	p0 ₆	q0 ₆	q1 ₆	q2 ₆	q3 ₆
P3 ₇	p2 ₇	p1 ₇	p0 ₇	q0 ₇	q1 ₇	q2 ₇	q3 ₇

Id. (annotated). If the conditions outlined above were met, it was then determined whether to apply a “strong” or “weak” filter depending on an additional set of conditions.

93. If weak filtering was applied, two pixels on each side of the boundary would be modified (p0/q0 and p1/q1), and B205 provided the offset equation (Δ) to

calculate the offset, and then applied the full offset to pixels p0 and q0, and half the offset to pixels p1 and q1:

$$\begin{aligned}\Delta &= \text{Clip}(-t_c, t_c, (13*(q0_i - p0_i) + 4*(q1_i - p1_i) - 5*(q2_i - p2_i) + 16) \gg 5)) \\ p0_i &= \text{Clip}_{0-255}(p0_i + \Delta) \\ q0_i &= \text{Clip}_{0-255}(q0_i - \Delta) \\ p1_i &= \text{Clip}_{0-255}(p1_i + \Delta/2) \\ q1_i &= \text{Clip}_{0-255}(q1_i - \Delta/2)\end{aligned}$$

Id., 131.

94. If strong filtering was applied, three pixels on each side of the boundary would be modified (p_0/q_0 , p_1/q_1 , and p_2/q_2), and B205 provided those equations as well (calculating new pixel values instead of offsets):

$$\begin{aligned}p0_0 &= \text{Clip}_{0-255}((p2_i + 2*p1_i + 2*p0_i + 2*q0_i + q1_i + 4) \gg 3) \\ q0_0 &= \text{Clip}_{0-255}((p1_i + 2*p0_i + 2*q0_i + 2*q1_i + q2_i + 4) \gg 3) \\ p1_0 &= \text{Clip}_{0-255}((p2_i + p1_i + p0_i + q0_i + 2) \gg 2) \\ q1_0 &= \text{Clip}_{0-255}((p0_i + q0_i + q1_i + q2_i + 2) \gg 2) \\ p2_0 &= \text{Clip}_{0-255}((2*p3_i + 3*p2_i + p1_i + p0_i + q0_i + 4) \gg 3) \\ q2_0 &= \text{Clip}_{0-255}((p0_i + q0_i + q1_i + 3*q2_i + 2*q3_i + 4) \gg 3)\end{aligned}$$

Id.

VI. '659 PATENT

95. Application No. 15/639,229 was filed on June 30, 2017. It claims priority, through a series of continuations, to provisional application No. 61/432,751, filed on January 14, 2011.

96. I therefore understand that for purposes of comparing the '659 Patent to the prior art, the earliest possible priority date is January 14, 2011.

97. I have reviewed the '659 Patent and provide the following summary.

A. Specification

98. The '659 Patent begins by providing an overview of the deblocking filter in both the H.264 standard³, as well as the proposed deblocking filter for the then-current draft of the HEVC standard, as outlined in B205. Ex-1001, 1:34-2:17, 2:19-3:33.

99. The then-current HEVC draft proposed the following filter-decision equation to determine whether to perform deblocking:

$$d = |p2_2 - 2p1_2 + p0_2| + |q2_2 - 2q1_2 + q0_2| \\ + |p2_5 - 2p1_5 + p0_5| + |q2_5 - 2q1_5 + q0_5| < \beta$$

100. Using p_{j_i} and q_{j_i} for reference, j represents the distance of the pixel from the block boundary (e.g., “p0” and “q0” being closest), and i represents the line or column of pixels within the block, Ex-1001, 2:26-40, as shown below.

³ The '659 Patent directs the reader to an article titled *Adaptive Deblocking Filter*, IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, no. 7 (July 2003) by List et al. for more information on H.264. Ex-1001, 2:14-17. This is the same List article discussed in this Declaration.

p3 ₀ p2 ₀ p1 ₀ p0 ₀	q0 ₀ q1 ₀ q2 ₀ q3 ₀
p3 ₁ p2 ₁ p1 ₁ p0 ₁	q0 ₁ q1 ₁ q2 ₁ q3 ₁
p3 ₂ p2 ₂ p1 ₂ p0 ₂	q0 ₂ q1 ₂ q2 ₂ q3 ₂
A p3 ₃ p2 ₃ p1 ₃ p0 ₃	q0 ₃ q1 ₃ q2 ₃ q3 ₃ B
P3 ₄ p2 ₄ p1 ₄ p0 ₄	q0 ₄ q1 ₄ q2 ₄ q3 ₄
P3 ₅ p2 ₅ p1 ₅ p0 ₅	q0 ₅ q1 ₅ q2 ₅ q3 ₅
P3 ₆ p2 ₆ p1 ₆ p0 ₆	q0 ₆ q1 ₆ q2 ₆ q3 ₆
P3 ₇ p2 ₇ p1 ₇ p0 ₇	q0 ₇ q1 ₇ q2 ₇ q3 ₇

101. The '659 Patent discloses that “β” is a threshold that depends on a characteristic (the “quantization parameter” or “QP”) of the blocks as a whole. *Id.*

102. If the condition ($d < \beta$) is met, one of two deblocking filters would be applied—a “strong” or “weak” filter—depending on a set of additional conditions. Ex-1001, 2:41-48. If a weak filter is used, the following offset equation would be applied:

$$\Delta = \text{Clip}(-t_c, t_c, (13(q0_i - p0_i) + 4(q1_i - p1_i) - 5(q2_i - p2_i) + 16) \gg 5))$$

Id., 2:58-67. Notably, this equation has two parts: (1) an equation (in red) to calculate the value of the offset (Δ); and (2) a “clipping” function (in black; discussed in the Technology Background section above) that limits the offset to stay between a minimum (“ $-t_c$ ”) and maximum (“ t_c ”) value. *Id.* The equation in red is provided in “bit-shift” form (discussed in the Technology Background section above) for hardware implementations. The equation above, in “bit-shift” form can be rewritten in typical division form by applying the bit-shift operation and dropping the rounding value (with the understanding that the output of the division form would

be rounded to the nearest integer as well for the reasons I discuss above). *See* Section

V.B.3.b. Here, the “>>5” term is changed out for a division by 32 (i.e., 2^5):

$$\frac{13(q_0 - p_0) + 4(q_1 - p_1) - 5(q_2 - p_2)}{32}$$

103. The result of this equation, Δ or the “offset,” is then added to pixels p_0 and p_1 and subtracted from pixels q_0 and q_1 , whereas as shown below, half of the offset is added or subtracted to pixels p_1 and q_1 respectively. The proposed “weak” filter for the then-current HEVC draft, therefore, would modify two pixels on each side of the block boundary.

$$\begin{aligned} p0_i &= Clip_{0-255}(p0_i + \Delta) \\ p1_i &= Clip_{0-255}\left(p1_i + \frac{\Delta}{2}\right) \\ q0_i &= Clip_{0-255}(q0_i - \Delta) \\ q1_i &= Clip_{0-255}\left(q1_i - \frac{\Delta}{2}\right) \end{aligned}$$

Ex-1001, 2:58-3:3.

104. Before the modified pixel value is derived, the result of adding (or subtracting) the offset (or half of the offset) to the original pixel value is clipped to stay within the bounds of 0 to 255, which as discussed above, is the 8-bit value given to each pixel.

105. The then-current proposed deblocking filter for HEVC had some problems according to the ’659 Patent. First, “the weak filter does not have good low-pass characteristics” because “the first and the second pixels from the block border are modified by adding or subtracting Δ and $\Delta/2$ respectively. Such a filter

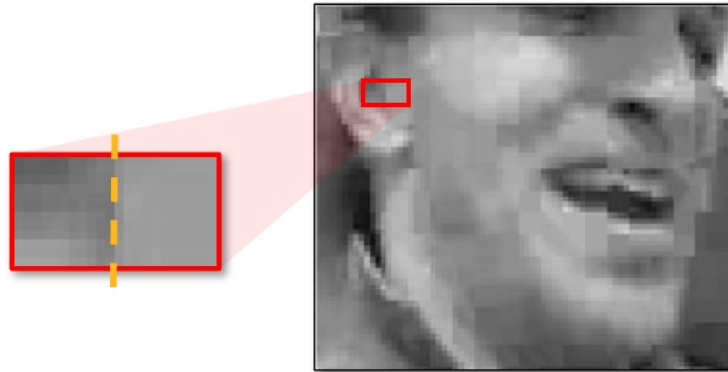
cannot remove high frequencies that can appear near the block border and cannot remove ringing.” Ex-1001, 3:18-24. Ringing is a known aspect of filter design, and can occur when a low-pass filter first overshoots its target output and then subsequently overcompensates and thus undershoots its target. *See, e.g.*, Ex-1073, 3. Over time, those in the art had found empirically that using six pixels or less in an offset equation tended to avoid significant ringing.

106. Second, “the frequency characteristics of the filter in HEVC show slight amplifications of higher frequencies when filtering the second pixel ($p1_i$ and $q1_i$) from the block boundary, which can result in worse rate-distortion (RD) performance of the coder.” Ex-1001, 3:24-28. Distortion corresponds to the amount of loss of video quality, while rate corresponds to the amount of data required to encode a video. *See generally* Ex-1019. These two factors can be optimized against each other, for example minimizing distortion for a desired rate, or minimizing rate for a given amount of distortion.

107. Third, “the offset for pixel $p1_i$ is equal to the magnitude to the offset of pixel $q1_i$ although the structure of the signal can be different on different sides of the block boundary.” Ex-1001, 3:28-31.

108. The ’659 Patent concludes that “the HEVC filter does not adapt well to the local structure of the side of the block boundary.” Ex-1001, 3:31-33. For instance, sometimes, a real detail in an image may align with a block boundary,

which could result in filtering of that detail. Lack of adaption to the local structure is shown in this example.



109. In the image above, there is a crisp line at the man's ear canal, which would cause the pixel values near the ear canal to increase in value as they get darker toward the ear canal. As one can see in the left image, the increase in darkness of pixels is gradual, such that it forms a ramp. According to the '659 Patent, this change in pixel values could be mistaken for a blocking artifact and removed by the deblocking filter by "smoothing" out the pixel values near the crease. Ex-1001, 13:11-49. In other words, it could remove the natural ramp in pixel values that appears as the ear canal is approached.

110. To remedy the perceived problem, the '659 Patent proposes various modifications to the deblocking filter of the then-current draft HEVC standard. The goal of the '659 Patent is to create a more efficient deblocking filter that has better filter characteristics. For example, the deblocking filter "is configured not to modify

the pixel values over a block boundary if these pixel values form a ramp but smooth the pixel values if they instead are in the form of a step.” Ex-1001, 8:57-60. The ’659 Patent touts that “[t]his implies that the deblocking filter will have good lowpass characteristics and can remove high frequencies that can appear near the block boundary.” *Id.*, 8:60-63. For example, considering the image above, the filter proposed in the ’659 Patent will preserve the ramp of pixel values so that the man’s ear canal is visible and defined, while at the same time being able to filter out (or smooth by bringing the pixel values closer together) large increases between neighboring pixels, or a “step.”

111. To achieve this goal, the ’659 Patent proposes modifying the “offset equations” that are used to calculate how much a pixel value should be modified to smooth out any steps that the deblocking filter is removing. The ’659 Patent proposes different offset equations, depending on what pixel is being filtered.

112. When referring to pixels, different notations to refer to pixels in each block are used. The ’659 Patent adopts the following pixel notation, shown in Figure 2A (annotated):

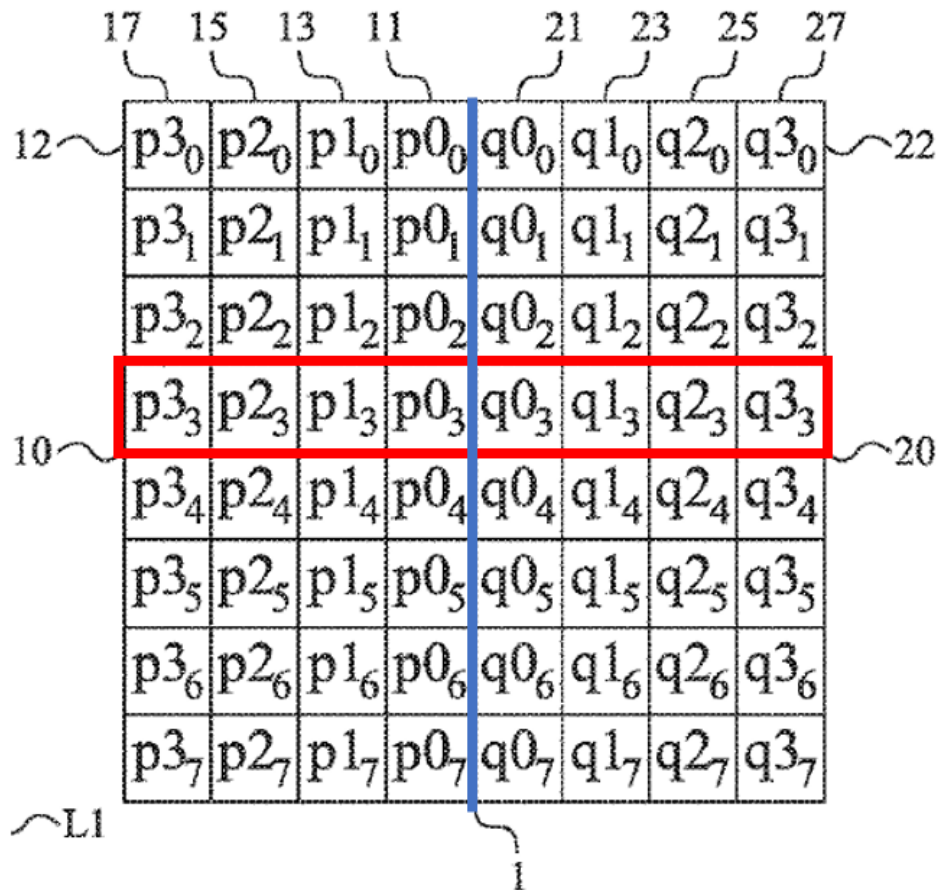


Fig. 2A

113. Figure 2A above has two neighboring blocks of pixels, pixel block 10 and pixel block 20, with a block boundary separating them. Ex-1001, 9:45-49. The block boundary is indicated by the solid blue line added above.

114. If one looks at the left block, 10, first, there are eight rows of pixels. Focusing on row 3 (i.e., $p3, p2, p1, p0$), it has four pixel values where (ignoring the “3” notation for the row), “p0 denotes the pixel value of the pixel 11 closest to the block boundary,” “p1 denotes the pixel value of the pixel 13 next closest to the block boundary 1,” etc. Ex-1001, 9:67-10:4. In other words, p0, p1, p2, and p3 are

consecutive pixel values, where the notation pX increases as one gets further from the block boundary.

115. The “p” block of pixels has a neighboring block, 20, that identifies pixels using a “q” notation. Similar to the “p” block of pixels, “q0 denotes the pixel value of the pixel 21 closest to the block boundary 1 in a corresponding or opposing line 22 of pixels,” “q1 denotes the pixel value of the pixel 23 next closest to the block boundary 1 in the corresponding line of 22 of pixels,” etc. Ex-1001, 10:4-9.

116. Turning back to the offset equations proposed by the ’659 Patent, it proposes modifying the offset of the pixels closest to the block boundary—p0 and q0—using a first offset equation based on

$$(9(q0_i - p0_i) + 3(q1_i - p1_i) + 8) \gg 4)$$

Ex-1001, claim 1.

117. A table of the first offset equation from the ’659 Patent and the then-current draft of the HEVC standard is below. In the table below, the top equation is in “bit-shift” form, while the bottom equation is shown in division form. The ’659 Patent’s equation is also multiplied by 2/2 (i.e., 1) to arrive at its mathematical equivalent with a divisor of 32 (i.e., 2⁵) instead of 16 (i.e., 2⁴).

Draft HEVC Standard (<i>Id.</i> , 2:62-63)	’659 Patent (<i>Id.</i> , , 12:57-13:7)
13(q0-p0)+4(q1-p1)-5(q2-p2)+16>>5	9(q0-p0)-3(q1-p1)+8>>4

$\frac{13(q_0 - p_0) + 4(q_1 - p_1) - 5(q_2 - p_2)}{32}$	$\frac{18(q_0 - p_0) - 6(q_1 - p_1)}{32}$
--	---

118. As one can see from the table above, the '659 Patent proposes multiple modifications to the then-current draft HEVC standard. First, the '659 Patent proposes considering four pixel values (two on each side of the block boundary)—pixels p_0 , p_1 from a block and pixels q_0 , q_1 from the neighboring block—instead of also looking at pixels p_2 and q_2 (colored purple above) as in the then-current draft HEVC standard. Second, the '659 Patent proposes modifying the deblocking coefficients for the p_0/q_0 and p_1/q_1 terms (colored blue above). For the term $(q_0 - p_0)$, the '659 Patent proposes a deblocking coefficient of 9 instead of 13 used in the then-current draft HEVC standard. For the term $(q_1 - p_1)$, the '659 Patent proposes a deblocking coefficient of 3 instead of 4 used in the then-current draft HEVC standard.

119. The result of this first offset equation is added to the original pixel value of pixel p_0 to arrive at the “modified” pixel value, which the '659 Patent denotes as p_0' . Conversely, the result of the first offset equation is subtracted from the original pixel value q_0 to arrive at the modified pixel value, which the '659 Patent denotes as q_0' . Ex-1001, 10:38-49.

120. The '659 Patent proposes different equations for the next closest pixels to the block boundary for both the p block of pixels and neighboring q block of pixels. The then-current draft of the HEVC standard proposes modifying pixel values p1 and q1 by dividing the result of the first offset equation by 2. Ex-1001, 2:56-3:3. The '659 Patent proposes a different offset equation, where the pixel value for the second closest pixel to the block boundary for each respective p block of pixels and q block of pixels (i.e., p2 and q2) is also considered.

121. Specifically, the '659 Patent proposes the second offset equation below for pixel p1—the pixel next closest to the block boundary:

$$p1' = p1 + \frac{p2 + p0 - 2 \times p1 + 2\Delta}{4}$$

122. In the second offset equation (indicated in red above), the pixel value for p2 is brought in, along with the pixel value of p1 (the pixel to be modified). The result from the first offset equation (i.e., Δ) is also considered. The result of the second offset equation is added to the pixel value of p1, which results in a modified value of pixel p1 (i.e., p1'). Ex-1001, 14:23-47.

123. Similarly, the '659 Patent proposes a third offset equation below (indicated in red) for pixel q1—the pixel next closest to the block boundary in the neighboring block of pixels. Like the second offset equation, it calculates a third offset considering consecutive pixel values, q2, q1, and q0, as well as the first offset

(i.e., Δ). The result of the third offset equation is subtracted from the pixel value q_1 , which results in a modified pixel value of q_1 (i.e., q_1'). Ex-1001, 14:23-47.

$$q_1' = q_1 - \frac{q_2 + q_0 - 2 \times q_1 + 2\Delta}{4}$$

124. Variations of the second and third offset equations are also disclosed in the '659 Patent. *E.g., id.*, 14:48-65. These variations are the mathematical equivalent to the embodiment discussed above. *Id.*, 14:65-66.

B. Prosecution History

125. I have reviewed the '659 prosecution history and provide the following summary.

126. The '229 Application was filed on June 30, 2017 and included 18 claims. The original claims were directed to a method of reducing blocking artifacts at a block boundary between neighboring blocks of pixels by calculating at least a first offset equation (considering claim 1) and modifying pixels p_0 and q_0 by adding and subtracting the first offset respectively. Ex-1002, 53-60. The Examiner issued a first non-final office action on January 26, 2018, which rejected all 18 claims on two grounds. *Id.*, 91-98. Neither of the Examiner's rejections affect my analysis of the Challenged Claims.

127. First, the Examiner issued a double patenting rejection over claims 1-23 of U.S. Patent No. 8,526,509, which is the oldest non-provisional application in the patent family. *Id.*, 93-94.

128. Second, the Examiner rejected all 18 claims under 35 U.S.C. § 101 because claims 1 and 10 “merely manipulate[] data without ever producing a useful, concrete and tangible result.” *Id.*, 94-96.

129. The Examiner did not issue any rejections over the prior art. Specifically, the Examiner found that the closest prior art—Hsu⁴ and Ugur⁵—failed to anticipate or render obvious two limitations of independent claims 1 and 10: (1) the first offset equation recited in claims 1 and 10 and (2) modifying the pixel values p_0 and q_0 . *Id.*, 96-97.

130. To overcome the Examiner’s rejections, Applicant submitted a terminal disclaimer. *Id.*, 121. Applicant also requested reconsideration of the Examiner’s

⁴ Chung-Lung Hsu, et al., *A Fast-Deblocking Boundary-strength Based Architecture Design of Deblocking Filter in H.264/AVC Applications*, Journal of Signal Processing Systems for Signal, Image, and Video Technology (2007). Ex-1039.

⁵ K. Ugur, et al., *High Performance, Low Complexity Video Coding and the Emerging HEVC Standard*, IEEE Transactions on Circuits and Systems for Video Technology at 1688-97 (2010). Ex-1059.

Section 101 rejection, arguing that the claimed features enable reducing blocking artifacts at a block boundary of neighboring pixel blocks, while having good low-pass characteristics, and improve upon existing technology. *Id.*, 121-23. Applicant also corrected non-substantive typographical errors in claims 4, 15, and 18. A notice of allowance issued on July 23, 2018. *Id.*, 133-37.

VII. PRIOR ART

A. Fu (Ex-1021 (original); Ex-1022 (translation))

131. Fu is an International Patent Application filed on August 21, 2008 and published on March 19, 2009. Because Fu was published (March 19, 2009) more than one year before the earliest-possible priority date of the '659 Patent (January 14, 2011), I understand that Fu constitutes prior art under at least §102(b). Fu was not cited during prosecution. Ex-1001, 1-2.

132. Fu teaches an offset filter that is applicable in “the field of image processing, and particularly to a method and device for acquiring block boundary strength and reducing block distortion.” Fu, 1:10-15. Fu calls out an issue with block distortion that “is caused by quantization error from block processing during quantization.” Fu, 1:20-22.

133. Fu recognizes two primary forms of block distortion—trapezoidal noise and lattice noise. Fu, 1:20-33. “[T]rapezoidal noise appears at the strong edge of the image.” *Id.* Specifically,

when performing discrete cosine transform of the image to acquire the transform coefficient of each frequency point, many high frequency coefficients are quantized to zero. Hence, the strong edge of the image cannot be fully digitized within the transform domain, and because of the block processing of the image, the continuity at the strong edge crossing the block boundary cannot be guaranteed. Hence, there is jagged noise at the strong edge of the image, causing visually unnatural data block edges called trapezoidal noise.

Fu, 1:20-29.

134. Fu teaches that “[l]attice noise appears in the flat region of the image.”

Specifically:

In the flat region of the image, where luma increases or decreases, the direct current (DC) coefficient of the transform domain may cross the adjacent quantization level detection threshold due to rounding during quantization rounding, resulting in sudden changes in luma in two adjacent blocks in the reconstructed image, visually appearing as sheet-like contours called lattice noise.

Id., 1:29-33.

135. To overcome this noise issue, Fu proposes a specific method for reducing block distortion. *Id.*, 1:34-38.

136. Fu’s proposes the following steps for its method of reducing block distortion. In step 1, Fu teaches “[u]sing the interpolation of the actual pixels on both sides of the present block boundary [to] calculate the flatness of the image region where the present block boundary is located, expressing the flatness of the region with the number of similarity pairs around the block boundary.” Fu, 2:1-3.

Fu notes that “[i]f there are many similarity pairs, the flatness of the image region where the present block boundary is located is considered to be relatively great.”

Fu, 2:3-12.

137. In step two, Fu teaches “[d]etermin[ing] the filtering strength of the present block boundary based on the specific flatness. The specific method is as follows: determine the value of the block boundary filtering strength (BS) based on the macroblock type and region flatness” using the following steps:

first, obtain BS based on the determined region flatness, specifically as follows:

BS is equal to 0 when RegionFlatness is less than 4;

when RegionFlatness is greater than 4, determine whether the horizontal pixel scale (horizontal_size) is greater than 500:

BS is equal to 1 if horizontal_size is greater than 500 and RegionFlatness is less than 12, otherwise BS is equal to 2;

BS is equal to 1 if horizontal_size is less than 500 and RegionFlatness is less than 16; otherwise, BS is equal to 2.

Fu, 4:4-14.

138. The BS can be adjusted according to the macroblock type where one “determine[s] whether one or both of the two 8x8 blocks on both sides of the block boundary belong to the intraframe prediction macroblock. If so, add 1 to BS, otherwise, do not change BS.” Fu, 4:14-18.

139. Fu, like the ’659 Patent, is concerned with filtering steps at the block boundary, not ramps, by calculating the region flatness and using that calculation as the basis for its filtering strength.

140. In step three, “[t]hree levels of BS values may be obtained from Step 2, while this step determines three different filtering processes based on the BS values of different levels, so as to filter each row or column in the block accordingly.” Fu, 4:21-23.

141. Fu discloses a filtering process for pixels p0, p1, p2, q0, q1, and q2, as shown in Figure 2 below:

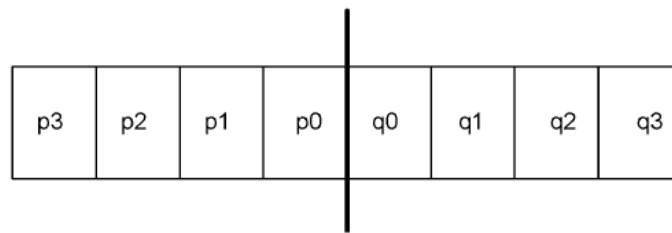


Figure 2

142. Fu, therefore, uses the same pixel naming convention as the '659 Patent, where pixel p0 is the closest pixel to the block boundary (solid black line) in the p block of pixels, p1 is the next closest pixel to the block boundary in the p block of pixels, and p2 is the second next closest pixel to the block boundary in the p block of pixels.

143. In the neighboring block, or q block, pixel q0 is the closest pixel to the block boundary, q1 is the next closest pixel to the block boundary in the q block of pixels, and q2 is the second next closest pixel to the block boundary in the q block of pixels.

144. Fu discloses three different processes for filtering pixels p_0 , p_1 , p_2 , q_0 , q_1 , and q_2 .

145. If BS equals 1, then Fu teaches that its process three “filter[s] the samples on both sides of the luma block boundary.” “[F]irst, filter p_0 and q_0 (P_0 and Q_0 are respectively filtered values of p_0 and q_0), which may be expressed as follows:”

$$\text{delta} = \text{Clip3}\left(-C, C, \left(\left((q_0 - p_0) \times 3 + (p_1 - q_1) + 4\right) \gg 3\right)\right)$$

$$P_0 = \text{Clip1}(p_0 + \text{delta})$$

$$Q_0 = \text{Clip1}(q_0 - \text{delta})$$

where, $\text{Clip3}(a,b,c)$ is defined by the following conditions: $a \leq c \leq b$; $c < a$; $c > b$; and c ; other, and $\text{Clip1}(a)$ is expressed as $\text{Clip1}(x) = \text{Clip3}(0, 255, x)$. *Id.*, 8:3-11.

146. In other words, for Fu’s $\text{Clip3}(a,b,c)$ function the result of the clipping function is “a” if c is less than a ; the result is “b” if c is greater than b and the result is “c” otherwise. Similarly, $\text{Clip1}(a)$ is Fu’s Clip3 function, where a is 0 and b is 255 (the range of values for a single 8-bit pixel, discussed above).

147. After q_0 and p_0 are filtered, Fu teaches to “then, determine whether there is a need to filter p_1 and q_1 , and when there is a need to filter p_1 and q_1 , the calculation of P_0 and Q_0 may be divided into three cases.” *Id.*, 8:12-13.

148. In the second case, “when a_p is greater than 0 at the luma block boundary, filter p_1 , and the filtered value is as follows:”

$$P1 = Clip1(p1 + Clip3(-C, C, ((P0 - p1) \times 3 + (p2 - Q0) + 4) \gg 3))$$

Id., 8:15-17.

149. And in the third case, “when a_q is greater than 0 at the luma block boundary, filter q_1 , and the filtered value is as follows:”

$$Q1 = Clip1(q1 - Clip3(-C, C, ((q1 - Q0) \times 3 + (P0 - q2) + 4) \gg 3))$$

Id., 8:18-20.

150. Fu teaches that its method of reducing block distortion through the use of offset equations can be implemented in a user device. Fu’s “[f]igure 9 is a schematic diagram of the structure of a device for reducing block distortion according to yet another example of the present disclosure, the device comprising a block boundary filtering strength acquisition module 910, a boundary determination module 920, a filter determination module 930, and a filter module 940.” Fu, 18:32-35.

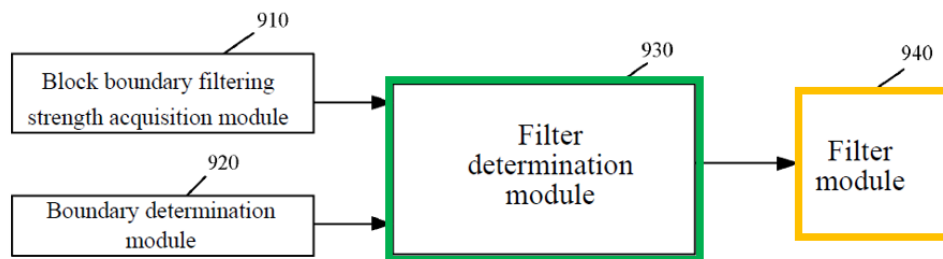


Figure 9

151. Fu’s “filter determination module 930 [green, above] determines the corresponding filter combination based on the pre-established correspondence relationship between block boundary filtering strength and filter combination according to the block boundary filtering strength acquired by the block boundary filtering strength acquisition module 910 when the determination results of the boundary determination module indicate that the block boundary is a pseudo-boundary.” Fu, 19:6-10.

152. Fu’s “filter module 940 filters the pixels of the blocks on both sides of the block boundary in the windows by using the filter combination determined by the filter determination module 930.” Fu, 19:11-12.

153. Fu teaches that “all or part of the steps of the method [discussed above] according to the above examples may be completed through a program instructing the relevant hardware, and the program may be stored in a type of computer-readable storage medium. When the program is executed, it comprises one or a combination of the steps of the method.” Fu, 19:13-16.

154. Fu further teaches that “each functional unit in the various examples of [its] present disclosure may be integrated in a processing module, where each unit may exist physically alone, or two or more units may be integrated in a module. The above integrated modules may be implemented in the form of hardware or software functional modules. If the integrated module is implemented in the form of a

software functional module and sold or used as a stand-alone product, it may be stored in a computer-readable storage medium” which may be “a read-only memory, magnetic disk, optical disc, or the like.” Fu, 19:18-23.

B. List D/ Applicant Admitted Prior Art (Ex-1024)

155. List is an IEEE article from the Journal for Transactions on Circuits and Systems for Video Technology, volume 13, issue 7 that was released in July 2003. List is titled *Adaptive Deblocking Filter* by Peter List, Anthony Joch, Jani Lainema, Gisle Bjontegaard (the same author as Bjontegaard, below) and Marta Karczewicz. Because List was published (July 2003) more than one year before the earliest-possible priority date of the '659 Patent (January 14, 2011), I understand that List constitutes prior art under at least §102(b).⁶ I also understand that the '659 Patent described List as applicant-admitted prior art, where a POSITA can turn for more information on deblocking filtering in H.264 (discussed above). Ex-1001, 2:14-17.

156. List “describes the adaptive deblocking filter used in the H.264/MPEG-4 AVC video coding standard. The filter performs simple operations to detect and

⁶ I have also reviewed the Declaration of June Munford, in which she explains that List was publicly available and accessible before the earliest-possible priority date of the '659 Patent. Ex-1007, ¶¶11-17, which supports my understanding that List constitutes prior art.

analyze artifacts on coded block boundaries and attenuates those by applying a selected filter.” List, Abstract. List recognizes two building blocks within H.264 that can be a source of blocking artifacts. “The most significant one is the block-based integer discrete cosine transforms (DCTs) in intra and inter frame prediction error coding. Coarse quantization of the transform coefficients can cause visually disturbing discontinuities at the block boundaries.” List, 614. “The second source of blocking artifacts is motion compensated prediction. Motion compensated blocks are generated by copying interpolated pixel data from different locations of possibly different reference frames. Since there is almost never a perfect fit for this data, discontinuities on the edges of the copied blocks of data typically arise. Additionally, in the copying process, existing edge discontinuities in reference frames are carried into the interior of the block to be compensated.” List, 614. List recognizes that “[a]lthough the small 4x4 sample transform size used in H.264/MPEG-4 AVC somewhat reduces the problem, a deblocking filter is still an advantageous tool to maximize coding performance.” *Id.*

157. List teaches that there are two approaches to deblocking filters—post filters or loop filters. *Id.* List teaches several advantages of loop filtering. “Firstly, the requirement of a loop filter *guarantees* a certain level of quality.” *Id.* “Secondly, there is no need for an extra frame buffer in the decoder.” *Id.* “Thirdly, empirical tests have shown that loop filtering typically improves both objective and subjective

quality of video streams with significant reduction in decoder complexity compared to post filtering.” *Id.*

158. List teaches that “[w]hen using a block transform for residual coding, it is well-known that the coding errors are larger near the block boundaries than in the middle of the block.” List, 615.

159. List, like Fu, recognizes that “[t]o every edge between two 4 x 4 luminance sample blocks, a *Boundary-Strength* (Bs) parameter is assigned an integer value from 0 to 4. Table I [below] shows how the value of Bs depends on the modes and coding conditions of the two adjacent blocks.” *Id.*, 615.

TABLE I
FILTER STRENGTH PARAMETER AS A
FUNCTION OF CODING MODE

Block modes and conditions	Bs
One of the blocks is Intra <i>and</i> the edge is a macroblock edge	4
One of the blocks is Intra	3
One of the blocks has coded residuals	2
Difference of block motion ≥ 1 luma sample distance	1
Motion compensation from different reference frames	1
Else	0

160. List further teaches that a “standard mode of filtering” is applied for edges with a Bs from 1 to 3, where “the value of Bs affects the maximum modification of the sample values that can be caused by filtering. The gradation of

Bs reflects that the strongest blocking artifacts are mainly due to intra and prediction error coding and are to a somewhat smaller extent caused by block motion compensation.” List, 615.

161. Figure 2 of List shows a “typical situation where the filter would be turned on.” List, 616.

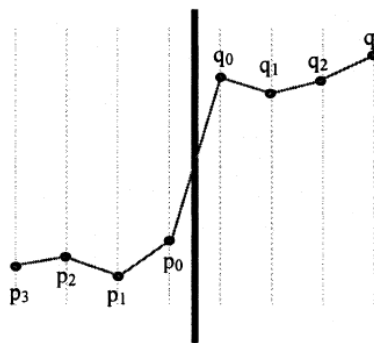


Fig. 2. One-dimensional visualization of a block edge in a typical situation where the filter would be turned on.

162. To filter edges with a Bs from 1 to 3, List discloses a basic filter operation, where modified values of p0 and q0 (p0' and q0' respectively) are calculated as:

$$p'_0 = p_0 + \Delta_0 \text{ and} \\ q'_0 = q_0 - \Delta_0$$

List, 617.

163. List uses the same pixel notation as the '659 Patent and Fu. Specifically, in one block of pixels you can have a row of pixels p0, p1, p2, and p3, where p0 is the pixel closest to the block boundary, p1 is the pixel next closest to the

block boundary, etc. Similarly, the neighboring block of pixels can have a row q_0 , q_1 , q_2 , q_3 , where q_0 is the pixel closest to the block boundary, q_1 is the pixel next closest to the block boundary, etc.

164. List teaches that its Δ_0 values are “calculated in a two-step process, with the calculation of an initial Δ_{0i} value, followed by clipping of this value before it is applied in the above equations.” The initial delta value is calculated using:

$$\Delta_{0i} = (4(q_0 - p_0) + (p_1 - q_1) + 4) \gg 3$$

List, 617.

165. The delta equation is then clipped. List recognizes that if the delta value was used directly, “it would result in far too much low-pass filtering (blurring).” List, 617. List teaches that its delta for the p_0 and q_0 equations can be clipped by the following equation:

$$\Delta_0 = \text{Min}(\text{Max}(-c_0, \Delta_{0i}), c_0)$$

166. List teaches that the values of p_1 and q_1 “are only modified if” the two additional spatial activity conditions below are under a threshold value (β). List, 617.

$$\begin{aligned} |p_2 - p_0| &< \beta(\text{Index}_B) \\ |q_2 - q_0| &< \beta(\text{Index}_B). \end{aligned}$$

167. If these equations are both true, then there is enough spatial activity that filtering p_1 and q_1 should occur, where p_1 is modified to p_1' using the following equation:

$$p_1' = p_1 + \Delta_{p1}.$$

List, 617.

168. The delta value for p_1 is calculated using the following equation:

$$\Delta p1i = (p2 + ((p0 + q0 + 1) \gg 1) - 2p1) \gg 1$$

List, 617.

169. Similarly, q_1 is modified to q_1' using the following equation:

$$q_1' = q_1 + \Delta_{q1}.$$

List, 617.

170. The delta value for q_1 is calculated using the same equation for p_1' by “substituting q_2 and q_1 for p_2 and p_1 , respectively.” List, 617. That equation would be:

$$\Delta q1i = (q2 + ((p0 + q0 + 1) \gg 1) - 2q1) \gg 1$$

List, 617.

171. List teaches that the delta for p_1 and q_1 can also be clipped using the following equations:

$$\Delta_{p1} = \text{Min}(\text{Max}(-c_1, \Delta_{p1i}), c_1)$$
$$\Delta_{q1} = \text{Min}(\text{Max}(-c_1, \Delta_{q1i}), c_1).$$

List, 617.

C. H.241 (Ex-1017)

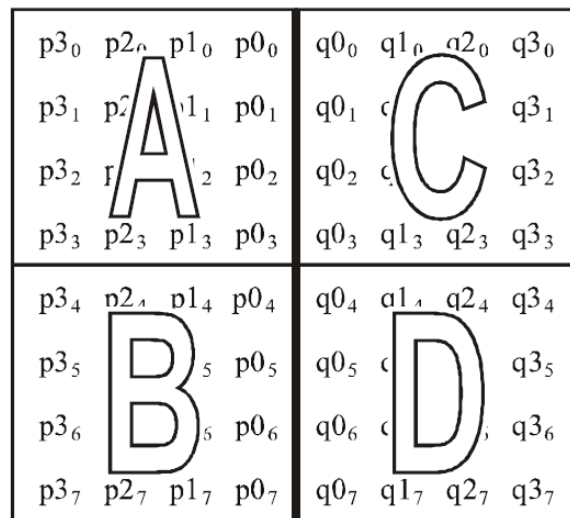
172. H.241 is an international telecommunications standard released by the ITU to “[e]xtend[] video procedures and control signals for H.300-series terminals,” which was published in December 2006. H.241, Cover. Because H.241 was published (December 2006) more than a year before the earliest-possible priority date of the ’659 Patent (January 14, 2011), I understand that H.241 would be prior art under at least pre-AIA §102(b).⁷ H.241 was not cited during prosecution. Ex-1001, 1-2.

173. The H.241 standard specifies how to implement other ITU video-coding standards—such as H.264—in video-processing terminals. H.241, 1. As part of the standard, H.241 includes “a reduced-complexity decoding process to be applied to H.264 Baseline profile bitstreams when such use has been negotiated

⁷ I have also reviewed the declaration of Dr. Cliff Reader, in which he explains that H.241 was publicly available and accessible before the earliest-possible priority date of the ’659 Patent, Ex-1005, ¶¶89-103, which supports my understanding that H.241 constitutes prior art.

using ITU-T Rec. H.241.” H.241, 1. In particular, Annex B provides a deblocking filter for video images different (and relatively less complex) than the deblocking filter used in H.264. H.241, 27.

174. Specifically, H.241 discloses an arrangement of two adjacent blocks, a “p” block and a “q” block, each having 8x8 pixels, that can be broken into smaller 4x4 blocks:



a) Filtering across vertical edges

H.241, 28.

175. As part of the modified deblocking filter process, H.241 states that a value “d” must first be calculated using the following equation:

$$d = \left| \begin{array}{c} p_{2_2} - 2 \cdot p_{1_2} + p_{0_2} \\ q_{0_5} - 2 \cdot q_{1_5} + q_{2_5} \end{array} \right| + \left| \begin{array}{c} q_{0_2} - 2 \cdot q_{1_2} + q_{2_2} \\ p_{2_5} - 2 \cdot p_{1_5} + p_{0_5} \end{array} \right| +$$

Id., 29. Then, “[a] determination is made whether or not to apply filtering across a block edge.” *Id.*, 29. To make that determination, H.241 enumerates a series of

conditions, in which if *any* “are true, no filtering across a block edge takes place.”

Id. One condition is if the value of $d \geq \beta$. *Id.*

176. H.241 explains that “ β ” is a function of “index” as specified in H.264, and “[t]o obtain β , use index = indexB,” *id.*, 27-28, which is not defined in H.241 itself.

Table B.2/H.241 – Value of t_c and β as a function of index

index	0-15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
t_c	0	1	1	1	1	1	2	2	2	2	2	2	2	2	2	2	3	3	3
β	0	6	7	8	9	10	11	12	13	14	15	16	17	18	20	22	24	26	28

To obtain t_c , use index = indexA. To obtain β , use index = indexB.

Table B.3/H.241 (concluded) – Value of t_c and β as a function of index

index	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51
t_c	3	4	4	4	5	5	6	6	7	8	9	9	11	12	13	13	16	18
β	30	32	34	36	38	40	42	44	46	48	50	52	54	56	58	60	62	64

To obtain t_c , use index = indexA. To obtain β , use index = indexB.

Id., 28.

177. H.241 also describes other conditions that control whether or not any deblocking filtering is performed apart from the inequality relationship ($d \geq \beta$), including if a variable “filterBlockEdge” is set equal to zero. *Id.*, 29. As H.241 explains, the variable filterBlockEdge is set to zero unless one of the following conditions are true: (1) the blocks are “intra” coded; (2) the blocks have non-zero transform coefficients; (3) the difference between motion vector components, and others, in which case the variable is set to 1. *Id.*

178. If deblocking filtering is to be performed based on one or more of the above conditions, H.241 then provides specific offset-filter equations to use to modify the pixel values:

$$\Delta = \text{Clip3}(-t_c, t_c, ((q_{0i} + ((p_{2i} + q_{1i}) \gg 1)) \gg 1) - ((p_{0i} + ((q_{2i} + p_{1i}) \gg 1)) \gg 1))$$

$$p_{0i} = \text{Clip1}_Y(p_{0i} + \Delta)$$

$$q_{0i} = \text{Clip1}_Y(q_{0i} - \Delta)$$

$$p_{1i} = \text{Clip1}_Y(p_{1i} + \Delta/2)$$

$$q_{1i} = \text{Clip1}_Y(q_{1i} - \Delta/2)$$

H.241, 29.

D. B205 / Applicant Admitted Prior Art (Ex-1018)

179. B205 is a working-draft proposal for the High-Efficiency Video Coding standard, and was published on October 9, 2010. Ex-1094, 3. Because B205 was published (October 9, 2010) less than a year before the earliest-possible filing date of the '659 Patent (January 14, 2011), I understand that B205 constitutes prior art under at least pre-AIA §102(a).⁸ However, I also understand that the '659 Patent described B205 as applicant-admitted prior art. Ex-1001, 2:19-22.

⁸ I have also reviewed the declaration of Dr. Cliff Reader, in which he explains that B205 was publicly available and accessible before the earliest-possible priority date of the '659 Patent, Ex-1005, ¶¶104-24, which supports my understanding that B205 constitutes prior art.

180. As both B205 and the '659 Patent explain, B205 disclosed a deblocking filter that used the same filter-decision equation as H.241. B205, 130; Ex-1001, 2:19-55.

$$|p_{2_2} - 2*p_{1_2} + p_{0_2}| + |q_{2_2} - 2*q_{1_2} + q_{0_2}| + |p_{2_5} - 2*p_{1_5} + p_{0_5}| + |q_{2_5} - 2*q_{1_5} + q_{0_5}| < \beta$$

181. And like H.241, B205 discloses that one of the conditions to decide whether to perform deblocking is if a decision value (d) is less than a threshold (β). B205, 130; Ex-1001, 2:19-55.

182. However, unlike H.241, if a decision is made to perform deblocking filtering, B205 provides a different equation used to modify two pixels on each side of the block boundary:

$$\begin{aligned}\Delta &= \text{Clip}(-t_c, t_c, (13*(q_{0_i} - p_{0_i}) + 4*(q_{1_i} - p_{1_i}) - 5*(q_{2_i} - p_{2_i}) + 16) \gg 5)) \\ p_{0_i} &= \text{Clip}_{0-255}(p_{0_i} + \Delta) \\ q_{0_i} &= \text{Clip}_{0-255}(q_{0_i} - \Delta) \\ p_{1_i} &= \text{Clip}_{0-255}(p_{1_i} + \Delta/2) \\ q_{1_i} &= \text{Clip}_{0-255}(q_{1_i} - \Delta/2)\end{aligned}$$

B205, 131; Ex-1001, 2:61-3:3.

E. B118 (Ex-1023)

183. B118 is a working draft of a standards proposal submitted for the Joint Video Team of ISO/IEC MPEG and ITU-T VCEG, and was published on the ITU's website on March 15, 2002. Because B118 was published (March 15, 2002), more than one year before the earliest-possible priority date of the '659 Patent (January

14, 2011), I understand that B118 is prior art under at least pre-AIA §102(b).⁹ B118 was not cited during prosecution. Ex-1001, 1-2.

184. B118 discloses that “[a]fter the reconstruction of a macroblock a conditional filtering of this macroblock is taking place, that effects the boundaries of the 4x4 block structure. Filtering is done on a macroblock level.” B118, 76.

185. B118 discloses a different pixel notation than Fu, the ’659 Patent, and List. In B118, “[t]he set of eight pixels across a 4x4 block horizontal or vertical boundary is denoted as:”

$$p_4, p_3, p_2, p_1 \mid q_1, q_2, q_3, q_4$$

“with the actual boundary between p1 and q1.” B118, 47.

186. In other words, the block boundary in B118 is denoted by the line above, where p1 is the closest pixel to the block boundary in the p block of pixels, p2 is the next closest pixel to the block boundary in the p block of pixels, etc. Similarly, q1 is the closest pixel to the block boundary in the neighboring q block of

⁹ I have also reviewed the declaration of Dr. Cliff Reader, in which he explains that B118 was publicly available and accessible before the earliest-possible priority date of the ’659 Patent, Ex-1005, ¶¶71-88, which supports my understanding that B118 constitutes prior art.

pixels, p2 is the next closest pixel to the block boundary in the q block of pixels, etc.

B118, 47.

187. Similar to Fu and List, B118 teaches that the filtering process is dependent on the boundary strength for the block boundary between two neighboring blocks of pixels, here p and q. B118, 47. B118 defines two “sorts” of filters. In the default case, B118 teaches that a weak filter is used for pixels p1 and q1 (in B118, the closest pixels to the block boundary for each respective block of pixels).

$$\begin{aligned}\Delta &= \text{Clip}(-C, C, ((q_1 - p_1) \ll 2 + (p_2 - q_2) + 4) \gg 3) \\ P_1 &= \text{Clip}(0, 255, (p_1 + \Delta)) \\ Q_1 &= \text{Clip}(0, 255, (q_1 - \Delta))\end{aligned}$$

188. For ease of reference, B118’s equations can be rewritten using the pixel notation from the ’659 Patent (color coded to identify the corresponding pixel notation):

$$\begin{aligned}\Delta &= \text{Clip}(-C, C, ((q_0 - p_0) \ll 2 + (p_1 - q_1) + 4) \gg 3) \\ P_0 &= \text{Clip}(0, 255, (p_0 + \Delta)) \\ Q_0 &= \text{Clip}(0, 255, (q_0 - \Delta))\end{aligned}$$

189. B118 then teaches that the following “two intermediate threshold variables” “are used to decide whether p_2 and q_2 [i.e., p_1 and q_1 in the ’659 Patent’s notation] are filtered.”

$$a_p = |p_3 - p_1|$$

$$a_q = |q_3 - q_1|$$

B118, 48 (showing the equations with B118’s pixel notation).

190. If the threshold value a_p is less than a threshold β , then the next closest pixel to the block boundary (p_2 in B118, p_1 in the ’659 Patent) is filtered by the following equation:

$$P_2 = p_2 + \text{Clip}(-C_0, C_0, (p_3 + P_1 - p_2 \ll 1) \gg 1)$$

Id. (showing the equation with B118’s pixel notation).

191. Similarly, if the threshold a_q is less than a threshold β , then the next closest pixel to the block boundary in the neighboring block (q_2 in B118, q_1 in the ’659 Patent) is filtered by the following equation:

$$Q_2 = q_2 + \text{Clip}(-C_0, C_0, (q_3 + Q_1 - q_2 \ll 1) \gg 1)$$

Id. (showing the equation with B118’s pixel notation).

192. B118 teaches that its offset equations are clipped by a clipping function with parameters $\text{Clip}(\text{Min}, \text{Max}, \text{Value})$ and clipping thresholds defined by the following table:

TABLE 1																																	
QP and Bs dependent Clipping thresholds																																	
QP	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
ClipTable[QP,1]	0	0	0	0	0	0	0	0	0	1	1	1	1	2	2	2	2	2	2	2	2	2	2	2	2	3	3	3	3	3	4	4	
ClipTable[QP,2]	0	0	0	0	0	0	0	0	1	1	2	2	2	2	2	2	2	2	2	2	3	3	3	3	4	4	4	4	5	5	6	6	

ClipTable[QP,3]	0	0	0	0	0	0	0	0	2	2	2	2	2	2	2	2	2	3	3	3	3	4	4	4	5	5	6	6	6	8	8	9	10
ClipTable[QP,4]	0	0	0	0	0	0	0	0	2	2	2	2	2	2	2	2	2	3	3	3	3	4	4	4	5	5	6	6	6	8	8	9	10

based on the equation $C0 = ClipTable[QP, Bs]$ where “C is set to C0 and then incremented by one if p_2 will be filtered and again by one if q_2 will be filtered.” B118, 48.

F. Bjontegaard (and Bjontegaard’s Code) (Ex-1009; Exs-1010-1015)

193. Bjontegaard (Ex-1009) is a working draft of a standards proposal submitted as part of the ITU Video Coding Experts Group (Question 15), Study Group 16, and was published on May 9, 2000. Ex-1082. Because Bjontegaard was published (May 9, 2000) more than one year before the earliest possible priority date of the ’659 Patent (January 14, 2011), I understand Bjontegaard constitutes prior art under at least pre-AIA §102(b).¹⁰ Bjontegaard was not cited during prosecution. Ex-1001, 1-2.

¹⁰ I have also reviewed the Declaration of Dr. Cliff Reader, in which he explains that Bjontegaard and Bjontegaard’s Code were publicly available and accessible before the earliest-possible priority date of the ’659 Patent. Ex-1005, ¶¶51-70, which further supports my understanding that Bjontegaard and Bjontegaard’s Code constitute prior art.

194. Moreover, alongside Bjontegaard's proposal disclosure, Bjontegaard provided a package of computer code in a series of computer files embodying the deblocking filter described in the proposal ("Bjontegaard's Code"). *See* Ex-1016 (Doc. Q15-J-27 stating "[t]he complete filter description is given in q15j28.doc"); Bjontegaard, 1 (labeled Q15-J-28); Ex-1010; Ex-1015, 1 (printout of the computer code's file system with an introductory "readme.txt"); Ex-1015, 1 (readme.txt file stating "lcodec4.0 is an implementation of the Telenor H.26L proposal for the Osaka meeting.... The new proposed Telenor elements described in document Q15J27"); *see also* Exs-1011-1014 (printouts of individual files within the code). Because Bjontegaard's Code was also published at the same time as Bjontegaard's proposal (May 9, 2000), Ex-1082 (showing Q15-J-27, 28, and 29 all uploaded May 9, 2000), I understand the code also constitutes prior art under at least pre-AIA §102(b). Moreover, because Bjontegaard's proposal and the associated code were published by the same contributor, at the same time, and expressly cross-reference each other, in my opinion, a POSITA would have read Bjontegaard in light of Bjontegaard's Code.

195. Bjontegaard was submitted as part of an early exploration to establish the next video-coding standard at the time, first under the title H.26L, which would ultimately become H.264. Ex-1038, 3 (noting H.26L was presumed to be a precursor to H.264 at the time).

196. Bjontegaard provides a proposal for a complete “coding model,” which was presented “by mainly defining the decoder actions.” Bjontegaard, 1, 4.

197. It is common for multimedia coding standards to be defined by the decoder’s operations, rather than the encoder. That is because coding standards describe the syntax to ensure that an encoded bitstream complying with a standard will be decodable by any standard-compliant decoder, thus ensuring interoperability among standards-compliant devices. What this means is that vendors that design encoders can modify the operation of the encoder in some ways, as long as the output from the encoder conforms to the syntax specified for decoders. For certain elements in which the encoder and decoder are intended to perform identically—such as “in-loop” elements used for image prediction and reconstruction—however, descriptions of the decoder would directly implicate the encoder’s operation as well. *See generally*, Section V.A (explaining how motion vectors and residuals determined at the encoder must be reconstructed, preferably identically, at the decoder).

198. Bjontegaard discloses a deblocking filter that is “in loop”—and thus would be expected to be present in both the encoder and decoder—because Bjontegaard describes the deblocking filter in Section 4, dedicated to the “[t]ransform and inverse transform” process associated with “in-loop” prediction coding. *See* Bjontegaard, 16-20; Section V.A. This is confirmed by Bjontegaard’s

Code, which includes the deblocking filter in the encoder file, *see* Ex-1014, 1 (encoder), 6 (deblocking filter), and is titled “loopfilter_new,” further indicating the deblocking filter is “in-loop.”

199. In the proposal, as part of the decoder, Bjontegaard discloses the incorporation of a “deblocking filter process.” Bjontegaard, 18. Bjontegaard explains that, before any filtering is performed, a decision is made whether or not to perform deblocking at all. *Id.*, 18-19. To make that decision, a series of “Strength” variables are determined based on the properties of the blocks under consideration, including, e.g., whether the block is “intra” coded, has “nonzero transform coefficients,” and the difference between motion vector components. *Id.*, 18-19.

200. Once it is decided to filter the boundary between two neighboring blocks, Bjontegaard describes the deblocking filter’s operations with reference to an arrangement of pixels that span two neighboring blocks, using pixels a-h:



Id., 19 (“Assume that the pixels a b c d e f g h belong to two neighbouring [sic] blocks (1 and 2). There is a block boundary between pixels d and e.”).

201. Bjontegaard then performs a series of calculations that are subsequently used to determine various operations of the deblocking filter:

$$\begin{aligned}S0 &= \mathbf{abs}(4(c-f)+12(e-d)) + \mathbf{abs}(9(c-d-e+f)) \\S0_1 &= 12\mathbf{abs}(a-2b+c) \\S1_1 &= \mathbf{abs}(4(a-d)+12(c-b)) + \mathbf{abs}(9(a-b-c+d)) \\S0_2 &= 12\mathbf{abs}(e-2f+g) \\S1_2 &= \mathbf{abs}(4(e-h)+12(g-f)) + \mathbf{abs}(9(e-f-g+h))\end{aligned}$$

Bjontegaard, 19. Specifically, the variable “S0” represents a measure of activity across the block boundary, because it uses pixels from both blocks 1 and 2. The variables “S0₁” and “S1₁” then represent different measures of activity in block 1, with S0₁ being a discrete version of the second-order derivative (or difference-of-difference) of three pixels in block 1, and the variable “S1₁” represents a weighted measure of activity that includes an additional pixel in block 1. The same is true for variables “S0₂” and “S1₂,” except these variables use pixels from block 2 instead of block 1.

202. After these five calculations are performed, Bjontegaard then uses these calculations to determine an overall measure of activity in each block and across the block boundary. Bjontegaard, 19. Specifically, Bjontegaard first determines the value of variable “Act_{overall}” equal to “3–i,” in which “i” is the smallest number such that $S0 < \text{Lim}(\text{qp}) * 2^i$. *Id.* As Bjontegaard explains, “Lim(qp)” is a “threshold value[] for the filtering process depending on qp,” otherwise known as the “quantization parameter.” *Id.* In fact, Bjontegaard provides a table to convert the quantization parameter (QP) to Lim(qp):

QP	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Lim	7	8	9	10	11	12	14	15	17	20	22	25	28	31	35	39	44	49	55	62	69	78	88	98	110	124	139	156	175	197	221	248

Id. In other words, a POSITA would have understood that $Act_{overall}$ is effectively an integer representation of the overall activity across the block boundary, and can take on a value between 0 and 3. *Id.* Moreover, because a lower value of $S0$ means less pixel activity across the boundary, which thereby means a lower value of “i” is required for the inequality $S0 < Lim(qp) \cdot 2^i$ to be true, $Act_{overall}$ will increase as activity decreases.

203. After the integer value for overall activity ($Act_{overall}$) is determined, Bjontegaard determines an integer value of the pixel activity in block 1 (Act_1) using the following conditions:

- Act_1 is initially set to 1
- If $S0_1 < Lim(qp)$, Act_1 is incremented by 1
- If $S1_1 < Lim(qp)$, Act_1 is incremented by 1

Bjontegaard, 19. In other words, Act_1 is incremented to take on a value between 1 and 3 depending on how much pixel activity is occurring in block 1. Because $S0_1$ and $S1_1$ will decrease with less activity, that will cause Act_1 to increase, meaning a higher value of “ Act_1 ” indicates that there is less activity in the block. But if $S0_1$ and $S1_1$ are large due to more activity in the block, Act_1 will not increment and will stay equal to 1.

204. Once the value of Act_1 has been determined, Bjontegaard explains that the “same procedure is made for Act_2 ” to determine the aggregate activity in block 2 (using $S0_2$ and $S1_2$). Bjontegaard, 19.

205. When each of $Act_{overall}$, Act_1 and Act_2 have been determined, a final comparison is made to determine variables Act_{11} and Act_{22} . Specifically, Bjontegaard explains that Act_{11} is set equal to the smaller of Act_1 and $Act_{overall}$, and the same for Act_{22} by comparing Act_2 to $Act_{overall}$. Bjontegaard, 19. This, in effect, ensures that the amount of filtering that is subsequently applied to the pixels is determined based on the overall activity across the boundary *or* the activity within each block, depending on which measure demonstrates more activity (because a smaller activity value indicates *more* activity in the region of interest).

206. Once the final activity levels (Act_{11} and Act_{22}) for each block have been determined, Bjontegaard describes a process to modify the values of pixels d and e—the two pixels closest to the block boundary. Bjontegaard, 20 (§4.5.5). Bjontegaard averages Act_{11} and Act_{22} to produce variable “ Act_{edge} ,” which is then used to determine which of two offset equations are applied to pixels d and e:

$$Act_{edge} = (Act_{11} + Act_{22})/2$$

$$\text{If } Act_{edge} \text{ equals 1 or 2:} \quad Diff = (3(c - f) - 8(d - e))/16$$

$$\text{If } Act_{edge} \text{ equals 3:} \quad Diff = (b + c - 8(d - e) - f - g)/16$$

$$d_{\text{modified}} = d + \text{Diff} \text{ (clipped to stay in the range 0-255)}$$

$$e_{\text{modified}} = e - \text{Diff} \text{ (clipped to stay in the range 0-255)}$$

Bjontegaard, 20. As shown here, the variable “Diff” represents how much the pixel value for pixels d and e should change. Once “Diff” is calculated (and clipped to a value whose absolute value does not exceed “Clip_{edge},” not shown), it is then added to pixel d and subtracted from pixel e so that their pixel values are closer to one another.

207. After filtering pixels d and e, Bjontegaard discloses that a decision is made whether to filter a second pixel (c) in block 1. Bjontegaard, 20 (§4.5.6.1). Specifically, Bjontegaard initializes “Diff” or the “offset” to zero, meaning by default pixel c will not be modified. Only if Act₁₁ equals 2 or 3 will an offset be calculated to modify pixel c:

$$\text{If Act}_{11} \text{ equals 2:} \quad \text{Diff} = 5(b - 2c + d)/16$$

$$\text{If Act}_{11} \text{ equals 3:} \quad \text{Diff} = 3(a + b - 4c + d + e)/16$$

$$c_{\text{modified}} = c + \text{Diff} \text{ (clipped to stay in the range 0-255)}$$

Bjontegaard, 20. Once “Diff” is calculated (and clipped to a value whose absolute value does not exceed “Clip(qp,Strength1),” not shown), it is then added to pixel c. By implication, that means if Act₁₁ equals 0 or 1, pixel c will not be modified because the offset “Diff” will remain zero.

208. This same process is then repeated to decide whether to filter pixel f in block 2, but using Act_{22} and pixels from block 2. Bjontegaard, 20 (§4.5.6.2).

209. Bjontegaard's Code then implements Bjontegaard—including Bjontegaard's deblocking filter—in both a video encoder and decoder. *See, e.g.*, Ex-1012 (decoder); Ex-1014 (encoder). Bjontegaard's Code uses different variables in the computer-code variables than what appears in Bjontegaard, but they can be mapped to one another. For instance, the equations provided in Bjontegaard can be mapped to the equations provided in Bjontegaard's Code as follows, with the image pointers "img->" removed from the Code and some equations restructured (but not in a way that alters the mathematics) for mapping and readability.

Bjontegaard (Bjontegaard, 19-20)	Bjontegaard's Code (Ex-1012, 10-11; Ex-1014, 10-12)
$S0 = \text{abs}(4(c-f)+12(e-d)) + \text{abs}(9(c-d-e+f))$	$i0=4*(li[2]-li[5])-12*(li[3]-li[4])$ $i1=9*(li[2]+li[5]-li[3]-li[4]);$ $iaa=\text{abs}(i0)+\text{abs}(i1)$ <i>thus, when combined:</i> $iaa=\text{abs}(4*(li[2]-li[5])+12*(li[4]-li[3])) + \text{abs}(9*(li[2]-li[3]-li[4]+li[5]))$
$S1_1 = \text{abs}(4(a-d)+12(c-b)) + \text{abs}(9(a-b-c+d))$	$i0=4*(li[0]-li[3])-12*(li[1]-li[2]);$ $i1=9*(li[0]+li[3]-li[1]-li[2]);$ $il4=\text{abs}(i0)+\text{abs}(i1);$ <i>thus, when combined:</i> $il4=\text{abs}(4*(li[0]-li[3])+12*(li[2]-li[1])) + \text{abs}(9*(li[0]-li[1]-li[2]+li[3]))$
$S0_1 = 12\text{abs}(a-2b+c)$	$il3=12*\text{abs}(li[1]-2*li[2]+li[3]);$
$S0_2 = 12\text{abs}(e-2f+g)$	$ir3=12*\text{abs}(li[4]-2*li[5]+li[6]);$
$S1_2 = \text{abs}(4(e-h)+12(g-f)) + \text{abs}(9(e-f-g+h))$	$i0=4*(li[4]-li[7])-12*(li[5]-li[6]);$ $i1=9*(li[4]+li[7]-li[5]-li[6]);$ $ir4=\text{abs}(i0)+\text{abs}(i1);$ <i>thus, when combined:</i> $ir4=\text{abs}(4*(li[4]-li[7])+12*(li[6]-li[5])) + \text{abs}(9*(li[4]-li[5]-li[6]+li[7]))$
$\text{Diff} = (3(c-f) - 8(d-e))/16$	$id=(3*(li[2]-li[5])-8*(li[3]-li[4]))/16;$
$\text{Diff} = (b+c-8(d-e)-f-g)/16$	$id=(li[1]+li[2]-8*(li[3]-li[4])-li[5]-li[6])/16;$
$\text{Diff} = 5(b-2c+d)/16$	$id=5*(li[1]-2*li[2]+li[3])/16;$
$\text{Diff} = 3(a+b-4c+d+e)/16$	$id=3*(li[0]+li[1]-4*li[2]+li[3]+li[4])/16;$
$\text{Diff} = 5(e-2f+g)/16$	$id=5*(li[4]-2*li[5]+li[6])/16;$
$\text{Diff} = 3(d+e-4f+g+h)/16$	$id=3*(li[3]+li[4]-4*li[5]+li[6]+li[7])/16;$

This mapping shows that the variable $li[n]$ in Bjontegaard's Code is mapped to pixels a-h in Bjontegaard as follows:

Bjontegaard	a	b	c	d	e	f	g	h
Bjontegaard's Code	$li[0]$	$li[1]$	$li[2]$	$li[3]$	$li[4]$	$li[5]$	$li[6]$	$li[7]$

This mapping, however, also reveals a typographical error in Bjontegaard's $S0_1$ equation (identified above in red), Bjontegaard, 19, because Bjontegaard's Code uses what should be pixels b, c, and d, rather than pixels a, b, and c, *see, e.g.*, Ex-1012, 10

210. Bjontegaard's Code also clarifies how second-pixels c and f are filtered compared to what Bjontegaard itself may suggest. Specifically, in some equations, Bjontegaard uses the subscript "modified" to identify when a pixel has been filtered. *See* Bjontegaard, 20 (labeling filtered pixels c_{modified} , d_{modified} , e_{modified} , f_{modified}). However, when filtering pixels c and f, which is performed *after* pixels d and e have been modified, Bjontegaard does not use the "modified" subscript for pixels d and e in the filter equations for pixels c and f (i.e., $5(b-2c+d_{\text{modified}})/16$ for pixel c; $5(e_{\text{modified}}-2f+g)/16$ for pixel f). Thus, Bjontegaard at least superficially suggests the original-pixel values for pixels d and e are used to filter pixels c and f, respectively. But, as I describe in more detail below, Bjontegaard's Code confirms that the *modified* values of pixels d and e (d_{modified} and e_{modified}) are used to filter pixels c and

f, respectively, because Bjontegaard's Code redefines the values of li[3] (pixel d) and li[4] (pixel e) *before* filtering pixels c and f. *See, e.g.,* Ex-1012, 11 (calculating the offset (Diff) for pixel d using equation " $id=(3*(li[2]-li[5])-8*(li[3]-li[4]))/16$ ", applying the offset to pixel d using equation " $il1=li[3]+id$ " ($d_{\text{modified}} = d + \text{Diff}$), and then redefining the value of pixel d using equation " $li[3] = il1$ " ($d = d_{\text{modified}}$) *before* using the redefined variable " $li[3]$ " in the filter equation for pixel c under heading " $/*$ center left $*/$ ").

211. Thus, when Bjontegaard and Bjontegaard's Code are read together, a complete and accurate picture of the operation of Bjontegaard's deblocking filter is provided.

212. Bjontegaard also discloses a clipping function that will set upper and lower bounds for Bjontegaard's offset equations. Bjontegaard discloses that its clipping of the offset equations will depend on the quantization parameter and Strength (discussed above). Bjontegaard discloses a table of clipping values, based on the quantization parameter (qp) and Strength (1, 2, or 3). Bjontegaard, 19-20.

```
QP.....0·1·2·3·4·5·6·7·8·9·10·11·12·13·14·15·16·17·18·19·20·21·22·23·24·25·26·27·29·30·31¶
Clip(qp,1)·0·0·0·0·0·0·0·0·0·0·0·0·0·0·1·1·1·1·1·1·1·1·1·1·1·1·1·1·2·2·2·2·2·2·3·3¶
Clip(qp,2)·0·0·0·0·0·0·0·0·0·0·0·0·1·1·1·1·1·1·1·1·1·1·2·2·2·2·2·2·3·3·3·3·4·4·5·5¶
Clip(qp,3)·0·0·0·0·0·0·0·0·0·0·1·1·1·1·1·1·1·1·1·1·2·2·2·2·2·2·3·3·3·3·4·4·5·5·5·5·7·8·9¶
```

213. Bjontegaard refers to this table as Clip(qp, Strength), where that function is used in Bjontegaard's Clip_{edge} equation, below.

$$\text{Clip}_{\text{edge}} = (\text{Clip}(\text{qp}, \text{Strength}_1) + \text{Clip}(\text{qp}, \text{Strength}_2) + \text{Act}_1 + \text{Act}_2 - 2) / 2$$

Bjontegaard, 20.

214. As seen above, $\text{Clip}_{\text{edge}}$ depends on the Strength of block 1, the Strength of block 2, and the Act_1 and Act_2 values discussed above.

215. Bjontegaard, in turn, relies on its $\text{Clip}_{\text{edge}}$ function to clip the first offset “Diff” using the equation shown below:

$$\text{Diff} = \max(-\text{Clip}_{\text{edge}}, \min(\text{Clip}_{\text{edge}}, \text{Diff}))$$

Bjontegaard, 20.

216. For its second and third offset equations, however, Bjontegaard relies on the $\text{Clip}(\text{qp}, \text{Strength})$ function. For pixel p1, Bjontegaard teaches that the result “Diff” from its second offset equation is clipped using the following equation that brings in the Strength value of block 1 (or the p block in terms of the ’659 Patent):

$$\text{Diff} = \max(-\text{Clip}(\text{qp}, \text{Strength}_1), \min(\text{Clip}(\text{qp}, \text{Strength}_1), \text{Diff}))$$

Bjontegaard, 20.

217. For pixel q1, Bjontegaard teaches that the result “Diff” from its third offset equation is clipped using the following equation that brings in the Strength value of block 2 (or the q block in terms of the ’659 Patent):

$$\text{Diff} = \max(-\text{Clip}(\text{qp}, \text{Strength}_2), \min(\text{Clip}(\text{qp}, \text{Strength}_2), \text{Diff}))$$

Bjontegaard, 20.

218. Bjontegaard also discloses clipping its modified pixel values— d_{modified} , e_{modified} , c_{modified} , and f_{modified} (as shown below)—using a lower bound of 0 and an upper bound of 255. Bjontegaard, 20. As discussed above, these lower and upper bounds correspond to the pixel values.

$$\underline{d_{\text{modified}}} = d + \text{Diff} \text{ (clipped to stay in the range 0-255)}$$

$$\underline{e_{\text{modified}}} = e - \text{Diff} \text{ (clipped to stay in the range 0-255)}$$

$$\underline{c_{\text{modified}}} = c + \text{Diff} \text{ (clipped to stay in the range 0-255)}$$

$$\underline{f_{\text{modified}}} = f + \text{Diff} \text{ (clipped to stay in the range 0-255)}$$

G. Gonzalez (Ex-1020)

219. Gonzalez is a textbook titled “Digital Image Processing” which was authored by Rafael C. Gonzalez and Richard E. Woods, and published in 2008. Because Gonzalez was published (2008) more than one year before the earliest possible priority date of the ’659 Patent (January 14, 2011), I understand Gonzalez

constitutes prior art under at least pre-AIA § 102(b).¹¹ Gonzalez was not cited during prosecution. Ex-1001, 1-2.

220. Gonzalez is a general-reference textbook commonly used for undergraduate courses that describes many of the underlying concepts of image processing.

221. For instance, Gonzalez provides chapters on “Spatial Filtering” and “Filtering in the Frequency Domain” to modify pixel values by “accepting (passing) or rejecting certain frequency components” in the image. *See, e.g.*, Gonzalez, 104, 145, 199-200. For instance, Gonzalez describes how “linear filters” can take a series of pixel values as inputs, and then (among other things) average them equally or weight pixels differently to calculate a new pixel value that eliminates unwanted image components, such as noise. *See generally*, Gonzalez, 151-53.

222. In another chapter on “Image Segmentation,” Gonzalez explains how images can be analyzed not to output modified pixels, but for the “attributes extracted from those images,” such as identifying objects through edge detection.

¹¹ I have also reviewed the Declaration of June Munford, in which she explains that Gonzalez was publicly available and accessible before the earliest-possible priority date of the '659 Patent. Ex-1007, ¶¶6-10, which supports my understanding that Gonzalez constitutes prior art.

Gonzalez, 689. Gonzalez explains that “segmentation methods ... are based on detecting sharp, *local* changes in intensity.” *Id.*, 692. To detect these features, Gonzalez explains how “first- and second-order derivatives are particularly well-suited for this purpose,” and that “[d]erivatives ... are defined in terms of differences.” *Id.*

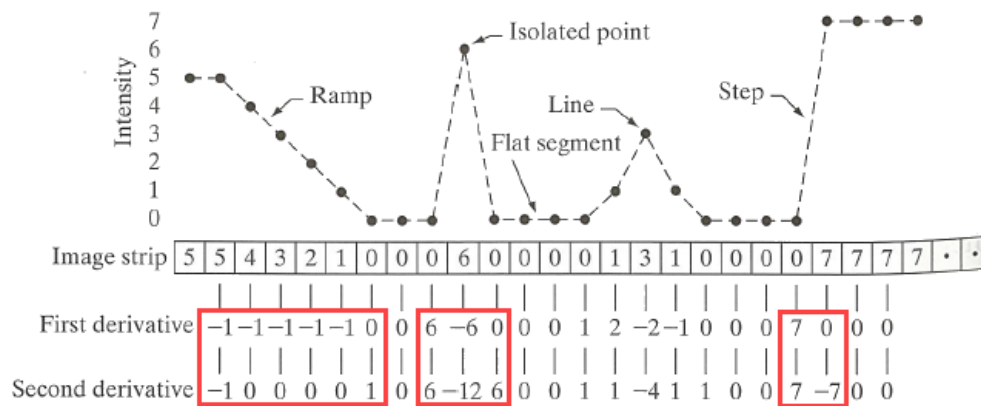
223. Gonzalez then provides a walkthrough of different algorithms that are used for image segmentation, beginning with the simplest one-dimensional “first derivative” that simply calculates the difference between two-pixel values, Gonzalez, 693, as I discussed previously, *see* Section V.B.2 (overview of filters). Gonzalez then explains that the second-order derivative takes the *difference-of-differences*, producing an equation known as the Laplacian:

$$\frac{\partial^2 f}{\partial x^2} = f''(x) = f(x + 1) + f(x - 1) - 2f(x)$$

Gonzalez, 693; *see also id.*, 160 (noting this second-order derivative is called the “Laplacian”). When written as a one-dimensional mask (to apply to a set of three pixels on a line), it would be written using the three-tap filter [1, -2, 1], because those three coefficients would be applied to each pixel ([$(x-1)$, (x) , $(x+1)$], with “ x ” representing the current pixel location) within the mask. As Gonzalez explains, “[b]ecause the Laplacian is a derivative operator, its use highlights intensity

discontinuities in an image and deemphasizes regions with slowly varying intensity levels.” *Id.*, 161.

224. Gonzalez then shows how a first-order difference calculation and a second-order Laplacian produce different outputs for purposes of image analysis:



Gonzalez, 694 (annotations included). Gonzalez explains that “first-order derivatives produce ‘thick’ edges and second-order derivatives much finer ones,” and that “a second-order derivative is much more aggressive than a first-order derivative in enhancing sharp changes. Thus, we can expect second-order derivatives to enhance fine detail (including noise) much more than first-order derivatives.” *Id.*, 694-95. Moreover, because the second-order derivative produces a characteristic “double edge” (also known as a “zero crossing,” Ex-1041, 149), this feature allows for more accurate identification of edges. Gonzalez, 695. For this reason, Gonzalez explains that the Laplacian is ideal for detecting “isolated points” in an image, lines, and edges. *See id.*, 695-97.

225. Moreover, Gonzalez explains how these two, one-dimensional equations can be combined into a two-dimensional “mask” that assigns the following weights to a 3x3 set of pixel values taken as inputs:

0	1	0
1	-4	1
0	1	0

Id., 161.

VIII. LEVEL OF A PERSON OF ORDINARY SKILL

226. I understand that when both interpreting the claims and assessing the prior art, I must do so from the perspective of a person of ordinary skill in the art (“POSITA”) as of the relevant priority date. My understanding is that the earliest possible priority date of the ’659 Patent is January 14, 2011.

227. In my opinion, at that time, a POSITA would have had a bachelor’s degree in electrical engineering, computer science, or a similar field and two to three years of experience in the design and development of relevant multimedia processing, such as video and/or image coding or compression. However, advanced graduate studies with coursework and/or research focusing on topics of video or image encoding/decoding and processing could substitute for working experience in the field.

228. I was at least a POSITA as of the earliest possible priority date of the '659 Patent, and I still am.

IX. CLAIM CONSTRUCTION

229. As I noted above, I understand that unless claim terms are provided an express construction, the terms must be given their plain and ordinary meaning.

230. I understand that Petitioner does not propose any constructions for terms in the Challenged Claims here.

231. But I also understand that in separate ITC proceedings, Lenovo and Patent Owner dispute whether the preamble of claim 10 is limiting in its entirety. I understand that Patent Owner proposes that the entire preamble of claim 10—“[a] user equipment comprising a deblocking filter unit for reducing blocking artifacts at a block boundary between a block of multiple pixels and a neighboring block of multiple pixels in a video frame...” is limiting.

232. I also understand that Lenovo agrees that the preamble is limiting except for the phrase “user equipment comprising.” In other words, I understand that Lenovo argues that the following language from the preamble is limiting: “a deblocking filter unit for reducing blocking artifacts at a block boundary between a block of multiple pixels and a neighboring block of multiple pixels in a video frame....” Ex-1026, 7.

233. I do not offer an opinion on the correct construction, because in my opinion, every claim element was taught by the prior art under any of the proposed construction, as I explain in further detail below.

234. I also understand that Patent Owner and Lenovo agreed that the term “configured to” as used in Challenged Claims 10 through 12 should be afforded its plain and ordinary meaning—i.e., “programmed to, as opposed to being merely capable of, performing a task.” Ex-1026, 4.

235. I apply this construction in my analysis of the prior art below.

X. GROUNDS

A. Ground 1: Obviousness over Fu in view of a POSITA’s Understanding of the Art, Evidenced by List, Bjontegaard, B205, and B118

1. Claim 1

236. In my opinion, Claim 1 is invalid as obvious over Fu in view of the knowledge of a POSITA.

a. Element 1[pre]: A method of reducing blocking artifacts at a block boundary between a block of multiple pixels and a neighboring block of multiple pixels in a video frame, said method comprises:

237. If the preamble is limiting, in my opinion, Fu discloses element 1[pre].

238. Fu teaches “[i]n video coding standards such as H.264 and digital audio and video codec technical standards, block processing is required for all images, for

example, the basic processing unit of the H.264 standard is a 16x16 macroblock.”
Fu, 1:16-18.

239. As discussed above, Fu recognizes that block distortion can occur due to a quantization error from block processing during quantization. Fu teaches that trapezoidal noise and lattice noise “make the edges of the image so clear that a block-shape region appears.” Fu, 1:20-23. A block boundary (or edge) can be clear if there is a step up in pixel values between pixels on each side of the block boundary. For example, if a pixel value p0 on one side of the block boundary is 50 and the value of pixel q0 on the other side of the block boundary in the neighboring block is 200, that step in value of 150 will result in a clear edge between the p and q blocks. This can also be referred to as a blocking artifact—or an artifact that remains after pixel blocks are processed. *See* List, 614 (“It can be seen that the main coding artifact in the loop-filter case is the blockiness on the 4 x 4 grid caused by prediction error coding. This artifact can be efficiently compensated by the deblocking filter described in this paper.”).

240. Fu, therefore, teaches “[a] specific method for reducing block distortion is introduced in detail below, using an 8x8 block boundary as an example, specifically illustrating the specific steps of the method for reducing block distortion.” Fu, 1:34-38. Fu gives an example 8x8 block boundary shown in Figure 1 below:

		N0	R0	L0	M0		
		N1	R1	L1	M1		
		N2	R2	L2	M2		
		N3	R3	L3	M3		
		N4	R4	L4	M4		
		N5	R5	L5	M5		
		N6	R6	L6	M6		
		N7	R7	L7	M7		

Figure 1

241. In Figure 1 above, the left and right pixel blocks are separated by a block boundary that is identified by the thick solid line. Fu, 2:5-8.

242. As discussed above, and in further detail below, Fu discloses a multistep process that includes “filter[ing] the samples on both sides of the luma block boundary.” Fu, 8:3-5; *see also id.*, 4:19-28 (“Below, the filtering process of p_0 , p_1 , p_2 , q_0 , q_1 and q_2 in Figure 2 is illustrated with a schematic diagram of the distribution of pixels of a row/column on both sides of the block boundary shown in Figure 2, and the filtered values of p_0 and q_0 are expressed by P_0 and Q_0 ,

respectively. If $P_1(Q_1)$ is not assigned a value during filtering, $p_1(q_1)$ is not filtered; otherwise, P_1 or Q_1 is the filtered value of p_1 or q_1 . P_2 or Q_2 is the filtered value of p_2 or q_2 .”).

243. To illustrate its filtering process and the pixels that will be filtered, Fu provides a naming convention through Figure 2, below (annotated):

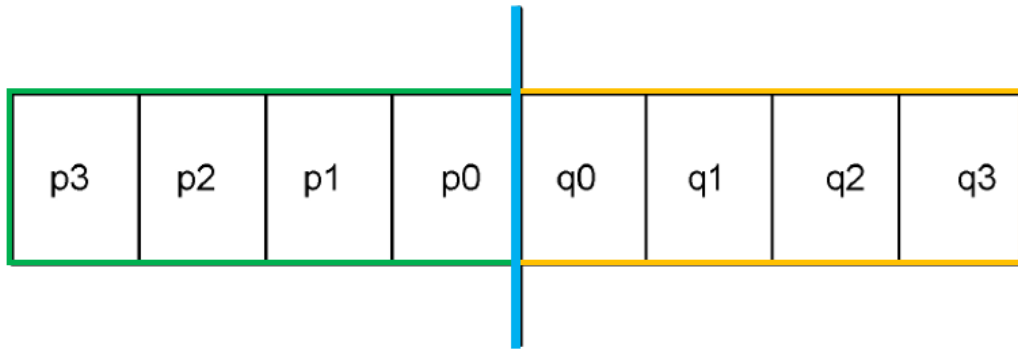


Figure 2

244. In Figure 2, there is a p block of pixels with a line of pixels (green), represented by consecutive pixels p_0 , p_1 , p_2 , and p_3 , where p_0 is the pixel closest to the block boundary (blue line), p_1 is the next closest pixel to the block boundary, p_2 is the second next closest pixel to the block boundary, and p_3 is the third next closest pixel to the block boundary. Figure 2 also shows a corresponding line of pixels in a neighboring block of pixels, q block (orange) represented by consecutive pixels q_0 , q_1 , q_2 , and q_3 , where q_0 is the pixel closest to the block boundary, q_1 is the next closest pixel to the block boundary, q_2 is the second next closest pixel to the block boundary, and q_3 is the third next closest pixel to the block boundary.

- b. Element 1[a]: calculating a first offset to be equal to $(9 \times (q_0 - p_0) - 3 \times (q_1 - p_1) + 8) \gg 4$, wherein $\gg$ denotes a right shift operation, p_0 denotes a pixel value of a pixel closest to said block boundary in a line of pixels in said block, wherein said line of pixels is perpendicular to said block boundary, p_1 denotes a pixel value of a pixel next closest to said block boundary in said line of pixels, q_0 denotes a pixel value of a pixel closest to said block boundary in a corresponding line of pixels in said neighboring block, wherein said corresponding line of pixels is perpendicular to said block boundary and q_1 denotes a pixel value of a pixel next closest to said block boundary in said corresponding line of pixels;**

245. In my opinion, Fu in view of a POSITA's knowledge discloses or suggests element 1[a].

246. Fu discloses a three-step process to reduce block distortion. In step 1, Fu calculates the flatness of the region. Fu, 2:1-3 ("Step 1: Using the interpolation of the actual pixels on both sides of the present block boundary, calculate the flatness of the image region where the present block boundary is located, expressing the flatness of the region with the number of similarity pairs around the block boundary."). In step 2, Fu determines the block boundary filter strength, which it labels BS. Fu, 4:3-5 ("Step 2: Determine the filtering strength of the present block boundary based on the specific flatness. The specific method is as follows: determine the value of the block boundary filtering strength (BS) based on the macroblock type and region flatness, and the steps are as follows:..."). Fu teaches

that the value of the block boundary filtering strength determines the filtering process to be used—i.e., whether a weak or strong filter is used. *See* List, 615 (“In the actual filtering algorithm, Bs determines the strength of the filtering performed on the edge, including the selection between the two primary filtering modes.”).

247. In step 3, Fu teaches that one should “filter the pixels on both sides of the block boundary according to the flatness of different non-zero BS values.” Fu, 4:19-20.

248. For purposes of my analysis, a BS of one is assumed. For luma block boundaries with a BS value of one, Fu teaches that one should “filter the samples on both sides of the luma block boundary.” In Fu’s filtering process three, pixels p0 and q0 are filtered first before a decision is made whether to filter pixels p1 and q1. Fu, 8:3-20. Figure 2 of Fu shows exemplary lines of pixels and the notation that Fu applies to each pixel.

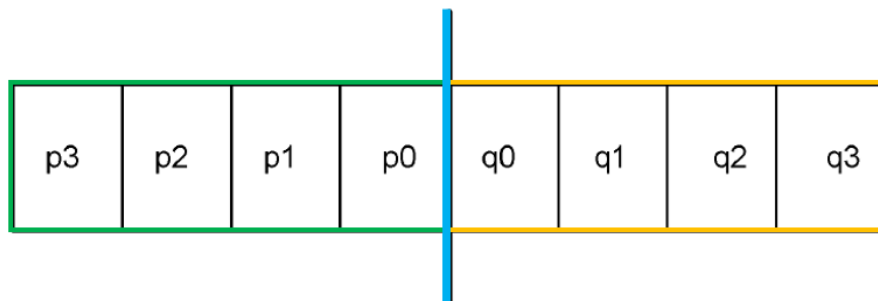


Figure 2

Fu, Fig. 2 (annotated).

249. As shown in annotated Figure 2 above, a line in p block of pixels—p0, p1, p2, and p3 (green) is on the left side of the block boundary (blue). One can see that the line of pixels, p0, p1, p2, p3, is perpendicular to the block boundary line. Figure 2 also depicts a corresponding line in neighboring q block of pixels—q0, q1, q2, and q3 (orange). The line of pixels q0, q1, q2, q3, is also perpendicular to the block boundary line.

250. In the line of p pixels, p0 is the closest pixel to the block boundary line. Similarly, in the corresponding line of pixels in the neighboring block q, q0 is the closest pixel to the block boundary line.

251. In Fu’s process 3, one “first, filter[s] p0 and q0” where “P0 and Q0 are respectively filtered values of p0 and q0).” Fu expresses its filtering equations for pixels p0 and q0 as:

$$\text{delta} = \text{Clip3}\left(-C, C, \left(\left((q0 - p0) \times 3 + (p1 - q1) + 4\right) \gg 3\right)\right)$$

$$P0 = \text{Clip1}(p0 + \text{delta})$$

$$Q0 = \text{Clip1}(q0 - \text{delta})$$

Fu, 8:3-10.

252. Turning to Fu’s equation to calculate delta first, I will disregard the clipping function for now to focus on the first offset portion of Fu’s delta equation, which is boxed in purple above. That expression is reproduced as:

$$\left((q_0 - p_0) \times \textcolor{red}{3} + (p_1 - q_1) + \textcolor{blue}{4} \right) \gg \textcolor{orange}{3}$$

253. For these expressions, I have added color to track the coefficients as I walk through mathematically equivalent equations below. The colors are as follows: red is the coefficient applied to the term $(q_0 - p_0)$; green is the coefficient applied to the $(p_1 - q_1)$ term (which is 1 in the above equation, and therefore not shown); blue is the color applied to the rounding offset; and orange is the color applied to the divisor (or right shift value).

254. Fu's first offset equation is disclosed in terms of a right shift operation, signified by the ">>" term above (as discussed above in the Technology Background section). For example, Fu also recognizes that "wherein >>1 indicates shifting one place to the right, in the binary computer system, this means division by 2. Similarly, >>2 indicates shifting two places to the right and means division by 4, >>3 indicates shifting three places to the right, and means division by 8, and so on." Fu, 3:5-8. Similarly, B205 teaches that " $x \gg y$ " is an "[a]rithmetic right shift of a two's complement integer representation of x by y binary digits. This function is defined only for positive integer values of y. Bits shifted into the MSBs as a result of the right shift have a value equal to the MSB of x prior to the shift operation." B205, §3.4; *see also* Ex-1001 (">> denotes a right shift operator").

255. Turning back to Fu's first offset equation, it can be multiplied by $2/2$ (i.e., 1) to reach a mathematical equivalent expression below, where the result of the

“numerator” (i.e., $(q_0 - p_0) \times 3 + (p_1 - q_1) + 4$) is right shifted by four instead of three (because multiplying a 2^3 by 2 equates to 2^4).

$$= ((6 \times (q_0 - p_0) + 2 \times (p_1 - q_1) + 8) \gg 4))$$

256. The “2” and “ $(p_1 - q_1)$ ” terms are mathematically the same as “ $-2 \times (q_1 - p_1)$,” and can be re-written (for ease of comparison to similar offset equations) as:

$$= ((6 \times (q_0 - p_0) - 2 \times (q_1 - p_1) + 8) \gg 4))$$

257. Fu teaches that its first offset is clipped by Fu’s Clip3(a, b, c) function (described above) before it is added to p_0 (or subtracted from q_0) to result in the modified pixel value of p_0 and q_0 , p_0' and q_0' respectively. Fu, 8:3-10. Claim 1 does not recite that the first offset is clipped before it is used to derive the modified pixel values p_0' and q_0' . Ex-1001, 23:55-26:17.

258. Even though Fu teaches that the first offset is clipped, it still discloses the first offset equation of element 1[a], as discussed further below. Fu’s Clip3(a,b,c) function is defined by the following conditions:

- If c is less than a, then the resulting value is a
- If c is greater than b, then the resulting value is b
- If c is any other value (i.e., c is between the bounds of a and b), then the resulting value is c

259. Turning again to Fu’s equation to derive delta (the mathematical equivalent of which is reproduced below): the “a” value of Fu’s Clip3 function is -C; the “b” value is C; and the “c” value is the first offset expression (boxed in purple).

$$\text{delta} = \text{Clip3}(-C, C, ((6 \times (q_0 - p_0) - 2 \times (q_1 - p_1) + 8) \gg 4))$$

260. Applying the logic of Fu’s Clip3 function, if the result of Fu’s first offset expression is within the bounds of -C and C, then “delta” is the result of Fu’s first offset expression. In that instance, the result of the first offset would not be altered, and therefore, would be the value of delta.

261. If one disregards Fu’s clipping function (i.e., considering a scenario where the result of the first offset expression is within the bounds of -C and C), the remaining first offset expression is compared to the first offset expression recited in element 1[a] below.

Reference	First Offset Equation
’659 Patent	$(9 \times (q_0 - p_0) - 3 \times (q_1 - p_1) + 8) \gg 4$
Fu	$(6 \times (q_0 - p_0) - 2 \times (q_1 - p_1) + 8) \gg 4$

262. There is one difference between Fu’s first offset expression and the first offset expression recited in element 1[a]—the coefficients applied to the $(q_0 - p_0)$ and $(q_1 - p_1)$ terms are different. Although these coefficients are different numerical values, they are both a 3:1 ratio, such that they apply the same relative weight to each term. In other words, Fu’s first offset equation places three times the weight

on the $(q_0 - p_0)$ term, compared to the $(q_1 - p_1)$ term, just as the first offset equation of element 1[a] does. The coefficients of the first offset equation in both Fu and the '659 Patent maintain a ratio of 3, which will preserve (and, therefore, not filter) a ramp signal consistent with the '659 Patent's disclosure of an objective of the claimed deblocking filter. Ex-1001, 8:57-60 ("The present deblocking filter is configured not to modify the pixel values over a block boundary if these pixel values form a ramp but smooth the pixel values if they instead are in the form of a step.").

263. A POSITA would have appreciated, however, that the art was filled with extremely similar offset equations for deblocking filters that varied only as to two variables: (1) the coefficients applied to each difference in pixel values: $X(q_0 - p_0)$; $Y(q_1 - p_1)$; and/or $Z(q_2 - p_2)$; and (2) the number of pixel differences— $(q_0 - p_0)$, $(q_1 - p_1)$, $(q_2 - p_2)$, etc.—incorporated into the equation. These two variables were known to control the frequency response of the filter, or, in other words, how much particular frequencies would be filtered. *See, e.g.,* Gonzalez, 153, 273. By modifying these two variables, therefore, a POSITA could produce the desired balance between how much blocking artifacts would be filtered and maintaining overall image quality (by not over-blurring block boundaries). And because the prior art taught numerous similar offset equations that differed only as to these two variables, in my opinion, a POSITA would have considered these two variables

points of routine experimentation, and therefore would have been motivated to experiment with these variables in Fu's offset equation.

264. Those involved in the image-processing arts had been coalescing around a similar form of offset equation as that disclosed by Fu.

265. For instance, B118 discloses a deblocking filter that incorporates a specific offset equation for the two pixels (p0/q0, which B118 calls p1/q1) closest to the block boundary:

$$\Delta = \text{Clip}(-C, C, ((q_1 - p_1) \ll 2 + (p_2 - q_2) + 4) \gg 3)$$

B118, 48. The "delta" or offset equation could be re-written as follows using p0/q0 convention:

$$\Delta = \text{Clip}(-C, C, ((q_0 - p_0) \ll 2 + (p_1 - q_1) + 4) \gg 3)$$

ignoring the clipping function for now:

$$\Delta = ((q_0 - p_0) \ll 2 + (p_1 - q_1) + 4) \gg 3$$

substituting the left shift of (q0 - p0):

$$\Delta = (4 * (q_0 - p_0) + (p_1 - q_1) + 4) \gg 3$$

substituting the right shift of 4 * (q0 - p0) + (p1 - q1) + 4:

$$\Delta = \frac{(4 * (q_0 - p_0) + (p_1 - q_1) + 4)}{8}$$

and assuming rounding to the nearest integer:

$$\Delta = \frac{4(q_0 - p_0) + (p_1 - q_1)}{8} = \frac{4(q_0 - p_0) - (q_1 - p_1)}{8}$$

See generally Section V.B.3.b (describing conversion of bit-shift form to division form).

266. Similarly, the H.264 standard uses the following offset equation:

$$\Delta_{p_0, q_0} = \text{Clip3}(-tC, tC, (((q_0 - p_0) \ll 2) + (p_1 - q_1) + 4) \gg 3))$$

H.264, 207, 227. This equation can be re-written in the following form:

$$\Delta = \frac{4(q_0 - p_0) + (p_1 - q_1)}{8}$$

267. Bjontegaard—which was a proposal during the pre-development of the H.264 standard—proposed the following offset equation:

$$\text{Diff} = (3(c - f) - 8(d - e))/16$$

Bjontegaard, 20. This equation can be re-written in the following form:

$$\Delta = \frac{8(q_0 - p_0) - 3(q_1 - p_1)}{16}$$

268. Similarly, as part of the Video Coding Experts Group, a proposal (M21) was provided that offered the following offset equation:

$$\text{Difference} = \text{clip}(32 * (E - D) + 10 * (C - F) + 32) \gg 6);$$

Ex-1069, 2. This equation can be re-written in the following form:

$$\Delta = \frac{32(q_0 - p_0) - 10(q_1 - p_1)}{64}$$

269. Although each of these offset-filter equations used four pixels as inputs, other prior-art offset-filter equations expanded the number to six pixels.

270. For instance, the H.241 standard uses the following offset equation:

$$\Delta = \text{Clip3}(-t_c, t_c, ((q0_i + (p2_i + q1_i) >> 1)) >> 1 - ((p0_i + (q2_i + p1_i) >> 1)) >> 1))$$

H.241, 29. This equation can be re-written in the following form:

$$\Delta = \frac{2(q0 - p0) - (q1 - p1) + (q2 - p2)}{4}$$

This same equation also appears in Bjontegaard '048. Ex-1025, ¶ 47.

271. Similarly, B205 proposed using the following offset equation:

$$\Delta = \text{Clip}(-t_c, t_c, (13*(q0_i - p0_i) + 4*(q1_i - p1_i) - 5*(q2_i - p2_i) + 16) >> 5))$$

B205, 131. This equation can be re-written in the following form:

$$\Delta = \frac{13(q0 - p0) + 4(q1 - p1) - 5(q2 - p2)}{32}$$

272. For ease of readability and comparison, I have summarized each of the above offset-filter equations, which I have normalized so that all equations are divided by 16, as well as the offset-filter equation identified in the '659 Patent, claim 1. The differences between the equations are identified in color:

Table 1: Condensed Offset Equation	
'659 Patent	$\frac{9(q_0 - p_0) - 3(q_1 - p_1)}{16}$
Fu	$\frac{6(q_0 - p_0) - 2(q_1 - p_1)}{16}$
H.264, List	$\frac{8(q_0 - p_0) - 2(q_1 - p_1)}{16}$
B118	$\frac{8(q_0 - p_0) - 2(q_1 - p_1)}{16}$
Bjontegaard	$\frac{8(q_0 - p_0) - 3(q_1 - p_1)}{16}$
VCEG-M21	$\frac{8(q_0 - p_0) - 2.5(q_1 - p_1)}{16}$
Bjontegaard '048	$\frac{8(q_0 - p_0) + 4(q_1 - p_1) - 4(q_2 - p_2)}{16}$
B205	$\frac{7.5(q_0 - p_0) + 2(q_1 - p_1) - 2.5(q_2 - p_2)}{16}$

These equations can also be expanded, applying each coefficient to each pixel, as follows:

Table 2: Expanded Offset Equation	
'659 Patent	$\frac{9q_0 - 9p_0 - 3q_1 + 3p_1}{16}$
Bjontegaard	$\frac{8q_0 - 8p_0 - 3q_1 + 3p_1}{16}$
B-118	$\frac{8q_0 - 8p_0 - 2q_1 + 2p_1}{16}$
Fu	$\frac{6q_0 - 6p_0 - 2q_1 + 2p_1}{16}$
H.264, List	$\frac{8q_0 - 8p_0 - 2q_1 + 2p_1}{16}$
VCEG-M21	$\frac{8q_0 - 8p_0 - 2.5q_1 + 2.5p_1}{16}$
H.241 Bjontegaard '048	$\frac{8q_0 - 8p_0 + 4q_1 - 4p_1 + 4q_2 - 4p_2}{16}$
B205	$\frac{7.5q_0 - 7.5p_0 + 2q_1 - 2p_1 - 2.5q_2 + 2.5p_2}{16}$

273. Because these are just the offset equations that are used to calculate *how much* to modify pixels p_0 and q_0 , however, I have also provided below the full filter equations ($p_0' = p_0 + \Delta$) that would be used to calculate the modified value of p_0 itself by adding the offsets above to p_0 (or, more specifically, $\frac{16p_0}{16}$):

Table 3: Full Filter Equation (adding 16p0/16)	
'659 Patent	$\frac{9q_0 + 7p_0 - 3q_1 + 3p_1}{16}$
Bjontegaard	$\frac{8q_0 + 8p_0 - 3q_1 + 3p_1}{16}$
B-118	$\frac{8q_0 + 8p_0 - 2q_1 + 2p_1}{16}$
Fu	$\frac{6q_0 + 10p_0 - 2q_1 + 2p_1}{16}$
H.264, List	$\frac{8q_0 + 8p_0 - 2q_1 + 2p_1}{16}$
VCEG-M21	$\frac{8q_0 + 8p_0 - 2.5q_1 + 2.5p_1}{16}$
H.241 Bjontegaard '048	$\frac{8q_0 + 8p_0 + 4q_1 - 4p_1 + 4q_2 - 4p_2}{16}$
B205	$\frac{7.5q_0 + 8.5p_0 + 2q_1 - 2p_1 - 2.5q_2 + 2.5p_2}{16}$

As shown, only the coefficient applied to p0 changes compared to the equations in Table 2.

274. As demonstrated above, and particularly as shown in Table 1, each of these offset equations followed similar patterns that differed only as to the two variables I identified above, and demonstrate the workable ranges for each of these variables. When experimenting with these two variables to design an offset equation, however, a POSITA would have also observed and been guided by several additional design constraints apparent from the above equations.

275. **First**, as demonstrated in Table 1, the magnitude of the coefficient applied to (q0-p0) would generally be 2-4 times larger than the coefficient applied

to $(q_1 - p_1)$ and $(q_2 - p_2)$, if included. In other words, the difference between the pixel values closest to the block boundary would be given more weight when calculating the offset. The ratio between these coefficients was also known to control the “width” of the filter, e.g., how wide the filtering curve is in the frequency domain. *See* Gonzalez, 153, 273. F_u and the first offset equation of element $1[a]$ have the same 3/1 ratio. A higher ratio between these coefficients introduces higher frequency components, which leads to a “wider” frequency response.¹² Ex-1072, 7 (“[L]arge value coefficients carry more weight in the construction of output samples,” and providing examples in which weighting the center coefficients differently relative to more-distant coefficients changes the width of the filter”). The “wider” frequency response filter will consequently attenuate sharp transitions in the image less, and thus better preserve edges in the image.

276. **Second**, summing the coefficients (in Table 1) in the numerator and dividing by the denominator would produce a fraction between $\frac{1}{4}$ and $\frac{1}{2}$. *See, e.g.*, F_u (summing $(6 - 2)$ and dividing by 16 which results in $\frac{1}{4}$); Bjontegaard (summing $(8 - 3)$ and dividing by 16 which results in $\frac{5}{16}$); Ex-1001 (summing $(9 - 3)$ and

¹² For example, a unit impulse (*i.e.*, a signal which has a value of one at the origin and zero everywhere else) has a flat frequency response (*i.e.*, the frequency response is constant for all frequencies), and vice versa.

dividing by 16 which results in $3/8$). This meant that the total amount the p_0 and q_0 pixel values would be modified would be approximately $1/4$ to $1/2$ of the differences in pixel values on each side of the boundary, depending on the amount of filtering desired. This ratio ensures that pixels would not be over-filtered. For instance, if the total amount of filtering exceeded $1/2$ the difference between p_0 and q_0 , then if p_0 started with a value lower than q_0 , it would end up producing a value higher than q_0 after filtering. The expectation of deblocking filtering, however, is that it will cause p_0 and q_0 to be closer together, and not to have their relationship invert. *See, e.g.,* Ex-1039, 215.

277. **Third**, the “window” used to calculate the offset (Tables 1 and 2) was typically either four (q_0-p_0 ; q_1-p_1) or six (q_0-p_0 ; q_1-p_1 ; q_2-p_2) pixels long. The prior-art equations typically used 6 pixels or less not only because more pixels increased the computing resources required, but because longer filters tended to introduce a “ringing” effect into the filtered image. This is a known aspect of filter design and can occur when a low-pass filter first overshoots its target output and subsequently overcompensates and thus undershoots its target. *See, e.g.,* Ex-1073, 3 (“The longer the impulse response length is, the wider the ringing noise spreads.”). Over time, those in the art had found empirically that using six pixels or less in an offset equation tended to avoid significant ringing.

278. **Fourth**, it was known that the “DC” gain of the offset should equal 1. *See, e.g.*, Gonzalez, 153-54 (noting “[t]he denominator ... is simply the sum of the mask coefficients”); Ex-1072, 7 (explaining that “[s]etting the coefficients [to] sum to 1 makes a filter with unity gain at DC”). This can be conceptualized more easily in audio-signal processing: audio signals are often filtered to eliminate specific frequency components to boost the *apparent* loudness of other frequencies, without changing the overall loudness (or gain) of *all* frequencies. In image processing, gain controls the “brightness” of the pixels (how light or dark they appear), and when performing deblocking filtering, it was understood that the goal is to blur the blocking artifacts by eliminating the target frequencies without altering the overall brightness of the pixels. Keeping a “DC” gain of 1 means that if a “flat” signal of 1 is input into to the offset equations, the offset produces a value of zero. This is accomplished by ensuring the coefficients in the numerator of the offset equation (Table 2 above) sum to zero. Gonzalez, 696, 699. Relatedly, when a “flat” signal of 1 is input into the full filter equation (*e.g.*, $p1' = p1 + \text{offset}$), the output should stay 1. This is accomplished by ensuring the coefficients in the filter equation (Table 3 above), when summed, equal the denominator so that they divide into 1. Gonzalez, 153-54. As shown in Table 3, any coefficient applied to pixel $p1$ and, if used (*e.g.*, B205 and H.241), pixel $p2$ is cancelled out by the corresponding coefficient for pixels $q1$ and $q2$, respectively, and then summing the coefficients of $p0$ and $q0$

produces a value of 16, which is then divided by 16 to produce a final ratio of 1. Although this design consideration is not of particular concern when coefficients are applied to pixel differences (*e.g.*, $X(q_0 - p_0)$) because the coefficient applied to one pixel will always be canceled by the coefficient applied to the other pixel (*e.g.*, $Xq_0 - Xp_0$), it is still a relevant design consideration for non-difference-formatted equations, such as for p_1/q_1 offset equations I address below.

279. **Fifth**, while there are exceptions, the value of the right-shift was generally between 3 and 6 (meaning the offset is divided by a power of 2^n between 8 and 64). A higher right-shift value would allow for more-precise filter design because the coefficients and their ratios could be controlled to a greater degree. For instance, VCEG-M21 relied on one extreme and used a right-shift of 6, thus providing more control over the ratio between its coefficients. On the other extreme is H.241 (that specified that it was intended to be a low-complexity deblocking filter, H.241, 1), which provided a right shift of 1, thus providing little room to experiment with different coefficient ratios. However, right shifts greater than 6 were rare because that required extra bits to represent the coefficients.

280. **Finally**, integer coefficients (not decimals) are used to minimize the computational complexity of the calculations. That is also why right shifts are used for offset equations instead of using decimal coefficients applied to a pixel (or pixel differences, like the first-offset equations).

281. In my opinion, these prior-art first-offset expressions demonstrate that those in the art were routinely experimenting with the coefficients and number of pixels used to control the frequency response of their filters. Even the inventors recognized in another patent that the equation proposed in the '659 Patent is not essential, as it explains that “[d]ifferent such functions are possible and can be used.” Ex-1063, 18:1-24; *see also id.*, 17:1-8.

282. A POSITA would have experimented with Fu’s equation (as those in the art had routinely experimented with the coefficients and filter length, usually through an empirical trial-and-error process), and specifically with other coefficient values that were close to those and had a similar ratio in Fu, to achieve a desired amount of filtering of blocking artifacts without also degrading the overall image quality. A POSITA would have experimented with different coefficient values within a workable range Fu and others had found throughout the prior art. A POSITA would have been driven to experiment in view of competing demands for filtering video, such as filter quality versus computational complexity or performance of the filter across a range of different scenarios. And, in my opinion, a POSITA would have found the claimed offset equation obvious because the coefficients (9 and 3) were nothing but a scaled version of Fu’s coefficients (6 and 2) by 50%. A POSITA would have expected only slight differences in filtering

performance, as shown below, through some example applications of the two equations (rounding to the nearest integer):

p2	p1	p0	q0	q1	Type	$\frac{9(q0 - p0) - 3(q0 - p0)}{16}$	$\frac{6(q0 - p0) - 2(q1 - p1)}{16}$
200	150	150	150	150	Step (p1)	0	0
200	200	150	150	150	Step (p0)	9	6
200	200	200	150	150	Step (q0)	-19	-13
150	160	170	180	190	Ramp (10)	0	0
125	150	175	200	225	Ramp (25)	0	0
200	250	200	250	200	Alternating	38	25
255	50	50	255	255	Line	77	51
255	255	50	50	255	Line	0	0
175	200	150	150	175	Random	5	3
50	125	100	150	125	Random	28	19
75	100	150	75	125	Random	-47	-31

283. A POSITA would have appreciated that the coefficients 9 and 3 would produce an offset that was *slightly* stronger than Fu's, as Fu would apply approximately $\frac{1}{4}$ offset to the p0/q0 pixels, whereas the coefficients 9 and 3 would apply approximately $\frac{3}{8}$ offset. Moreover, a POSITA would have recognized that the coefficients 9 and 3 are just one or two values different than those used by Bjontegaard's (8 and 3) and H.264's (8 and 2). And a POSITA would have

appreciated that the coefficients 9 and 3 met every general design requirement for such offset filters: (1) it maintained a ratio of 3 that is between 2 and 4 for $X(q_0 - p_0)$; $Y(q_1 - p_1)$; (2) it would produce an offset of approximately $\frac{3}{8}$, just slightly more than Fu's original offset equation of $\frac{1}{4}$, and still between $\frac{1}{4}$ and $\frac{1}{2}$; (3) it used four pixels; and (4) it maintained a DC gain of 1 (as shown in Tables 2 and 3).

284. To the extent the Patent Owner argues that selecting the numerator coefficients 9 and 3, and denominator coefficient of 16 would have been non-obvious because it produced unexpected results, I would disagree.

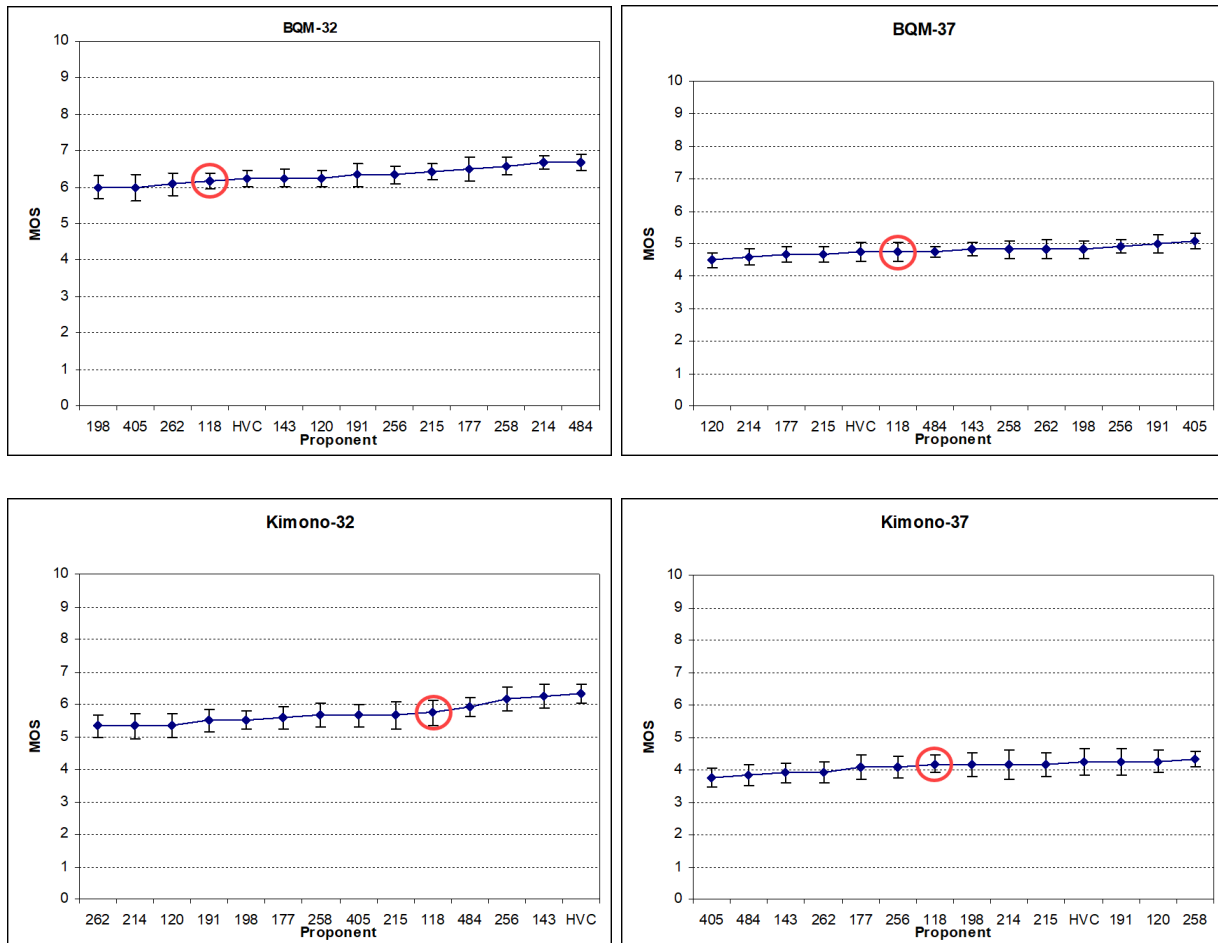
285. I understand that the claimed filter equation of $\frac{9(q_0 - p_0) - 3(q_1 - p_1)}{16}$ as well as the specific offset equations for the p_1 and q_1 pixels addressed in claim 4, below—had been proposed by the inventors for inclusion in the HEVC standard. Ex-1043, 2. I also understand that others had suggested using different offset-filter equations for the deblocking filter. I have summarized some of the alternative offset equations other standards participants had proposed below, all of which have been normalized to be divided by 16 and written in standard convention of $q_j - p_j$, with the differences between the equations colored.

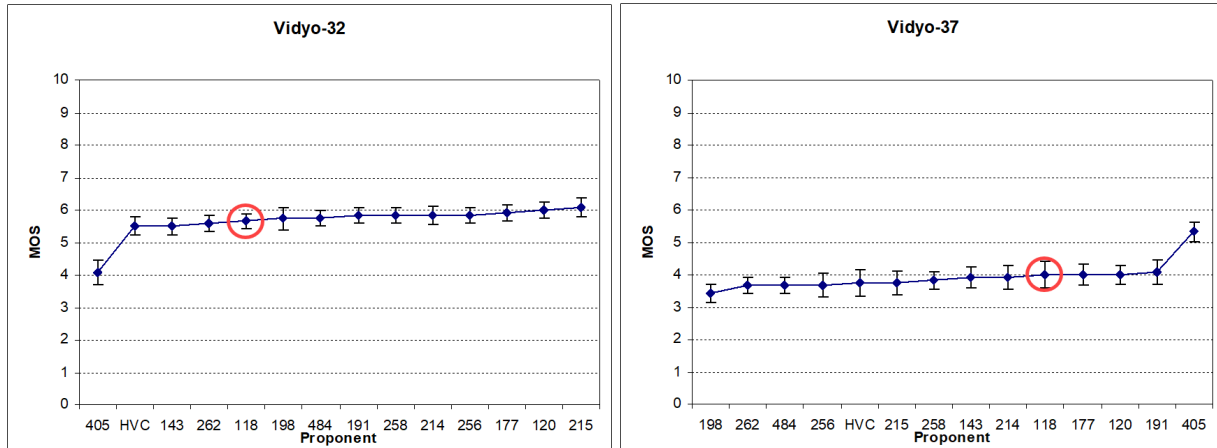
JCT-VC Documents	Filter Equations (Normalized)
'659 Patent	$\frac{9(q_0 - p_0) - 3(q_1 - p_1)}{16}$
F120	$\frac{6(q_0 - p_0) - 1.5(q_1 - p_1)}{16}$
F256	$\frac{6.5(q_0 - p_0) + 2(q_1 - p_1) - 2(q_2 - p_2)}{16}$
F258	$\frac{8(q_0 - p_0) + 2(q_1 - p_1)}{16}$

Ex-1044, 2; Ex-1045, 6; Ex-1046, 5.¹³

¹³ It appears as if the filter equation proposed in F-258 contains a typographical error, and the second coefficient for $Y(q_1 - p_1)$ should be “-2” instead of “+2.” Specifically, F-258 states that “a normal filter of MPEG-4 Part.10 AVC/H.264 is used for weak filtering as follows: $\Delta = \text{Clip3}(-tc, tc, ((q_0 - p_0) \ll 2) + (q_1 - p_1) \gg 3)$,” Ex-1046, 5. But H.264 recites the equation: $\Delta = \text{Clip3}(-tc, tc, ((q_0 - p_0) \ll 2) + (p_1 - q_1) \gg 3)$. Ex-1037, 207. By mistakenly switching the order of operation for p_1 and q_1 , that switches the polarity of the coefficient. As I noted previously, H.264’s equation written in division form is: $\frac{4(q_0 - p_0) - (q_1 - p_1)}{8}$, or $\frac{8(q_0 - p_0) - 2(q_1 - p_1)}{16}$. Moreover, using “+” instead of “-” would mean that the p_0/q_0 would be at risk of over-filtering because the total offset would be approximately 10/16ths of the difference in pixels over the boundary.

286. Those involved in the standardization process ultimately performed subjective-quality tests of thirteen different deblocking-filter proposals (compared to one “anchor,” used as a baseline) with a variety of different offset equations. Ex-1047, 1. Below are the results of those comparisons, which involved three different test-video sequences (called BQM, Kimono, and Vidyo3), using two different encoding conditions for each proposal. *Id.*, 1. The deblocking filter that incorporated the claimed offset equations is circled in red.





Ex-1047, 3-8 (red circles added).

287. The test demonstrated that “for most of the cases under test the performance of the proposals was equal to that of the anchor,” and only in very limited circumstances did a few proposals materially deviate. Ex-1047, 8. Ultimately, the group recommended that the “luma part” of the deblocking filter proposed by Ericsson should be adopted because “luma filter design and separate on/off decision for P/Q edge side” produced a “1.2% gain [in deblocking efficiency] without subjective quality loss.” Ex-1048, 176. In other words, the exact offset equation used had no material impact on the perceptual quality of the deblocking filter, but the inventor’s proposal was adopted because it resulted in a 1.2% increase in deblocking efficiency, which had largely been attributed to its asymmetric filtering decision that existed in the prior art.

288. Even if the 1.2% deblocking efficiency gain could be attributed to the offset equations themselves, I consider that a difference in degree, and not kind,

because, in my opinion, a POSITA would have considered such small efficiency gains relatively minor, and one that would be expected when optimizing the parameters of prior art filter equations. In my opinion, those involved in the standardization process did not recognize any unexpected, material transformation provided by those equations compared to what existed in the prior art.

289. To the extent Patent Owner argues that the prior art taught away from the claimed invention, I have not seen such evidence. I reserve the right to respond if Patent Owner identifies such alleged evidence.

290. To the extent Patent Owner argues that the coefficients for the offset equations or number of pixels used were not recognized as result-effective variables, I would disagree with such an argument. Based on my understanding of the art, these equations were routinely experimented with based on trial-and-error testing to produce a result—the amount block boundaries were filtered without degrading overall image quality—that most satisfied the designer. Indeed, the fact that those in the video-coding community proposed various equations with slightly different coefficients or number of pixels incorporated into the equations demonstrates that those in the art had recognized these variables as result-effective, otherwise they would have all used the same equation and experimented only with other aspects of the deblocking filter.

291. Finally, to the extent Patent Owner argues that there were too many options to choose from that would not invite routine experimentation, I would disagree with that argument. As I had mentioned above, those involved in deblocking-filter design understood the general parameters within which a deblocking filter should operate, which greatly limited the available options from which to choose (e.g., maintaining a ratio between coefficients to ensure proper filter performance; not using too many pixels for efficiency reasons and to avoid introducing ringing artifacts; not using large coefficients for efficiency, etc.). Moreover, the fact that those in the art were routinely experimenting with and proposing different (but similar) coefficients reinforces that the options were not so numerous so as to discourage routine experimentation. Instead, those skilled in the art were actively engaged in such experimentation at the relevant time.

292. For these reasons, in my opinion, the offset equation ultimately selected by the inventors would have been a result of routine experimentation and optimization by modifying known result-effective variables, and therefore the recited equation would have been obvious to a POSITA at the time of the invention.

- c. **Element 1[b]: modifying said pixel value p0 of said pixel closest to said block boundary in said line of pixels by adding said first offset to said pixel value p0 of said pixel closest to said block boundary in said line of pixels to form a modified pixel value p0' of said pixel closest to said block boundary in said line of pixels; and**

293. In my opinion, Fu in view of a POSITA's knowledge discloses or suggests element 1[b].

294. Fu discloses that the modified value of pixel p0, which as shown in Figure 2 of Fu below (boxed in red) is the pixel closest to the block boundary in the line of pixels from the p block, is derived by adding the result of the first offset to the value of pixel p0.

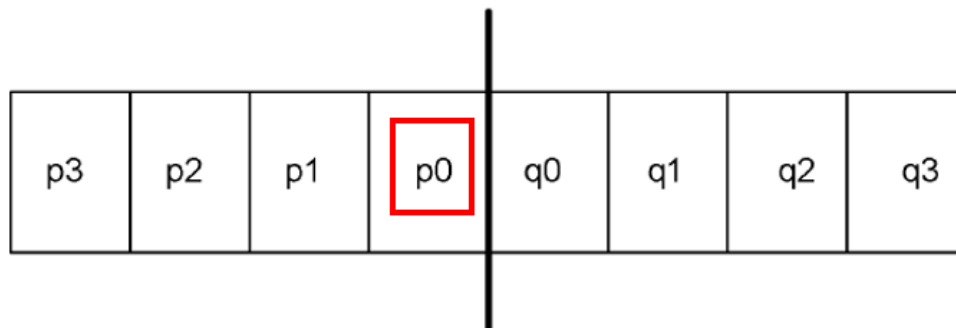


Figure 2

295. Fu's equation to calculate the modified value of p0 (i.e., P0, *see* Fu, 8:5 ("P0 and Q0 are respectively filtered values of p0 and q0")) is reproduced below:

$$P0 = Clip1(p0 + delta)$$

Fu, 8:7-11.

296. As discussed above in Section X.A.1.b, delta is the result of Fu's first offset equation. The result is P0, which is the modified pixel value of p0 (i.e., p0' in terms of element 1[b]), which is the pixel closest to the block boundary in the line of pixels p0, p1, p2, p3.

297. Fu discloses that the result of $p0 + \text{delta}$ is clipped by Fu's Clip1 function. Fu, however, still discloses element 1[b]. Fu's Clip1 function is expressed as $\text{Clip1}(x) = \text{Clip3}(0, 255, x)$. Fu, 8, 10-11. Applying the logic of Fu's Clip3 function, the following conditions are recognized:

- If the result of $p0 + \text{delta}$ is less than 0; then P0 (i.e., the modified value of p0) is 0;
- If the result of $p0 + \text{delta}$ is greater than 255, then P0 is 255;
- If the result of $p0 + \text{delta}$ is within the range of 0-255, then P0 is $p0 + \text{delta}$.

298. In the third condition above, the modified value of pixel p0 (in Fu, P0) is the sum of the result of the first offset expression and pixel value p0.

- d. **Element 1[c]: modifying said pixel value q_0 of said pixel closest to said block boundary in said corresponding line of pixels by subtracting said first offset from said pixel value q_0 of said pixel closest to said block boundary in said corresponding line of pixels to form a modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels.**

299. In my opinion, Fu in view of a POSITA's knowledge discloses or suggests element 1[c].

300. Fu discloses that the modified value of pixel q_0 , which as shown in Figure 2 of Fu below is the pixel closest to the block boundary in the corresponding line of pixels of the neighboring block of pixels, is derived by subtracting the result of the first offset from the value of q_0 .

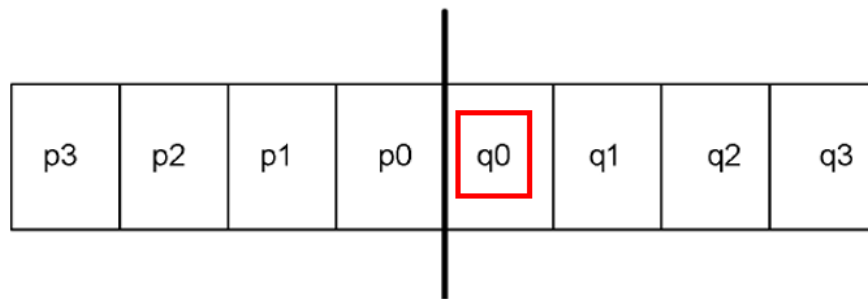


Figure 2

301. Fu's equation to calculate the modified value of q_0 (i.e., Q_0 , see Fu, 8:5 ("P0 and Q0 are respectively filtered values of p0 and q0")) is reproduced below.

$$Q_0 = \text{Clip1}(q_0 - \text{delta})$$

Fu, 8:7-11.

302. As discussed above in Section X.A.1.b, delta is the result of Fu's first offset equation. Q0 is the modified pixel value of q0 (i.e., q0' in terms of element 1[c]), which is the pixel closest to the block boundary in the corresponding line of pixels q0, q1, q2, q3.

303. Fu discloses that the result of $q0 - \text{delta}$ is clipped by Fu's Clip1 function. Fu, however, still discloses element 1[c]. Fu's Clip1 function is expressed as $\text{Clip1}(x) = \text{Clip3}(0, 255, x)$. Fu, 8:10-11. Applying the logic of Fu's Clip3 function, the following conditions are recognized:

- If the result of $q0 - \text{delta}$ is less than 0; then Q0 (i.e., the modified value of q0) is 0;
- If the result of $q0 - \text{delta}$ is greater than 255, then Q0 is 255;
- If the result of $q0 - \text{delta}$ is within the range of 0-255, then Q0 is $q0 - \text{delta}$.

304. In the third condition above, the modified value of pixel q0 (in Fu, Q0) is the difference between the result of the first offset expression and pixel value q0.

2. Claim 2

305. As discussed above in Section X.A.1, Fu in view of the knowledge of a POSITA renders claim 1 obvious.

- a. **Element 2[a]: The method according to claim 1, further comprising clipping said first offset to be within an interval of $-t_c$ and t_c by setting said first offset to be equal to $-t_c$ if said first offset is smaller than $-t_c$ and by setting said first offset to be equal to t_c if said first offset is larger than t_c ,**

306. In my opinion, Fu discloses or suggests element 2[a].

307. As discussed above in Section X.A.1.b, Fu teaches that its first offset is clipped using Fu's Clip3 function. Fu's equation to derive its first offset, delta, is reproduced below, where Fu's clipping threshold values (-C, C) are colored purple:

$$\text{delta} = \text{Clip3}(-C, C, ((q0 - p0) \times 3 + (p1 - q1) + 4) \gg 3))$$

Fu, 8:7.

308. Fu's Clip3(a,b,c) function is defined by the following conditions:

- If c is less than a, then the resulting value is a
- If c is greater than b, then the resulting value is b
- If c is any other value (i.e., c is between the bounds of a and b), then the resulting value is c

309. Turning again to Fu's equation to derive delta (the mathematical equivalent of which is reproduced below): the "a" value of Fu's Clip3 function is -C; the "b" value is C; and the "c" value is the first offset equation (boxed in purple).

$$\text{delta} = \text{Clip3}(-C, C, ((6 \times (q0 - p0) - 2 \times (q1 - p1) + 8) \gg 4))$$

Fu, 8:5-11.

310. Applying the logic of Fu's Clip3 function, if the result of Fu's first offset expression is less than $-C$ (or $-t_c$, in terms of element 2[a]), then "delta" is the set to $-C$. Conversely, if the result of Fu's first offset expression is greater than C (or t_c in terms of element 2[a]), then "delta" is set to C .

b. Element 2[b]: wherein t_c is a threshold value that is dependent on a quantization parameter value assigned to said block.

311. In my opinion, Fu discloses or suggests element 2[b].

312. In the '659 Patent, it teaches that " t_c is a threshold value that is dependent on a quantization parameter value assigned to the block." Ex-1001, 4:1-7.

313. As discussed in Section X.A.2.a, Fu teaches that its first offset is clipped by Fu's Clip3 function, reproduced below, to derive the value of delta.

$$\text{delta} = \text{Clip3}(-C, C, ((q0 - p0) \times 3 + (p1 - q1) + 4) \gg 3))$$

314. In Fu's Clip3 function, $-C$ and C (colored purple above) are threshold values because Fu teaches that delta will equal $-C$ if the result of the first offset expression is less than $-C$; Fu teaches that delta will equal C if the result of the first offset expression is greater than C . Fu, 8:5-11 (disclosing Clip3 (a, b, c,) as equaling a ; $c < a$, b ; $c > b$, c ; other).

315. Fu further teaches that its filtering clipping parameter C is dependent on the quantization parameter value assigned to the block.

316. First, Fu teaches that “C is called the filtering and clipping parameter” and that C has a relationship to value IndexA, as shown in Fu’s Table 3 (Fu, 8:21-25), which is reproduced below:

Index	<i>C</i>	Index	<i>C</i>	Index	<i>C</i>	Index	<i>C</i>
0	0	16	1	32	2	48	5
1	0	17	1	33	2	49	5
2	0	18	1	34	2	50	5
3	0	19	1	35	2	51	6
4	0	20	1	36	2	52	6
5	0	21	1	37	2	53	6
6	0	22	1	38	3	54	7
7	0	23	1	39	3	55	7
8	0	24	1	40	3	56	7
9	0	25	1	41	3	57	7
10	0	26	1	42	3	58	8
11	0	27	1	43	3	59	8
12	0	28	1	44	3	60	8
13	0	29	1	45	4	61	9
14	0	30	2	46	4	62	9

Fu, 9.

317. Fu further teaches that IndexA is derived from the following equation:

$$IndexA = Clip3(0, 63, QP_{av} + AlphaCOffset)$$

Fu, 5:20. Fu teaches that IndexA is the “AlphaCOffset” calculation lookup table index value. Fu, 5:15-19.

318. Fu also teaches that index value IndexA is derived from QP_{av} , where “the average QP_{av} may be first calculated based on the quantization parameters (QPs) of the two blocks,” Fu, 2:24-30, and where “ QP_{av} is used to determine the table lookup index *IndexA*.” *Id.* Fu teaches that “[t]he QP of each block is the data known when the block is quantized, and QP_{av} may be expressed as[:.]”

$$QP_{av} = (QP_p + QP_q + 1) \gg 1$$

Fu, 3:3-5.

319. As discussed in Section X.A.1, Fu discloses two blocks—a p block and q block. In Fu, the p block can be considered “the block,” whereas the q block is the neighboring block. Fu teaches that its quantization parameter is assigned to the two blocks—p block and q block. Fu, 2:24-30; *see also id.*, 3:3-4 (“The QP of each block is the data known when the block is quantized, and QP_{av} may be expressed as formula 2 [reproduced above].”).

3. Claim 3

320. As discussed above in Section X.A.1, Fu in view of the knowledge of a POSITA renders claim 1 obvious.

- a. **Element 3[a]: The method according to claim 1, further comprising: clipping said modified pixel value p0' of said pixel closest to said block boundary in said line of pixels to be within an interval of zero and a defined maximum value by setting said modified pixel value p0' of said pixel closest to said block boundary in said line of pixels to be equal to zero if said modified pixel value p0' of said pixel closest to said block boundary in said line of pixels is smaller than zero and by setting said modified pixel value p0' of said pixel closest to said block boundary in said line of pixels to be equal to said defined maximum value if said modified pixel value p0' of said pixel closest to said block boundary in said line of pixels is larger than said defined maximum value; and**

321. In my opinion, Fu in view of a POSITA's knowledge discloses or suggests element 3[a].

322. Fu teaches that its modified pixel value P0 (i.e., p0' in terms of element 3[a]) is clipped. As discussed in Section X.A.1 and shown in Fu's Figure 2, reproduced below, P0 (i.e., p0') is the modified value of pixel p0, which is the pixel closest to the block boundary in the line of pixels p0, p1, p2, p3.

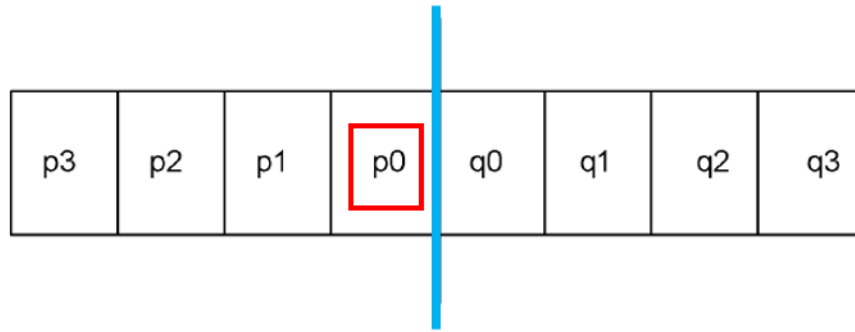


Figure 2

323. Fu teaches that P0 is calculated using Fu's Clip1 function, shown below:

$$P0 = \text{Clip1}(p0 + \text{delta})$$

where Fu teaches that Clip1(a) is expressed as $\text{Clip1}(x) = \text{Clip3}(0, 255, x)$. Fu, 8:7-11.

324. Fu's Clip1 function is expressed as $\text{Clip1}(x) = \text{Clip3}(0, 255, x)$. Fu, 8:10-11. Applying the logic of Fu's Clip3 function, the following conditions are recognized:

- If the result of $p0 + \text{delta}$ is less than 0; then P0 (i.e., the modified value of p0) is 0;
- If the result of $p0 + \text{delta}$ is greater than 255, then P0 is 255;
- If the result of $p0 + \text{delta}$ is within the range of 0-255, then P0 is $p0 + \text{delta}$.

325. Here, as discussed in Section V.B.1, 255 is defined by the maximum value that an 8-bit pixel can be (2^8-1). Fu, therefore, discloses to a POSITA that 255 is a defined maximum.

326. As another example, B205 teaches that 255 is a common defined maximum value of a clip function for pixels. Ex-1001, 2:58-3:3; Section V.B.1 (discussing pixel values). Similarly, the '659 Patent discloses that for its clipping function: “The parameter M denotes a defined maximum value that the modified pixel values can assume. In a particular embodiment, this maximum value is defined based on the number of bits that are spent on the pixel values. Thus, assume that the pixel values are in the form of a m-bit value, then M is preferably equal to 2^m-1 . For instance, if each pixel value is a 8-bit value, i.e., $m=8$, then $M=255$.” Ex-1001, 11:64-12:17.

- b. Element 3[b]: clipping said modified pixel value $q0'$ of said pixel closest to said block boundary in said corresponding line of pixels to be within an interval of zero and said defined maximum value by setting said modified pixel value $q0'$ of said pixel closest to said block boundary in said corresponding line of pixels to be equal to zero if said modified pixel value $q0'$ of said pixel closest to said block boundary in said corresponding line of pixels is smaller than zero and by setting said modified pixel value $q0'$ of said pixel closest to said block boundary in said corresponding line of pixels to be equal to said defined maximum if said modified pixel value $q0'$ of said pixel closest to said block boundary in said corresponding line of pixels is larger than said defined maximum value.**

327. In my opinion, Fu in view of a POSITA's knowledge discloses or suggests element 3[b].

328. Fu teaches that its modified pixel value $Q0$ (i.e., $q0'$ in terms of element 3[b]) is clipped. As discussed in Section X.A.1 and shown in Fu's Figure 2, reproduced below, $Q0$ (i.e., $q0'$) is the modified value of pixel $q0$, which is the pixel closest to the block boundary in the corresponding line of pixels $q0, q1, q2, q3$ of the neighboring block of pixels.

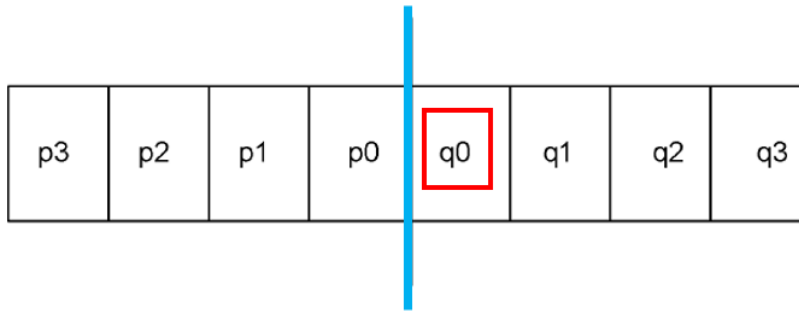


Figure 2

329. Fu teaches that Q0 is calculated using Fu's Clip1 function, shown below:

$$Q0 = \text{Clip1}(q0 - \text{delta})$$

where Fu teaches that Clip1(a) is expressed as $\text{Clip1}(x) = \text{Clip3}(0, 255, x)$. Fu, 8:7-11.

330. Fu's Clip1 function is expressed as $\text{Clip1}(x) = \text{Clip3}(0, 255, x)$. Fu, 8:10-11. Applying the logic of Fu's Clip3 function, the following conditions are recognized:

- If the result of $q0 - \text{delta}$ is less than 0; then Q0 (i.e., the modified value of q0) is 0;
- If the result of $q0 - \text{delta}$ is greater than 255, then Q0 is 255;
- If the result of $q0 - \text{delta}$ is within the range of 0-255, then Q0 is $q0 - \text{delta}$.

331. Here, as discussed in Section V.B.1, 255 is defined by the maximum value that an 8-bit pixel can be (2^8-1). Fu, therefore, discloses to a POSITA that 255 is a defined maximum.

332. For example, B205 teaches that 255 is a common defined maximum value of a clip function for pixels. Ex-1001, 2:58-3:3; Section V.B.1 (discussing pixel values). Similarly, the '659 Patent discloses that for its clipping function: “The parameter M denotes a defined maximum value that the modified pixel values can assume. In a particular embodiment, this maximum value is defined based on the number of bits that are spent on the pixel values. Thus, assume that the pixel values are in the form of a m-bit value, then M is preferably equal to 2^m-1 . For instance, if each pixel value is a 8-bit value, i.e., $m=8$, then $M=255$.” Ex-1001, 11:64-12:17.

4. Claim 4

333. As discussed above in Section X.A.1, Fu in view of the knowledge of a POSITA renders claim 1 obvious.

- a. **Element 4[a]: The method according to claim 1, further comprising: calculating a second offset to be based on $(p_2+p_0-2 \times p_1+2 \times \Delta)/4$, wherein p_2 denotes a pixel value of a pixel second next closest to said block boundary in said line of pixels and Δ denotes said first offset;**

334. In my opinion, Fu in view of a POSITA's knowledge discloses or suggests element 4[a].

335. Fu teaches that a second offset can be calculated to derive a modified value of pixel p_1 —the next closest pixel to the block boundary in the line of pixels p_0, p_1, p_2, p_3 .

336. Fu discloses that after p_0 and q_0 are filtered, a decision is made whether to filter p_1 and/or q_1 . In the first case, “when the block boundary is a chroma block boundary, do not filter p_1 and q_1 .” Fu, 8:14. In the second case, “when a_p is greater than 0 at the luma block boundary, filter p_1” Fu, 8:15-16.

337. Fu teaches that for process 3, “the definition of a_p and a_q is the same as that of process 1 and process 2” and that a_p is obtained from the following formula:

$$a_p = \text{Abs}(p_2 - p_0) < \beta \&\& \text{Abs}(p_0 - q_0) < ((\alpha \gg 2) + 2)$$

Fu, 5:13, 8:21-22.

338. Fu teaches that the modified value of p_1 (P_1) is derived from the following equation:

$$P_1 = \text{Clip1}(p_1 + \text{Clip3}(-C, C, ((P_0 - p_1) \times 3 + (p_2 - Q_0) + 4) \gg 3))$$

where the second offset expression is boxed in red above. Fu’s second offset expression is added to pixel value p_1 to calculate P_1 (i.e., p_1'). Fu, 8:15-17.

339. Removing Fu’s Clip3 and Clip1 functions for simplicity, Fu’s second offset expression is:

$$(((P_0 - p_1) \times 3 + (p_2 - Q_0) + 4) \gg 3)$$

340. As shown in green above, Fu's second offset expression relies on modified pixel values P0 and Q0, pixel value p1, as well as value p2, which is the second next closest pixel to the block boundary in the line of pixels p0, p1, p2, p3 shown in Fu Figure 2 below:

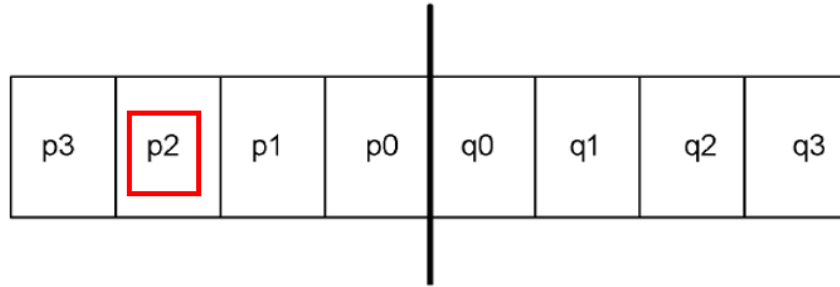


Figure 2

341. Fu includes the modified version of P0 and Q0 (i.e., p0' and q0' in terms of the '659 Patent) in its second offset equation, which can be replaced with their underlying equations $P0 = (Clip1(p0 + \text{delta}))$ and $Q0 = (Clip1(q0 - \text{delta}))$:

$$\begin{aligned}
 & \frac{3(p0 + \text{delta} - p1) - 1(q0 - \text{delta} - p2)}{8} \\
 &= \frac{(p2 - 3p1 + 3p0 - q0 + 4(\text{delta}))}{8} \\
 &= \frac{(p2 - 3p1 + 3p0 - q0)}{8} + \frac{2(\text{delta})}{4}
 \end{aligned}$$

342. As shown above, Fu combines a high-pass filter component $\frac{(p2 - 3p1 + 3p0 - q0)}{8}$ with a first-offset component $\frac{2(\text{delta})}{4}$ as does the second offset equation in element 4[a] of the '659 Patent.

343. A table comparing Fu’s second offset expression (dropping the rounding values¹⁴) and the second offset expression recited in element 4[a] is reproduced below:

Reference	Second Offset Equation
'659 Patent	$\Delta = \frac{1(p2 - 2p1 + p0)}{4} + \frac{2\Delta}{4}$
Fu	$\Delta = \frac{1(p2 - 3p1 + 3p0 - q0)}{8} + \frac{2\Delta}{4}$

344. As shown above, Fu’s second offset expression differs from the second offset expression recited in element 4[a] in two ways: (1) the exact high-pass filter used (in blue, above) and (2) the coefficient applied to that component (in red, above).

345. A POSITA would have known, however, that each of these differences are known result-effective variables that control the frequency response of the filter and therefore how much pixel p1 is filtered to achieve a desired balance between reducing blocking artifacts without sacrificing overall image quality. Moreover, the

¹⁴ Rounding and truncation are the only two options when limiting the results of a division operation to a finite bit field. The rounding term used in the right shift form of the first offset equation would be dropped when expressed in division form.

exact high-pass filter component used, and the coefficient that component is assigned, are two variables that are subject to routine experimentation and optimization because others in the art had similarly combined these two components and used different high-pass filter components with different coefficients applied.

346. For example, during the JVT proceedings to develop the H.264 standard, the JVT proposed as part of the draft standard, B118, which discloses a three-tap filter, reproduced below:

$$\text{Clip}(-C0, C0, (p3 + P1 - p2 \ll 1) \gg 1)$$

B118, 48.

347. As discussed in Section VII.E, B118 uses a different naming convention for its pixel values than Fu and the '659 Patent (e.g., p1 is the pixel closest to the block boundary). Using Fu's naming convention for B118's second offset expression, it can be rewritten as:

$$\text{Clip}(-C0, C0, (p2 + P0 - p1 \ll 1) \gg 1)$$

348. If the clipping function of B118 is removed for simplicity, its second offset expression is:

$$(p2 + P0 - p1 \ll 1) \gg 1$$

349. Next, the mathematical equivalent of B118's second offset expression can be rewritten, (1) where P0 is substituted for its underlying equation ($p0 + \Delta$), (2) the left shift operation ($\ll$) is substituted for a mathematically equivalent

multiplication operation and (3) the right shift operation ($\gg$) is substituted for a mathematically equivalent division equation:

$$(1): (p2 + (p0 + \Delta) - p1 \ll 1) \gg 1$$

$$(2): (p2 - 2 \times p1 + p0 + \Delta) \gg 1$$

$$(3): (p2 - 2 \times p1 + p0)/2 + \Delta/2$$

350. A mathematical equivalent to the second offset expression of B118, therefore, discloses the same three tap filter equation disclosed in element 4[a]—namely a three-tap filter with coefficient values [1, -2, 1].

351. And although B118 taught scaling a Laplacian (in this case the [1, -2, 1] term) by dividing it by 2, those in the art would have understood that how much the Laplacian was scaled simply controlled how much additional high-frequency component was preserved as part of the offset. For instance, a POSITA would have been motivated to confirm through routine experimentation that, if it was desired to preserve Fu's level of filtering, the Laplacian should be divided by 4 (or $\frac{p2-2p1+p0}{4}$) to produce similar outputs to Fu's high-pass filter. Or, if more filtering was desired, B118's scaling by dividing by 2 could be selected. To demonstrate why, I've provided several example pixel configurations to show how Fu's high-pass filter and a scaled Laplacian would produce substantially similar outputs to modify p1, but scaling by 2 would produce more filtering:

p2	p1	p0	q0	q1	Type	Fu	Laplacian/4	Laplacian/2
200	150	150	150	150	Step (p1)	6	13	25
200	200	150	150	150	Step (p0)	-13	-13	-25
200	200	200	150	150	Step (q0)	6	0	0
150	160	170	180	190	Ramp (10)	0	0	0
125	150	175	200	225	Ramp (25)	0	0	0
200	250	200	250	200	Alternating	-25	-25	-50
255	50	50	255	255	Line	0	51	103
255	255	50	50	255	Line	-51	-51	-103
175	200	150	150	175	Random	-16	-19	-38
50	125	100	150	125	Random	-22	-25	-50
75	100	150	75	125	Random	19	6	13

352. While these high-pass offsets do not perform identically in every instance (particularly because the Laplacian does not use q0 as part of its calculation like Fu), these high-pass offsets all operate extremely similarly across various pixel conditions. And, in my opinion, the differences observed between them are small, where Laplacian/4 would be within a workable range that a POSITA would have experimented with to balance competing quality and efficiency objectives.

353. As another example, Bjontegaard discloses the following second offset equation to modify pixel p1 (discussed in more detail in Section X.B.4 below):

$$Diff = 5(b - 2c + d)/16$$

Bjontegaard, 20.

354. However, in my opinion and as discussed in more detail below in Section X.B.4, a POSITA would have understood that Bjontegaard’s equation would be read as: $Diff_{[1]} = 5(b - 2c + d_{\text{modified}})/16$, because Bjontegaard provides a logical walkthrough of the deblocking filter’s operation, in which each Section logically follows those that precede it. That is why Bjontegaard re-initializes the variable “Diff” to equal zero in Sections §§ 4.5.6.1 and 4.5.6.2, because Bjontegaard uses the same variable “Diff” (for efficiency) for multiple calculations. Without re-initializing “Diff” at the start of each section, the value of “Diff” previously stored in memory from calculations in a preceding section would be used.

355. The same reasoning therefore applies to the value of pixels d and e calculated in Section § 4.5.5. Although Bjontegaard labels the modified value of d as d_{modified} , logically it is just a redefined value of d (*i.e.*, $d = d + Diff$) because d_{modified} will replace d in the image.

356. Bjontegaard’s Code would have confirmed this understanding, because Bjontegaard discloses modifying pixel d *before* then filtering pixel c. *See* Ex-1014, 11. Specifically, Bjontegaard’s Code labels the first offset using variable “id” and pixel d using variable “li[3].” *Id.* The offset “id” is then applied to “li[3]” using equations: “ $il1 = \text{img} \rightarrow \text{li}[3] + \text{id}$ ” (to calculate the new pixel value) and “ $\text{img} \rightarrow \text{li}[3] =$

il1” (to apply the new pixel value to li[3] (pixel d)). Only then does filtering proceed to pixel c (li[2]) using the new value of pixel d (li[3]). *Id.* (identified by “/* center left */” section of code). In my opinion, therefore, Bjontegaard teaches using the *modified* value of d in the equation $\text{Diff}_{[1]} = 5(b - 2c + d_{\text{modified}})/16$.

357. A POSITA would have found it obvious to modify a second offset equation to use modified values of the pixels closer to the block boundary because, as was common in the art, filtering pixels further from the boundary should depend in some way on how much pixels closer to the boundary are filtered. *See, e.g.*, H.241, 29 (modifying p0 using Δ , and p1 using $\Delta/2$); B205, 131 (same); Ex-1039, 215 (applying $\Delta/2$, $\Delta/4$, $\Delta/8$ when moving further from boundary). Otherwise, if the offset applied to pixels further from the boundary did not depend on the offset applied to pixels closer to the boundary, that could result in new discontinuities along the boundary as a result of deblocking, which would be contrary to the intended purpose of deblocking to eliminate such discontinuities. By using d_{modified} instead of d, therefore, Bjontegaard would ensure the new pixel value for pixel c has some relationship to the offset applied to pixel d.

358. Applying this reasoning, Bjontegaard’s equation either was or would obviously be re-written as follows:

$$\text{Diff}_c = \frac{5(b - 2c + d_{\text{modified}})}{16}$$

$$\begin{aligned} \text{where } d_{\text{modified}} &= d + \text{Diff}_1: \\ &= \frac{5(b - 2c + d + \text{Diff}_1)}{16} \\ &= \frac{5(b - 2c + d)}{16} + \frac{5 * \text{Diff}_1}{16} \end{aligned}$$

And re-writing this equation with delta and p, consistent with the '659 Patent:

$$\Delta_{p1} = \frac{5(p2 - 2p1 + p0)}{16} + \frac{5\Delta_{p0}}{16}$$

359. The prior art also discloses a more simplistic version of a second offset equation, which simply took the result of the first offset equation, divided it by two, and added that result to the pixel value of p1 to arrive at a modified pixel p1':

$$p1_i = \text{Clip}_{0-255}(p1_i + \frac{\Delta}{2})$$

B205, 131.

360. Considering the second offset equations that were disclosed in the prior art, Fu's second offset equation takes the simple second offset equation from B205 and adds a 4-tap high pass filter.

361. "When using a block transform for residual coding, it is well known that the coding errors are larger near the block boundaries than in the middle of the block." List, 615. As the pixel being modified gets further away from the block boundary, therefore, it should be modified to a lesser degree because the coding error is not as large, and the pixel should not be overcorrected. As such, if the first offset

is calculated as Δ , which is used to modify the pixels closest to the block boundary (on either side of the block boundary), then a POSITA would have recognized that at least half of the offset value should be used when modifying the pixel next closest to the block boundary (and then $1/4$ for the pixel second closest to the block boundary; $1/8$ for the pixel third closest to the block boundary, etc.). See Ex-1039, Figure 4 (below, showing a first modified pixel value as $x/2$ and a second modified pixel value as $x/4$).

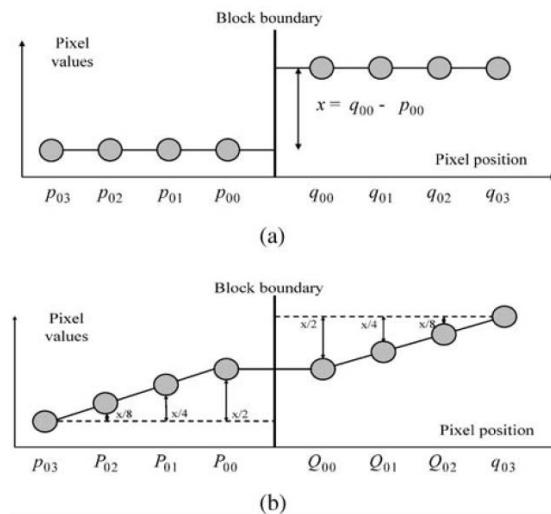


Figure 4. One dimensional view of the block boundary by offset-based filtering mode. a Before filtering. b After filtering.

Ex-1039, Fig. 4.

362. Moreover, adding half of the offset would ensure a gradual slope in the modification of pixels near the block boundary. Fu also used “ $\Delta/2$ ” to offset p_1 and q_1 , but Fu added something to the equation: a high-pass filter of $\frac{p_2 - 3p_1 + 3p_0 - q_0}{8}$ that kept a small amount of the high-frequency component.

363. Furthermore, B118 suggests —just like Fu—to combine a high-pass filter with a “ $\Delta/2$.” Specifically, its p1 offset equation combines a scaled Laplacian high-pass filter with an offset. Ex-1023, 48. B118 presents the following equation, using “p1” as a notation for the pixel closest to the block boundary, which I have converted to p0 notation for consistency:

$$\begin{aligned} P0 &= p0 + \Delta \\ P1 &= p1 + (p2 + \textcolor{red}{P0} - p1 \ll 1) \gg 1) \\ &= p1 + (p2 + \textcolor{red}{p0} + \Delta - p1 \ll 1) \gg 1) \\ &= p1 + \frac{p2 - 2p1 + p2 + \Delta}{2} \\ &= p1 + \frac{p2 - 2p1 + p2}{2} + \frac{\Delta}{2} \end{aligned}$$

364. In other words, those in the art knew that different high-pass filters could be combined with “ $\Delta/2$ ” as part of the p1 equation. Those in the art would have understood that doing so preserved some additional high-frequency components that “ $\Delta/2$ ” alone may have filtered out. Specifically, a POSITA would have been motivated to modify Fu by using a 3-tap high-pass filter instead of a 4-tap high-pass filter because doing so would reduce computational complexity. The [1, -2, 1] filter is a well-known high pass filter, and its usage in Fu would provide a POSITA with a reasonable expectation of success.

365. A POSITA would have been motivated to replace Fu's four-pixel (i.e., four-tap) high-pass filter component with three-pixel (i.e., three-tap) high-pass filter component found in the prior art because a three-tap equation is more computationally efficient than a four-tap equation. This replacement would have been the result of routine experimentation, as shown by the prior art equations above. For example, the three-tap equation utilized by B118 and Bjontegaard ($p_2 - 2p_1 + p_0$) was among the simplest and most efficient equations to characterize high frequencies. *See* Section VII.E (B118 overview); Section VII.F (Bjontegaard overview); Section X.B.4 (claim 4, Ground 2 (Bjontegaard)). B118 and Bjontegaard also demonstrate that replacing Fu's four-tap equation with a three-tap equation would not produce unexpected results in this same context.

366. A POSITA would have also been motivated to apply a coefficient of $\frac{1}{4}$ to the high-pass filter component of Fu for several reasons. First, the '659 Patent recognized that the specific equation used for the p_1 and q_1 pixels and the coefficient applied to this component are not critical, as other equations could be used if more filtering were desired. Ex-1001, 15:42-45 ("one can use a stronger low-pass filter in combination with the first offset. In this case, the second offset is calculated based on, such as equal to, $(p_2 + p_0 - 2xp_1 + \Delta)/2$," instead of $(p_2 + p_0 - 2xp_1 + 2\Delta)/4$, as claimed). In my opinion, this reinforces what would have been already apparent to a POSITA: that the coefficients applied to each component simply control how much

the pixel is ultimately modified, balancing the desired contributions of the high-pass and low-pass components. A POSITA would have recognized that the first second offset $((p_2 + p_0 - 2x_{p1} + \Delta)/2)$ is like B118.

367. Second, a POSITA would have known that when replacing Fu's four-tap filter of $(p_2 - 3p_1 + 3p_0 - q_0)$ with a simpler three-tap filter of $(p_2 - 2p_1 + p_0)$, the three-tap filter would need to have a coefficient of $\frac{1}{4}$ applied to produce similar offset values as Fu, which is consistent with B118 and Bjontegaard. A POSITA looking to modify Fu's second offset equation to make it more efficient, while maintaining approximately the same amount of filtering would have found the second offset equation recited in element 4[a] obvious.

368. As discussed in Section X.A.1.b, the second offset equation ultimately selected by Fu versus that of element 4[a] is the result of routine experimentation and optimization by modifying known result-effective variables—here the coefficients (i.e., weight applied to each pixel) and the number of pixels considered (i.e., what size filter to use).

369. In my opinion, a POSITA also would have had a reasonable expectation of success in designing a filter equation for the p_1 and q_1 pixels because the prior art was replete with examples, the design considerations were known, and it was largely understood to be an empirical exercise of testing different filter coefficients

to arrive at a desired outcome (e.g., overall image quality and deblocking performance, which depends on the amount of the offset).

- b. Element 4[b]: calculating a third offset to be based on $(q_2 + q_0 - 2 \times q_1 - 2 \times \Delta)/4$, wherein q_2 denotes a pixel value of a pixel second next closest to said block boundary in said corresponding line of pixels;**

370. In my opinion, Fu in view of a POSITA's knowledge discloses or suggests element 4[b].

371. Fu teaches that its third offset can be calculated to derive the value of pixel q_1 —the next closest pixel to the block boundary in the corresponding line of pixels q_0, q_1, q_2 , and q_3 .

372. Fu discloses that after p_0 and q_0 are filtered, a decision is made whether to filter q_1 . In the third case, “when a_q is greater than 0 at the luma block boundary, filter q_1” Fu, 8:18-20.

373. Fu teaches that for process 3, “the definition of a_p and a_q is the same as that of process 1 and process 2” and that a_q is obtained from the following formula:

$$a_q = \text{Abs}(q_2 - q_0) < \beta \&\& \text{Abs}(p_0 - q_0) < ((\alpha \gg 2) + 2)$$

Fu, 5:14, 8:21-22.

374. Fu teaches that it's the modified value of q_1 (Q_1) is derived from the following equation:

$$Q_1 = \text{Clip1}(q_1 - \text{Clip3}(-C, C, ((q_1 - q_0) \times 3 + (p_0 - q_2) + 4) \gg 3))$$

where the third offset equation is boxed in red above. Fu's third offset equation is subtracted from pixel value q1 to calculate Q1 (i.e., q1'). Fu, 8:18-20.

375. Removing Fu's Clip3 and Clip1 functions for simplicity, Fu's third offset equation is:

$$\left(((q1 - Q0) \times 3 + (P0 - q2) + 4) \gg 3 \right)$$

376. As shown in green above, Fu's third offset equation relies on modified pixel values P0 and Q0, pixel value q1, as well as pixel value q2, which is the second next closest pixel to the block boundary in the corresponding line of pixels q0, q1, q2, q3 shown in Fu Figure 2 below:

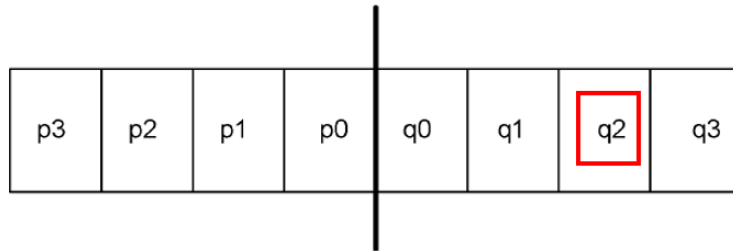


Figure 2

377. Fu includes the modified version of P0 and Q0 (i.e., p0' and q0' in terms of the '659 Patent) in its third offset equation, which can be replaced with their underlying equations $P0 = (Clip1(p0 + \text{delta}))$ and $Q0 = (Clip1(q0 - \text{delta}))$:

$$\frac{3(q1 - q0 + \text{delta}) - 1(q2 - p0 - \text{delta})}{8}$$

$$= \frac{(p0 - 3q0 + 3q1 - q2 + 4(delta))}{8}$$

$$= \frac{(p0 - 3q0 + 3q1 - q2)}{8} + \frac{2(delta)}{4}$$

378. As shown above, Fu combines a high-pass filter component $\frac{(p0-3q0+3q1-q2)}{8}$ with a first-offset component $\frac{2(delta)}{4}$ as does the third offset equation in element 4[b] of the '659 Patent.

379. A table comparing Fu's third offset expression (dropping the rounding values¹⁵) and the third offset expression recited in element 4[b] is reproduced below:

Reference	Third Offset Equation
'659 Patent	$\Delta = \frac{1(q0 - 2q1 + q2)}{4} - \frac{2\Delta}{4}$
Fu	$\Delta = \frac{1(p0 - 3q0 + 3q1 - q2)}{8} + \frac{2\Delta}{4}$

380. Although Fu discloses the equation $\Delta_{q1} = \frac{1(p0-3q0+3q1-q2)}{8} + \frac{1\Delta}{2}$, which *adds* the first-offset to Fu's high-pass component rather than subtract it, Fu, 8:18, Fu does so because the third offset is subsequently *subtracted* from pixel q1,

¹⁵ Rounding and truncation are the only two options when limiting the results of a division operation to a finite bit field. The rounding term used in the right shift form of the first offset equation would be dropped when expressed in division form.

id., 8:20 (“Q1=Clip1(q1−[Δ_{q1}])”). However, a POSITA would have understood that Fu’s equation can be re-written in its negative form as $\Delta_{q1} = \frac{1(q2-3q1+3q0-p0)}{8} - \frac{1\Delta}{2}$, which *subtracts* the first offset from the high-pass component (like element 4[b] does), as long as the third offset is then *added* to the value of q1 rather than subtracted (e.g., Q1=Clip1(q1+[Δ_{q1}])” (like element 4[d] does). The negative form of Fu’s third-offset equation aligns with how B118 and Bjontegaard (as well as the ’659 Patent, elements 4[b] and [d]) treat the third offset, making it the more appropriate form for comparison between these equations:

	Third Offset	Application of Third Offset
Fu (Original)	$\frac{1(p0 - 3q0 + 3q1 - q2)}{8} + \frac{1\Delta}{2}$	Q1=q1−[offset]
Fu (Negative)	$\frac{1(q2 - 3q1 + 3q0 - p0)}{8} - \frac{1\Delta}{2}$	Q1=q1+[offset]
B118	$\frac{1(q2 - 2q1 + q0)}{2} - \frac{1\Delta}{2}$	Q1=q1+[offset]
Bjontegaard	$\frac{5(q2 - 2q1 + q0)}{16} - \frac{5\Delta}{16}$	Q1=q1+[offset]
’659 Patent	$\frac{1(q2 - 2q1 + q0)}{4} - \frac{1\Delta}{2}$	Q1 = q1+[offset]

And mathematically, Fu’s original and Fu’s negative third-offset equations produce the same modified value of q1. A comparison of these two offset equations is provided below, in which I take the example signals from above, extend the

examples to include q2 pixel values, and use Fu's first-offset values for each example signal calculated in element 1[a]:

p2	p1	p0	q0	q1	q2	Type	$\frac{1(p0 - 3q0 + 3q1 - q2)}{8} + \frac{1\Delta}{2}$	$\frac{1(q2 - 3q1 + 3q0 - p0)}{8} - \frac{1\Delta}{2}$
200	150	150	150	150	150	Step (p1)	0	0
200	200	150	150	150	150	Step (p0)	3	-3
200	200	200	150	150	150	Step (q0)	0	0
150	160	170	180	190	200	Ramp (10)	0	0
125	150	175	200	225	250	Ramp (25)	0	0
200	250	200	250	200	250	Alternating	-13	13
255	50	50	255	255	255	Line	0	0
255	255	50	50	255	255	Line	51	-51
175	200	150	150	175	200	Random	5	-5
50	125	100	150	125	100	Random	0	0
75	100	150	75	125	75	Random	13	-13

Subtracting Fu's third offset calculated by $\frac{1(p0 - 3q0 + 3q1 - q2)}{8} + \frac{1\Delta}{2}$ would therefore produce the same modified pixel value for pixel q1 as adding the third offset calculated by $\frac{1(q2 - 3q1 + 3q0 - p0)}{8} - \frac{1\Delta}{2}$.

381. As shown above, Fu's third offset expression differs from the third offset expression recited in element 4[a] in two ways: (1) the exact high-pass filter

used (in blue, above) and (2) the coefficient applied to that component (in red, above).

382. For similar reasons discussed above in Section X.A.4.a, a POSITA would have found it obvious to use (1) a three-tap, high-pass filter component $q_2 - 2q_1 + q_0$ like B118 and Bjontegaard and (2) to apply a weight of $\frac{1}{4}$ to this component because they were known result effective variables and because they would have produced a more-efficient third offset equation that provided similar filtering properties as F_u .

383. A POSITA also would have had a reasonable expectation of success in designing a filter equation for the p_1 and q_1 pixels because the prior art was replete with examples, the design considerations were known, and it was largely understood to be an empirical exercise of testing different filter coefficients to arrive at a desired outcome (e.g., overall image quality and deblocking performance, which depends on the amount of the offset).

- c. **Element 4[c]: modifying said pixel value p1 of said pixel next closest to said block boundary in said line of pixels by adding said second offset to said pixel value p1 of said pixel next closest to said block boundary in said line of pixels to form a modified pixel value p1' of said pixel next closest to said block boundary in said line of pixels; and**

384. In my opinion, Fu in view of a POSITA's knowledge discloses or suggests element 4[c].

385. Fu discloses that the modified value of pixel p1, which as shown in Figure 2 of Fu below (boxed in red) is the pixel next closest to the block boundary in the line of pixels p0, p1, p2, p3, is derived by adding the result of the second offset to the value of pixel p1.

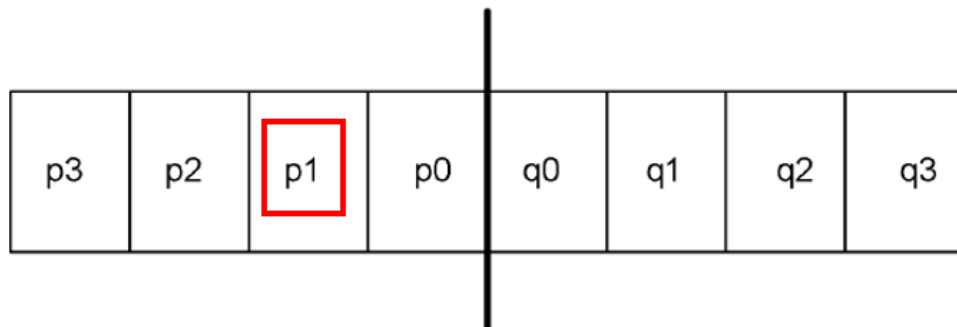


Figure 2

Fu, 8:15-17.

386. Fu's equation to calculate the modified value of p1 (i.e., P1) is reproduced below:

$$P1 = \text{Clip1}(p1 + \text{Clip3}(-C, C, ((P0 - p1) \times 3 + (p2 - Q0) + 4) \gg 3))$$

387. Fu discloses that its second offset equation (boxed in red above and discussed above in element 4[a]) can be clipped by Fu's Clip3 function. Applying the logic of Fu's Clip3 function, the following conditions are recognized:

- If the result of Fu's second offset equation is less than -C, then the result of the clipping function is -C;
- If the result of Fu's second offset equation is greater than C, then the result of the clipping function is C;
- If the result of Fu's second offset equation is within the range of -C and C, then the result is Fu's second offset equation.

388. Fu also discloses that the result of $p1 +$ the second offset equation is clipped by Fu's Clip1 function. Fu's Clip1 function is expressed as $\text{Clip1}(x) = \text{Clip3}(0, 255, x)$. Fu, 8, 10-11. Applying the logic of Fu's Clip3 function, the following conditions are recognized:

- If the result of $p1 +$ second offset is less than 0, then P1 (i.e., the modified value of $p1$) is 0;
- If the result of $p1 +$ second offset is greater than 255, then P1 is 255;
- If the result of $p1 +$ second offset is within the range of 0-255, then P1 is $p1 +$ second offset.

389. If the third conditions are met for both of Fu's clipping equations, then P1 is the sum of the result of the second offset equation and pixel value p1.

- d. Element 4[d]: modifying said pixel value q1 of said pixel next closest to said block boundary in said corresponding line of pixels by adding said third offset to said pixel value q1 of said pixel next closest to said block boundary in said corresponding line of pixels to form a modified pixel value q1' of said pixel next closest to said block boundary in said corresponding line of pixels.**

390. In my opinion, Fu in view of a POSITA's knowledge discloses or suggests element 4[d].

391. Fu discloses that the modified value of pixel q1, which as shown in Figure 2 of Fu below (boxed in red) is the pixel next closest to the block boundary in the corresponding line of pixels q0, q1, q2, q3, is derived by subtracting the result of the third offset to the value of pixel q1. Although Fu subtracted the third offset, this is the mathematical equivalent of adding its negative, which simply depends on how the equation is written but does not change the result. *See, e.g.*, Section X.A.4.b

(element 4[b], noting equivalence of $Q1 = q1 - \frac{p0-3q0+3q1-q2}{8} + \frac{\Delta}{2} = q1 + \frac{q2-3q1+3q0-p0}{8} - \frac{\Delta}{2}$).

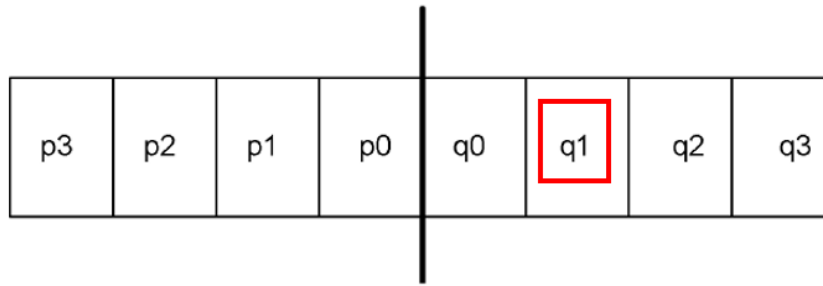


Figure 2

Fu, 8:18-20.

392. Fu's equation to calculate the modified value of q1 (i.e., Q1) is reproduced below:

$$Q1 = \text{Clip1}(q1 - \text{Clip3}(-C, C, ((q1 - q0) \times 3 + (p0 - q2) + 4) \gg 3))$$

393. Fu discloses that its third offset equation (boxed in red above and discussed above in element 4[b]) can be clipped by Fu's Clip3 function. Applying the logic of Fu's Clip3 function, the following conditions are recognized:

- If the result of Fu's third offset equation is less than -C, then the result of the clipping function is -C;
- If the result of Fu's third offset equation is greater than C, then the result of the clipping function is C;
- If the result of Fu's third offset equation is within the range of -C and C, then the result is Fu's third offset equation.

394. Fu also discloses that the result of $q1 - \text{third offset}$ equation is clipped by Fu's Clip1 function. Fu's Clip1 function is expressed as $\text{Clip1}(x) = \text{Clip3}(0, 255, x)$. Fu, 8, 10-11. Applying the logic of Fu's Clip3 function, the following conditions are recognized:

- If the result of $q1 - \text{third offset}$ is less than 0, then Q1 (i.e., the modified value of $q1$) is 0;
- If the result of $q1 - \text{third offset}$ is greater than 255, then Q1 is 255;
- If the result of $q1 - \text{third offset}$ is within the range of 0-255, then Q1 is $q1 - \text{third offset}$.

395. If the third conditions are met for both of Fu's clipping equations, then Q1 is the difference (or sum of the negative form of Fu's third offset equation, *see* Section X.A.4.b (element 4[b]) of the pixel value $q1$ and the third offset equation.

5. Claim 5

396. As discussed above in Section X.A.4, Fu in view of the knowledge of a POSITA renders claim 4 obvious.

- a. **Element 5[a]: The method according to claim 4, wherein calculating said second offset comprises calculating said second offset to be equal to $((p_2 + p_0 + 1) \gg 1) - p_1 + \Delta \gg 1$, wherein $\gg$ denotes a right shift operation; and**

397. In my opinion, Fu in view of a POSITA's knowledge discloses or suggests element 5[a].

398. The '659 Patent discloses that the second offset can be calculated using a traditional division form, or can alternatively be calculated in a "hardware-adapted implementation" using a right-shift function instead of division. Ex-1001, 15:41-51.

399. As I discuss above in Section X.A.4.a, in my opinion Fu renders the second-offset equation from element 4[a] obvious. The second-offset equation recited here uses a right-shift function instead of division but that is the obvious substitute for a hardware implementation. The second-offset from element 4[a] is reproduced below as a starting point:

$$= [(p_2 + p_0 - 2 \times p_1 + 2 \times \Delta)]/4$$

400. The mathematical equivalent of the second-offset expression from element 4[a] is reproduced below, where the entire expression is divided by two:

$$= \left[\left(\frac{1}{2} p_2 + \frac{1}{2} p_0 - p_1 + \Delta \right) \right] / 2$$

401. If the like $\frac{1}{2}$ terms, p_2 and p_0 , are combined, then that results in the mathematically equivalent expression:

$$= \left[\left(\frac{1}{2} (p_2 + p_0) - p_1 + \Delta \right) \right] / 2$$

402. From here, a POSITA would have recognized that a right-shift function can be substituted for the division functions in the expression above. That would result in the following:

$$= \left(((p_2 + p_0) \gg 1) - p_1 + \Delta \right) \gg 1$$

403. As discussed in Section V.B.2.b above, however, the output of a right-shift function is always an integer. As such, the right-shift function truncates the value of the result obtained after the division function to the greatest integer whose value is less than or equal to that result, and thus introduces some inaccuracies to the result. It is well known to a POSITA that those inaccuracies can be reduced by adding an offset to the value prior to the right-shift function whose value is equal to half of the value of the denominator. If you consider a denominator of 4 (i.e., a right-shift of 2), then the rounding value that is added to offset is 2. This approach results in an approximation of the result obtained by the division function value by producing the integer whose value is closest to that result.

404. Here, the $p_2 + p_0$ term is right shifted by 1 bit, which is the hardware alternative to dividing by 2 (i.e., 2^1). To compensate for the inaccuracies introduced

by this operation, an offset value of 2/2 (or 1) is added to the $p_2 + p_0$ term prior to the right-shift function, which would result in:

$$= ((p_2 + p_0 + 1) \gg 1) - p_1 + \Delta \gg 1$$

405. As explained above, the output of a right-shift operation is always rounded down to the integer value that is less than or equal to the value obtained by the division function. To round the output to the nearest integer, an offset value is added prior to the right-shift function, which has a value that is half of the value of the denominator. A POSITA would have understood that whether or not to include an offset value is a design choice of whether the result of the right shift is rounded to the nearest integer value or truncated to the greatest integer whose value is less than or equal to that result. Here, to round the output of $(p_2 + p_0) \gg 1$ to the nearest integer, an offset value of 1 is added prior to the right-shift function.

406. Applying the hardware alternative to the second-offset expression of element 4[a] arrives at the second-offset expression recited in element 5[a]. Fu renders element 5[a] obvious for the same reasons discussed above in Section X.A.4.a.

b. Element 5[b]: calculating said third offset comprises calculating said third offset to be equal to $((q_2 + q_0 + 1) \gg 1) - q_1 - \Delta \gg 1$

407. In my opinion, Fu in view of a POSITA's knowledge discloses or suggests element 5[b].

408. As I discuss above in Section X.A.4.b, in my opinion Fu renders the third offset expression from element 4[b] obvious. The third-offset expression recited here uses a right-shift function instead of division but that is the obvious substitute for a hardware implementation. The third-offset from element 4[b] is reproduced below as a starting point:

$$= [(q2 + q0 - 2 \times q1 - 2 \times \Delta)]/4$$

409. The mathematical equivalent of the third-offset expression from element 4[b] is reproduced below, where the entire expression is divided by two:

$$= \left[\left(\frac{1}{2} q2 + \frac{1}{2} q0 - q1 - \Delta \right) \right] / 2$$

410. If the like $\frac{1}{2}$ terms, $q2$ and $q0$, are combined, then that results in the mathematically equivalent expression:

$$= \left[\left(\frac{1}{2} (q2 + q0) - q1 - \Delta \right) \right] / 2$$

411. From here, a POSITA would have recognized that a right-shift function can be substituted for the division functions in the expression above. That would result in the following:

$$= \left(((q2 + q0) \gg 1) - q1 - \Delta \right) \gg 1$$

412. As discussed in Section V.B.2.b above, however, the output of a right-shift function is always an integer. As such, the right-shift function truncates the

value of the result obtained after the division function to the greatest integer whose value is less than or equal to that result, and thus introduces some inaccuracies to the result. It is well known to a POSITA that those inaccuracies can be reduced by adding an offset to the value prior to the right-shift function whose value is equal to half of the value of the denominator. If you consider a denominator of 4 (i.e., a right-shift of 2), then the rounding value that is added to offset is 2. This approach results in an approximation of the result obtained by the division function value by producing the integer whose value is closest to that result.

413. Here, the $q_2 + q_0$ term is right shifted by 1 bit, which is the hardware alternative to dividing by 2 (i.e., 2^1). To compensate for the inaccuracies introduced by this operation, an offset value of $2/2$ (or 1) is added to the $q_2 + q_0$ term prior to the right-shift function, which would result in:

$$= ((q_2 + q_0 + 1) \gg 1) - q_1 - \Delta \gg 1$$

414. As explained above, the output of a right-shift operation is always rounded down to the integer value that is less than or equal to the value obtained by the division function. To round the output to the nearest integer, an offset value is added prior to the right-shift function, which has a value that is half of the value of the denominator. A POSITA would have understood that whether or not to include an offset value is a design choice of whether the result of the right shift is rounded to the nearest integer value or truncated to the greatest integer whose value is less

than or equal to that result. Here, to round the output of $(q_2 + q_0) \gg 1$ to the nearest integer, an offset value of 1 is added prior to the right-shift function.

415. Applying the hardware alternative to the third-offset equation of element 4[b] arrives at the third-offset equation recited in element 5[b]. Fu renders element 5[b] obvious for the same reasons discussed above in Section X.A.4.b.

6. Claim 6

416. As discussed above in Section X.A.4, Fu in view of the knowledge of a POSITA renders claim 4 obvious.

- a. **Element 6[a]: The method according to claim 4, further comprising: clipping said second offset to be within an interval of $-t_{C2}$ and t_{C2} by setting said second offset to be equal to $-t_{C2}$ if said second offset is smaller than $-t_{C2}$ and by setting said second offset to be equal to t_{C2} if said second offset is larger than t_{C2} ,**

417. In my opinion, Fu discloses or suggests element 6[a].

418. Fu teaches that its second offset is clipped by Fu's Clip3 function, as shown below:

$$P1 = Clip1(p1 + Clip3(-C, C, ((P0 - p1) \times 3 + (p2 - Q0) + 4) \gg 3))$$

419. Fu's Clip3(a,b,c) function is defined by the following conditions:

- If c is less than a, then the resulting value is a
- If c is greater than b, then the resulting value is b

- If c is any other value (i.e., c is between the bounds of a and b), then the resulting value is c

420. Turning again to Fu's expression to derive the second offset (reproduced below): the "a" value of Fu's Clip3 function is $-C$; the "b" value is C ; and the "c" value is the second offset expression (boxed in purple).

$$\text{Clip3}(-C, C, ((P0 - p1) \times 3 + (p2 - Q0) + 4) \gg 3)$$

Fu, 8:5-11.

421. Applying the logic of Fu's Clip3 function, if the result of Fu's second offset expression is less than $-C$ (or $-t_{c2}$, in terms of element $6[a]$), then the result of the second offset expression is the set to $-C$. Conversely, if the result of Fu's second offset expression is greater than C (or t_{c2} in terms of element $6[a]$), then the result of the second offset expression is set to C .

- b. Element 6[b]: wherein $-t_{c2}$ is a threshold value that is dependent on a quantization parameter value assigned to said block; and**

422. In my opinion, Fu discloses or suggests element 6[b].

423. Fu discloses that its second offset is clipped between the threshold values $-C$ and C , as discussed in Section X.A.6.a. As discussed in Section X.A.2.b, the threshold values $-C$ and C of Fu's Clip3 function depend on a quantization parameter QP , which is assigned to the p and q blocks.

- c. **Element 6[c]: clipping said third offset within an interval of $-t_{C2}$ and t_{C2} by setting said third offset to be equal to $-t_{C2}$ if said third offset is smaller than $-t_{C2}$ and by setting said third offset to be equal to t_{C2} if said third offset is larger than t_{C2} .**

424. In my opinion, Fu discloses or suggests element 6[c].

425. Fu teaches that its third offset is clipped by Fu's Clip3 function, as shown below:

$$Q1 = Clip1(q1 - Clip3(-C, C, ((q1 - Q0) \times 3 + (P0 - q2) + 4) \gg 3))$$

426. Fu's Clip3(a,b,c) function is defined by the following conditions:

- If c is less than a, then the resulting value is a
- If c is greater than b, then the resulting value is b
- If c is any other value (i.e., c is between the bounds of a and b), then the resulting value is c

427. Turning again to Fu's expression to derive the third offset (reproduced below): the "a" value of Fu's Clip3 function is -C; the "b" value is C; and the "c" value is the third offset expression (boxed in purple).

$$Clip3(-C, C, ((q1 - Q0) \times 3 + (P0 - q2) + 4) \gg 3)$$

Fu, 8:5-11.

428. Applying the logic of Fu's Clip3 function, if the result of Fu's third offset expression is less than -C (or $-t_{C2}$, in terms of element 6[c]), then the result of the third offset expression is the set to -C. Conversely, if the result of Fu's third

offset expression is greater than C (or t_{c2} in terms of element 6[c]), then the result of the third offset expression is set to C .

7. Claim 7

429. As discussed above in Section X.A.4, Fu in view of the knowledge of a POSITA renders claim 4 obvious.

- a. **Element 7[a]: The method according to claim 4, further comprising: clipping said modified pixel value $p1'$ of said pixel next closest to said block boundary in said line of pixels to be within an interval of zero and a defined maximum value by setting said modified pixel value $p1'$ of said pixel next closest to said block boundary in said line of pixels to be equal to zero if said modified pixel value $p1'$ of said pixel next closest to said block boundary in said line of pixels is smaller than zero and by setting said modified pixel value $p1'$ of said pixel next closest to said block boundary in said line of pixels to be equal to said defined maximum value if said modified pixel value $p1'$ of said pixel next closest to said block boundary in said line of pixels is larger than said defined maximum value; and**

430. In my opinion, Fu discloses or suggests element 7[a].

431. Fu teaches that its modified pixel value $P1$ (i.e., $p1'$ in terms of element 7[a]) is clipped. As shown in Fu's Figure 2, $P1$ is the modified value of $p1$, which is the pixel next closest to the block boundary in the line of pixels $p0, p1, p2, p3$.

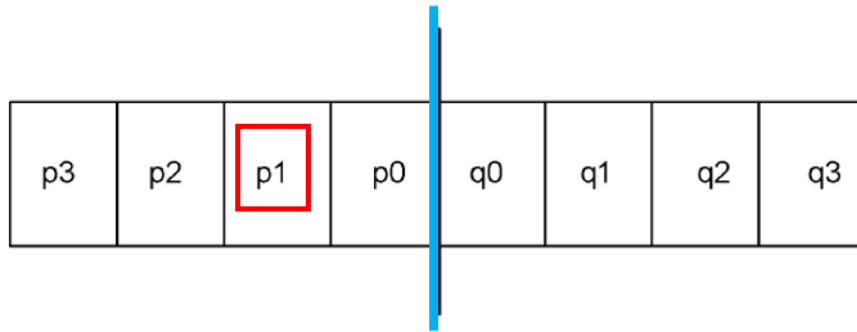


Figure 2

432. Fu teaches that P1 is calculated using Fu's Clip1 function, shown below, after the clipped second offset is added to the value of pixel p1:

$$P1 = \text{Clip1}(p1 + \text{Clip3}(-C, C, ((p0 - p1) \times 3 + (p2 - q0) + 4) \gg 3))$$

Where Fu teaches that Clip1(a) is expressed as $\text{Clip1}(x) = \text{Clip3}(0, 255, x)$. Fu, 8:15-17.

433. Fu's Clip1 function is expressed as $\text{Clip1}(x) = \text{Clip3}(0, 255, x)$. Fu, 8:10-11. Applying the logic of Fu's Clip3 function, the following conditions are recognized:

- If the result of $p1 + \text{second offset}$ is less than 0; then P1 (i.e., the modified value of p1) is 0;
- If the result of $p1 + \text{second offset}$ is greater than 255, then P1 is 255;
- If the result of $p1 + \text{second offset}$ is within the range of 0-255, then P1 is $p1 + \text{second offset}$.

434. Here, as discussed in Section V.B.1, 255 is defined by the maximum value that an 8-bit pixel can be (2^8-1). Fu, therefore, discloses to a POSITA that 255 is a defined maximum value.

435. For example, B205 teaches that 255 is a common defined maximum value of a clip function for pixels. Ex-1001, 2:58-3:3; Section V.B.1 (discussing pixel values). Similarly, the '659 Patent discloses that for its clipping function: “The parameter M denotes a defined maximum value that the modified pixel values can assume. In a particular embodiment, this maximum value is defined based on the number of bits that are spent on the pixel values. Thus, assume that the pixel values are in the form of a m-bit value, then M is preferably equal to 2^m-1 . For instance, if each pixel value is a 8-bit value, i.e., $m=8$, then $M=255$.” Ex-1001, 11:64-12:17.

- b. **Element 7[b]: clipping said modified pixel value $q1'$ of said pixel next closest to said block boundary in said corresponding line of pixels to be within an interval of zero and said defined maximum value by setting said modified pixel value $q1'$ of said pixel next closest to said block boundary in said corresponding line of pixels to be equal to zero if said modified pixel value $q1'$ of said pixel next closest to said block boundary in said corresponding line of pixels is smaller than zero and by setting said modified pixel value $q1'$ of said pixel next closest to said block boundary in said corresponding line of pixels to be equal to said defined maximum if said modified pixel value $q1'$ of said pixel next closest to said block boundary in said corresponding line of pixels is larger than said defined maximum value.**

436. In my opinion, Fu discloses or suggests element 7[b].

437. Fu teaches that its modified pixel value $Q1$ (i.e., $q1'$ in terms of element 7[b]) is clipped. As shown in Fu's Figure 2, $Q1$ is the modified value of $q1$, which is the pixel next closest to the block boundary in the corresponding line of pixels $q0$, $q1$, $q2$, $q3$.

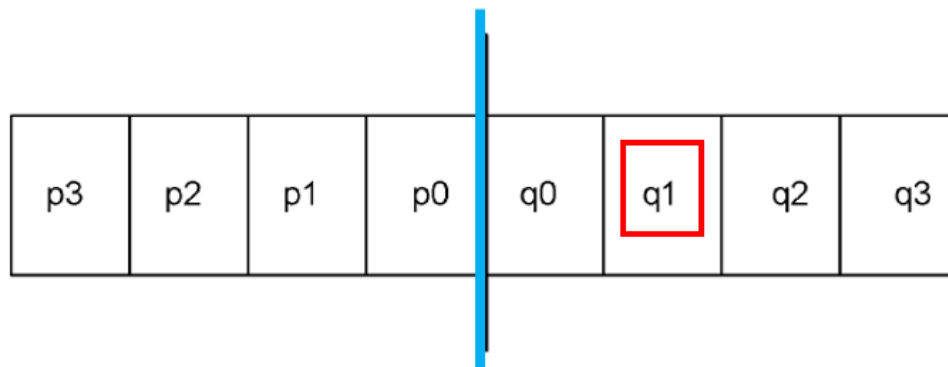


Figure 2

438. Fu teaches that Q1 is calculated using Fu's Clip1 function, shown below, after the clipped third offset is subtracted from the value of pixel q1:

$$Q1 = \text{Clip1}(q1 - \text{Clip3}(-C, C, ((q1 - Q0) \times 3 + (P0 - q2) + 4) \gg 3))$$

Where Fu teaches that Clip1(a) is expressed as $\text{Clip1}(x) = \text{Clip3}(0, 255, x)$. Fu, 8:18-20.

439. Fu's Clip1 function is expressed as $\text{Clip1}(x) = \text{Clip3}(0, 255, x)$. Fu, 8:10-11. Applying the logic of Fu's Clip3 function, the following conditions are recognized:

- If the result of $q1 - \text{third offset}$ is less than 0; then Q1 (i.e., the modified value of Q1) is 0;
- If the result of $q1 - \text{third offset}$ is greater than 255, then Q1 is 255;
- If the result of $q1 - \text{third offset}$ is within the range of 0-255, then Q1 is $q1 - \text{third offset}$.

440. Here, as discussed in Section V.B.1, 255 is defined by the maximum value that an 8-bit pixel can be (2^8-1). Fu, therefore, discloses to a POSITA that 255 is a defined maximum value.

441. For example, B205 teaches that 255 is a common defined maximum value of a clip function for pixels. Ex-1001, 2:58-3:3; Section V.B.1 (discussing pixel values). Similarly, the '659 Patent discloses that for its clipping function: "The

parameter M denotes a defined maximum value that the modified pixel values can assume. In a particular embodiment, this maximum value is defined based on the number of bits that are spent on the pixel values. Thus, assume that the pixel values are in the form of a m-bit value, then M is preferably equal to 2^m-1 . For instance, if each pixel value is a 8-bit value, i.e., $m=8$, then $M=255$.” Ex-1001, 11:64-12:17.

8. Claim 10

- a. **Element 10[pre]: A user equipment comprising a deblocking filter unit for reducing blocking artifacts at a block boundary between a block of multiple pixels and a neighboring block of multiple pixels in a video frame, wherein the deblocking filter unit comprises:**

442. In my opinion, Fu in view of a POSITA’s knowledge discloses or suggests element 10[pre] under Patent Owner’s proposed construction at the ITC—that the entire preamble is limiting. It follows, that 10[pre] would also be obvious under Lenovo’s construction, which agrees that the preamble is limiting with the exception of “user equipment comprising.”

443. In addition to the method for calculating first, second, and third offset equations discussed above, Fu discloses “a device for reducing block distortion that is capable of improving the accuracy of the acquired block boundary filtering strength, thereby effectively reducing block distortion.” Fu, 10:21-25; *see also id.*, 11:28-38 (“A device for reducing block distortion, comprising a block boundary filtering strength acquisition module, used to determine a similarity threshold and

calculate region flatness by using the similarity threshold, and acquire block boundary filtering strength with the region flatness; a boundary determination module, used to determine whether the block boundary is a pseudo-boundary; a filter determination module, used to determine the corresponding filter combination based on the pre-established correspondence relationship between block boundary filtering strength and filter combination according to the block boundary filtering strength acquired by the block boundary filtering strength acquisition module when the determination results of the boundary determination module indicate that the block boundary is a pseudo-boundary.”); 12:3-10 (“It can be seen from the above technical solution that the method and device for acquiring block boundary filtering strength and reducing block distortion provided by the examples of the present disclosure determine a similarity threshold based on block boundary thresholds pre-obtained from table lookup, or determine a similarity threshold based on quantization parameters, macroblock modes, color spaces or frame types, and adjust the initial value of block boundary filtering strength by using header information of the block. Hence, they are capable of being unrestricted to the coding block unit and make full use of the relationship between the blocks on both sides and the header information, thereby improving the accuracy of the acquired block boundary filtering strength, thereby effectively reducing block distortion.”).

444. The device described in Fu is used to reduce blocking artifacts, referred to as blocking distortion in Fu, between neighboring blocks of pixels.

		N0	R0	L0	M0		
		N1	R1	L1	M1		
		N2	R2	L2	M2		
		N3	R3	L3	M3		
		N4	R4	L4	M4		
		N5	R5	L5	M5		
		N6	R6	L6	M6		
		N7	R7	L7	M7		

Figure 1

445. Fu’s reference to reducing blocking distortion is known to be a reference to reducing blocking artifacts—or an artifact that remains after pixel blocks are processed. *E.g.*, List, 614 (“It can be seen that the main coding artifact in the loop-filter case is the blockiness on the 4 x 4 grid caused by prediction error coding. This artifact can be efficiently compensated by the deblocking filter described in this paper.”).

446. The device disclosed in Fu employs Fu’s method of reducing blocking artifacts, as discussed above in Section X.A.1.

447. Fu discloses an example schematic diagram of the device it teaches to reduce blocking distortion, which is reproduced below (annotated):

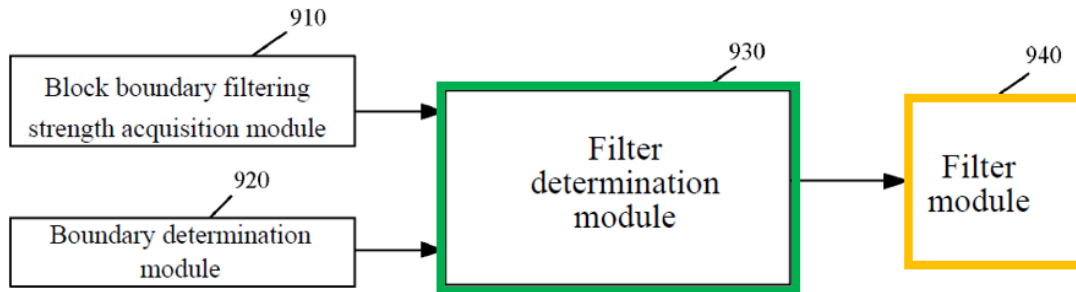
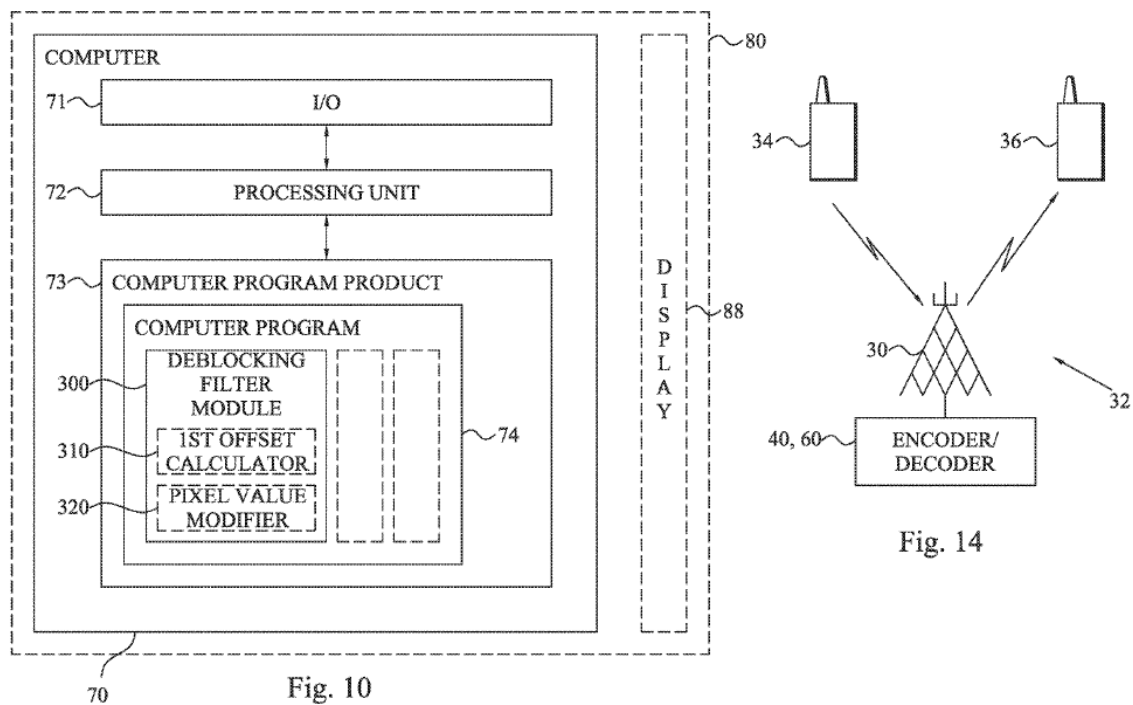


Figure 9

448. Fu’s “filter determination module 930 [green, above] determines the corresponding filter combination based on the pre-established correspondence relationship between block boundary filtering strength and filter combination according to the block boundary filtering strength acquired by the block boundary filtering strength acquisition module 910 when the determination results of the boundary determination module indicate that the block boundary is a pseudo-boundary.” Fu, 19:6-10.

449. Fu’s “filter module 940 [yellow, above] filters the pixels of the blocks on both sides of the block boundary in the windows by using the filter combination determined by the filter determination module 930.” Fu, 19:11-12. Fu’s offset calculators, pixel value modifier, and clipping unit, as recited in claim 10 and its dependents would reside in Fu’s filter module 940, which filters the blocks of pixels on both sides of the block boundary.

450. The '659 Patent describes that its disclosed offset calculators, clipping units, etc. “all may be special purpose circuits, such as ASICs (Application Specific Integrated Circuits), [but] alternative embodiments of the device 100, 200 are possible where some or all of the units 110, 120, and 210-290 are implemented as computer program modules running on a general purpose of processor,” such as shown in figure 10 reproduced below:



Ex-1001, Fig. 10, 20:50-59.

451. Fu teaches that “all or part of the steps of the method [discussed above] according to the above examples may be completed through a program instructing the relevant hardware, and the program may be stored in a type of computer-readable

storage medium. When the program is executed, it comprises one or a combination of the steps of the method.” Fu, 19:13-16.

452. Fu further teaches that “each functional unit in the various examples of [its] present disclosure may be integrated in a processing module, where each unit may exist physically alone, or two or more units may be integrated in a module. The above integrated modules may be implemented in the form of hardware or software functional modules. If the integrated module is implemented in the form of a software functional module and sold or used as a stand-alone product, it may be stored in a computer-readable storage medium” which may be “a read-only memory, magnetic disk, optical disc, or the like.” Fu, 19:18-23.

453. While Fu discloses that its module can be sold as a stand-alone product, a POSITA would also have recognized that Fu’s module can be implemented in the form of a “software functional module” that is incorporated into a user device like a cellular device. *Id.* It would be well within the skill of a POSITA to implement Fu’s software module in a media player of a computing device like shown in Figure 13 of the ’659 Patent. *See, e.g.*, Ex-1040, 1-5 (describing importance of video codecs, and particularly H.264, to various consumer electronics); Ex-1061, 9:49-57 (describing “[a]n electronic device for deblocking and edge enhancement”), 10:9-13 (“Those of ordinary skill would further appreciate that the various illustrative logical blocks, modules, and algorithm steps ... may be implemented as electronic hardware,

computer software, or combinations of both.”); Ex-1062, 18:17-18 (“a deblocking filter is utilized for loop filtering”), 22:23-24 (providing “a computer system with which some embodiments of the invention is implemented”). A POSITA would have been motivated to incorporate Fu’s software functional module into a user device because it would consolidate the number of components required for a media player solution on the user device, allowing for a simpler and all-inclusive solution versus one that relies on numerous stand-alone components.

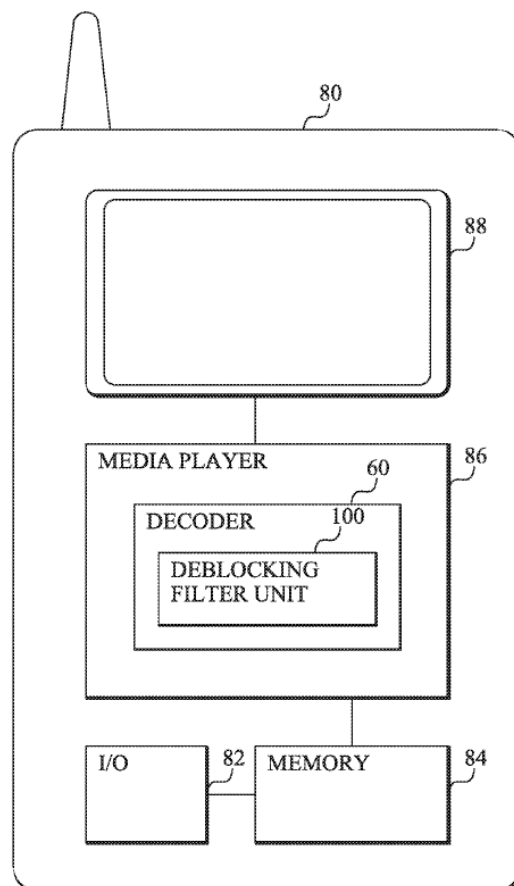


Fig. 13

Ex-1001, Fig. 13.

- b. Element 10[a]: a first offset calculator configured to calculate a first offset to be equal to $(9x(q_0 - p_0) - 3x(q_1 - p_1) + 8) \gg 4$, wherein $\gg$ denotes a right shift operation, p_0 denotes a pixel value of a pixel closest to said block boundary in a line of pixels in said block, wherein said line of pixels is perpendicular to said block boundary, p_1 denotes a pixel value of a pixel next closest to said block boundary in said line of pixels, q_0 denotes a pixel value of a pixel closest to said block boundary in a corresponding line of pixels in said neighboring block, wherein said corresponding line of pixels is perpendicular to said block boundary, and q_1 denotes a pixel value of a pixel next closest to said block boundary in said corresponding line of pixels; and**

454. In my opinion, Fu in view of a POSITA's knowledge discloses or suggests element 10[a] for the same reasons as I discuss above in Section X.A.1.b (element 1[a]). Element 10[a] is the apparatus equivalent of element 1[a].

- c. Element 10[b]: a pixel value modifier configured to modify said pixel value p_0 of said pixel closest to said block boundary in said line of pixels by adding said first offset to said pixel value p_0 of said pixel closest to said block boundary in said line of pixels to form a modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels**

455. In my opinion, Fu in view of a POSITA's knowledge discloses or suggests element 10[b] for the same reasons as I discuss above in Section X.A.1.c (element 1[b]). Element 10[b] is the apparatus equivalent of element 1[b].

- d. Element 10[c]: and to modify said pixel value q_0 of said pixel closest to said block boundary in said corresponding line of pixels to form a modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels.**

456. In my opinion, Fu in view of a POSITA's knowledge discloses or suggests element 10[c] for the same reasons as I discuss above in Section X.A.1.d (element 1[c]). Element 10[c] is the apparatus equivalent of element 1[c].

9. Claim 11

457. As discussed above in Section X.A.8, Fu in view of the knowledge of a POSITA renders claim 10 obvious.

- a. Element 11[a]: The user equipment according to claim 10, wherein the deblocking filter unit further comprises a first clipping unit configured to clip said first offset within an interval of $-t_c$ and t_c by setting a first offset to be equal to $-t_c$ if said first offset is smaller than $-t_c$ and by setting a first offset to be equal to t_c if said first offset is larger than t_c ,**

458. In my opinion, Fu discloses or suggests element 11[a] for the same reasons as I discuss above in Section X.A.2.a (element 2[a]). Element 11[a] is the apparatus equivalent of element 2[a].

- b. Element 11[b]: wherein t_c is a threshold value that is dependent on a quantization parameter value assigned to said block.**

459. In my opinion, Fu discloses or suggests element 11[b] for the same reasons as I discuss above in Section X.A.2.b (element 2[b]). Element 11[b] is the apparatus equivalent of element 2[b].

10. Claim 12

460. As discussed above in Section X.A.8, Fu in view of the knowledge of a POSITA renders claim 10 obvious.

- a. Element 12[a]: The user equipment according to claim 10, wherein the deblocking filter unit further comprises a second clipping unit configured to clip said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels to be within an interval of zero and a defined maximum value by setting said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels to be equal to zero if said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels is smaller than zero and by setting said modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels to be equal to said defined maximum value if said modified pixel value of p_0' of said pixel closest to said block boundary in said line of pixels is larger than said defined maximum value and**

461. In my opinion, Fu in view of a POSITA's knowledge discloses or suggests element 12[a] for the same reasons as I discuss above in Section X.A.3.a (element 3[a]). Element 12[a] is the apparatus equivalent of element 3[a].

- b. Element 12[b]: to clip said modified pixel value q0' of said pixel closest to said block boundary in said corresponding line of pixels to be within an interval of zero and said defined maximum value by setting said modified pixel value q0' of said pixel closest to said block boundary in said corresponding line of pixels to be equal to zero if said modified pixel value q0' of said pixel closest to said block boundary in said corresponding line of pixels is smaller than zero and by setting said modified pixel value q0' of said pixel closest to said block boundary in said corresponding line of pixels to be equal to said defined maximum if said modified pixel value q0' of said pixel closest to said block boundary in said corresponding line of pixels is larger than said defined maximum value.**

462. In my opinion, Fu in view of a POSITA's knowledge discloses or suggests element 12[b] for the same reasons as I discuss above in Section X.A.3.b (element 3[b]). Element 12[b] is the apparatus equivalent of element 3[b].

B. Ground 2: Obviousness over Bjontegaard in view of Bjontegaard's Code and a POSITA's Understanding of the Art, Evidenced by List, Fu, B205, and B118

1. Claim 1

463. In my opinion, Claim 1 is invalid as obvious over Bjontegaard in view of the knowledge of a POSITA.

- a. **Element 1[pre]: A method of reducing blocking artifacts at a block boundary between a block of multiple pixels and a neighboring block of multiple pixels in a video frame, said method comprises:**

464. If the preamble is limiting, in my opinion, Bjontegaard discloses element 1[pre].

465. Bjontegaard discloses “a complete description of the Telenor coding model presented” at the Osaka meeting of Study Group 16, Question 15, and includes some differences from H.263, including:

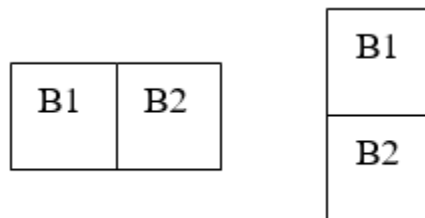
- Only one regular VLC is used for symbol coding
- $\frac{1}{4}$ pixel positions are used for motion prediction
- A number of different blocksizes [sic] are used for motion prediction
- Residual coding is based on 4x4 blocks and a [sic] integer transform is used
- Multiple reference frames may be used for prediction and this is considered to replace any use of B-frames.

Bjontegaard, 4. Bjontegaard’s disclosure and coding method are mainly defining decoder actions to decode video frames. *Id.*, 1, 4 (noting use of “CIF” and “QCIF” video frames).

466. Bjontegaard describes a “deblocking filter process [that] takes place both across vertical block edges and horizontal block edges.” Bjontegaard, 18. Part of Bjontegaard’s disclosed deblocking filter includes a method of “filtering of pixels

on both sides of a 4x4 block boundary.” Bjontegaard, 19. Bjontegaard’s disclosed deblocking filter would include a method for reducing blocking artifacts from the block boundary. *E.g.*, List, 616 (“In deblocking filtering, it is crucially important to be able to distinguish between true edges in the image and those created by quantization of the DCT coefficients.”). It was well-known that “using positive offsets to increase the amount of filtering can improve subjective quality on content where visible artifacts remain if the default values [of a deblocking filter] are used.” *Id.* Below, I explain how Bjontegaard uses offset equations to reduce blocking artifacts that would otherwise remain visible if default values of a deblocking filter are used.

467. Bjontegaard describes two exemplary blocks of pixels, B1 and B2, where they can be located “in one of the [following] positions:”



Bjontegaard, 18.

468. Bjontegaard uses a different pixel notation scheme than the ’659 Patent and Fu. Instead of using p0, p1, q0, q1, etc., Bjontegaard “[a]ssume[s] that the pixels a b c d e f g h belong to two neighbouring [sic] blocks (1 and 2)” where “[t]here is a

block boundary between pixels d and e.” Bjontegaard, 19. Bjontegaard’s pixel arrangement can be depicted below:



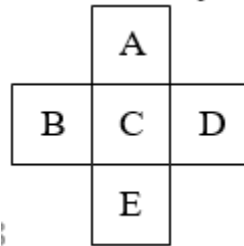
where pixels a, b, c, and d are part of a block of pixels and pixels e, f, g, and h are part of a neighboring block of pixels.

- b. **Element 1[a]: calculating a first offset to be equal to $(9 \times (q_0 - p_0) - 3 \times (q_1 - p_1) + 8) \gg 4$, wherein $\gg$ denotes a right shift operation, p_0 denotes a pixel value of a pixel closest to said block boundary in a line of pixels in said block, wherein said line of pixels is perpendicular to said block boundary, p_1 denotes a pixel value of a pixel next closest to said block boundary in said line of pixels, q_0 denotes a pixel value of a pixel closest to said block boundary in a corresponding line of pixels in said neighboring block, wherein said corresponding line of pixels is perpendicular to said block boundary and q_1 denotes a pixel value of a pixel next closest to said block boundary in said corresponding line of pixels;**

469. In my opinion, Bjontegaard in view of a POSITA’s knowledge discloses or suggests element 1[a].

470. Bjontegaard discloses a multi-step process for a deblocking filter to reduce blocking artifacts through the application of offset equations to modify at least pixels p_0 , q_0 , p_1 , and q_1 (pixels c, d, e, f in Bjontegaard). Bjontegaard, 18-20.

471. First, Bjontegaard discloses determining a block strength. Initially, “[e]ach 4x4 block in a reconstructed picture is assigned a Strength value ranging from 0 to 3” where the “value is assigned based on different criteria.” Bjontegaard, 18. Bjontegaard discloses that the following A, B, C, and D blocks are 4x4 blocks:



472. Bjontegaard assigns a strength value using the following procedure:

- Initially each 4x4 block of a picture is assigned **Strength = 0**
- If C is intra coded: **Strength_C = 3**, **Strength_A = max(Strength_A, 2)**, **Strength_B = max(Strength_B, 2)**, **Strength_D = max(Strength_D, 2)**, **Strength_E = max(Strength_E, 2)**
- If C has nonzero transform coefficients: **Strength_C = max(Strength_C, 2)**, **Strength_A = max(Strength_A, 1)**, **Strength_B = max(Strength_B, 1)**, **Strength_D = max(Strength_D, 1)**, **Strength_E = max(Strength_E, 1)**
- If the absolute difference between one of the vector components of block A and block C is at least one integer pixel (four ¼ pixels) then: **Strength_A = max(Strength_A, 1)**, **Strength_C = max(Strength_C, 1)**. The same procedure is applied to vector differences between blocks (B and C), (C and D) and (C and E)

Bjontegaard, 19.

473. Next, Bjontegaard calculates an “activity value” to determine how much filtering is performed. Bjontegaard discloses a series of “parameters” that are calculated to determine the activity value:

$$\begin{aligned}S0 &= \text{abs}(4(c-f)+12(e-d)) + \text{abs}(9(c-d-e+f)) \\S0_1 &= 12\text{abs}(a-2b+c) \\S1_1 &= \text{abs}(4(a-d)+12(c-b)) + \text{abs}(9(a-b-c+d)) \\S0_2 &= 12\text{abs}(e-2f+g) \\S1_2 &= \text{abs}(4(e-h)+12(g-f)) + \text{abs}(9(e-f-g+h))\end{aligned}$$

where abs() is the absolute value function. Bjontegaard, 19. Bjontegaard's activity value calculations are described above in Section VII.F.

474. I'll note that Bjontegaard has a typo in its $S0_1$ equation, where it points to a, b, and c as the three closest pixels to the block boundary rather than pixels b, c, and d. A POSITA would have recognized this typo and understand that d, for example, is the closest pixel to the block boundary, as represented below:



See also Bjontegaard at 19 (stating that “[t]here is a block boundary between pixels d and e”).

475. If the activity value, defined by Act_{edge} , is higher than zero then at least pixels d and e in Bjontegaard (pixels p0 and q0 in the '659 Patent) are filtered, such that a first offset is calculated and the pixel values are filtered to derive modified pixel values. Bjontegaard, 20. Specifically, if Act_{edge} equals 1 or 2, then a weak

filter is used and the following first offset equation is used to calculate the value of “Diff” (i.e., the offset to be applied to derive a modified pixel value):

$$Diff = (3(c - f) - 8(d - e))/16$$

Id.

476. Alternatively, if Act_{edge} equals three, then a strong filter is used and the following first offset equation is used to calculate the value of “Diff”:

$$Diff = (b + c - 8(d - e) - f - g)/16$$

Id.

477. Diff is then used to modify the pixels closest to the block boundary—here d and e, as shown below:



478. In Bjontegaard, pixel d represents a pixel value of a pixel closest to the block boundary line (bold black line, above) in a line of pixels a, b, c, d of a block of pixels, where the line of pixels a, b, c, d is perpendicular to the block boundary. Similarly, pixel e represents a pixel value of a pixel closest to the block boundary in a corresponding line of pixels e, f, g, h of a neighboring block of pixels, where the corresponding line of pixels e, f, g, h is perpendicular to the block boundary.

479. As stated above, if Act_{edge} equals 1 or 2, Bjontegaard teaches that a first offset equation is calculated to derive a value “Diff”

$$Diff = (3(c - f) - 8(d - e))/16$$

480. For the Bjontegaard ground, I use a similar color-coding scheme, where the coefficient of term $(d - e$; i.e., $p_0 - q_0$) is red; the coefficient of term $(c - f$; i.e., $p_1 - q_1$) is green; any numerical offset is blue and the divisor is orange.

481. The mathematical equivalent to Bjontegaard's first offset equation can be rewritten in terms of the pixel nomenclature of the '659 Patent, where the first offset equation is multiplied by a value of -1 and the terms of the equation are rearranged to match the order of the first offset equation in element 1[a]:

$$Diff = (8 \times (q_0 - p_0) - 3 \times (q_1 - p_1))/16$$

482. Bjontegaard teaches that Diff is used to derive a modified pixel value for pixel d and pixel e, where Diff may be clipped by the following equation:

$$Diff = \max(-Clip_{edge}, \min(Clip_{edge}, Diff))$$

$$d_{modified} = d + Diff \text{ (clipped to stay in the range 0-255)}$$

$$e_{modified} = e - Diff \text{ (clipped to stay in the range 0-255)}$$

Bjontegaard, 20.

483. Bjontegaard does not disclose use of a right shift operation. But, as discussed above in Section V.B.2.b, right shift operations were the approximate equivalents of multiplication and division processes (here, focusing on right shift operations (i.e., division)). Unlike division operations, which are computationally intensive, right shift operations are computationally efficient because they are

performed on the actual binary bits of data stored in a computer. Right shift operations are, however, less exact because they are rounded down to the next-lowest integer value and will never equate to a decimal or fraction of a remainder. To compensate for this output, a numerical offset is applied that is typically half of the denominator. For example, if a right shift of 2 is considered, which equates to division by 4 (i.e., 2^2), then a numerical offset of 2 is added to compensate for the integer-only nature of the result of a right shift operation. *See* Section V.B.2.b, *supra*.

484. The use of a right shift operation in the context of offset equations was well-known in the prior art. For example, List discloses its first offset equation in terms of a right shift by 3 operation. List, 617. As another example, B118 discloses its first offset equation in terms of right shift operations. B118, 48. As yet another example, B205 discloses its first offset equation in terms of right shift operations. B205, 113.

485. As shown above, Bjontegaard's first offset equation has a divisor of 16, which corresponds to a right shift of 4. Bjontegaard's first offset equation can be rewritten below, using a right shift operation:

$$Diff = (8 \times (q_0 - p_0) - 3 \times (q_1 - p_1) + 8) \gg 4$$

486. Bjontegaard teaches that the result of its first offset equation is clipped by the following function before it is used to derive modified pixel values for d and e:

$$Diff = \max(-Clip_{edge}, \min(Clip_{edge}, Diff))$$

Bjontegaard, 20. As discussed above in Ground 1, however, if the result of the first offset is within the bounds of the clipping function (e.g., $-t_c$ and t_c used in the '659 Patent), then the result of the first offset would not be altered, and therefore, would be the value of Diff. *See* Section X.A.1.b.

487. A comparison of Bjontegaard's first offset expression (with the clipping function omitted) and the first offset expression recited in element 1[a] is below:

Reference	First Offset Equation
'659 Patent	$(9 \times (q_0 - p_0) - 3 \times (q_1 - p_1) + 8) \gg 4$
Bjontegaard	$(8 \times (q_0 - p_0) - 3 \times (q_1 - p_1) + 8) \gg 4$

488. The only difference between the first offset expression of Bjontegaard and the first offset expression of the '659 Patent is the coefficient of the term $(q_0 - p_0)$, shown in red above, where Bjontegaard uses a coefficient of 8 and the first offset of the '659 Patent uses a coefficient of 9. As discussed in Section X.A.1.b, however, modifying this result effective variable would have been a result of routine experimentation and optimization by modifying known result-effective variables,

and therefore the equation would have been apparent to a POSITA at the time of the invention.

489. For Bjontegaard, a POSITA would have experimented with its first offset equation (through an empirical trial-and-error process, which is how these filters were typically designed), and specifically with other coefficient values that were close to those in Bjontegaard, to achieve a desired amount of filtering of blocking artifacts without also degrading the overall image quality. And, in my opinion, a POSITA would have found the claimed offset equation obvious because the coefficients are so close—the claimed coefficients (9 and 3) can be obtained by increasing Bjontegaard’s (8 and 3) by one value—that a POSITA would have expected only slight differences between their filtering performance, as shown below through some example applications¹⁶ of the two equations:

¹⁶ I have included the p2 pixel in these examples, even though it is not used in the equations, because I later use the same sets of pixels to evaluate the differences between Bjontegaard and the claimed p1 offset equations.

p2	p1	p0	q0	q1	Type	$\frac{9(q0 - p0) - 3(q0 - p0)}{16}$	$\frac{8(q0 - p0) - 3(q0 - p0)}{16}$
200	150	150	150	150	Step (p1)	0	0
200	200	150	150	150	Step (p0)	9	9
200	200	200	150	150	Step (q0)	-19	-16
150	160	170	180	190	Ramp (10)	0	-1
125	150	175	200	225	Ramp (25)	0	-2
200	250	200	250	200	Alternating	38	34
255	50	50	255	255	Line	77	64
255	255	50	50	255	Line	0	0
175	200	150	150	175	Random	5	5
50	125	100	150	125	Random	28	25
75	100	150	75	125	Random	-47	-42

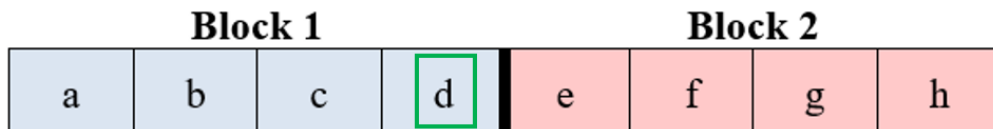
490. A POSITA would have appreciated that coefficients 9 and 3 place slightly more emphasis on the difference between the p0 and q0 pixels relative to the difference between the p1 and q1 pixels in defining the offset (and therefore tend to apply *slightly* more filtering as the difference between these pixels increases), while still meeting all of the other offset-equation considerations: (1) it maintained a ratio of 3 that is between 2 and 4 for $X(q0-p0)$; $Y(q1-p1)$; (2) it would produce an offset of approximately $\frac{3}{8}$, just slightly more than Bjontegaard's original offset equation

of $\frac{1}{4}$, and still between $\frac{1}{4}$ and $\frac{1}{2}$; (3) it used four pixels; and (4) it maintained a DC gain of 1 (as shown in Tables 2 and 3 from Section X.A.1.b).

- c. **Element 1[b]: modifying said pixel value p_0 of said pixel closest to said block boundary in said line of pixels by adding said first offset to said pixel value p_0 of said pixel closest to said block boundary in said line of pixels to form a modified pixel value p_0' of said pixel closest to said block boundary in said line of pixels; and**

491. In my opinion, Bjontegaard in view of a POSITA's knowledge discloses or suggests element 1[b].

492. Bjontegaard discloses that the modified pixel value of d (i.e., p_0'), which is shown below (boxed in green), is the pixel closest to the block boundary in the line of pixels d, c, b, a , and that d_{modified} is derived by adding the result of the first offset to the value of pixel d .



493. Bjontegaard's equation to calculate the modified value of d (i.e., d_{modified} ; p_0' in terms of the '659 Patent) is reproduced below:

$$d_{\text{modified}} = d + \text{Diff (clipped to stay in the range 0-255)}$$

Bjontegaard, 20.

494. As discussed above in Section X.B.1.b, Diff is the result of Bjontegaard's first offset equation, which is added to the value of pixel d as shown in the equation above. The result is d_{modified} , which is the modified pixel value of d (i.e., p_0' in terms of element 1[b]), which is the pixel value closest to the block boundary in the line of pixels d, c, b, a.

495. Bjontegaard discloses that the result of $d + \text{Diff}$ is clipped to stay within the range of 0 to 255. Bjontegaard, 20.

496. Applying the logic of Bjontegaard's clipping function, the following conditions are recognized:

- If the result of $d + \text{Diff}$ is less than 0; then d_{modified} (i.e., the modified value of d) is 0;
- If the result of $d + \text{Diff}$ is greater than 255, then d_{modified} is 255;
- If the result of $d + \text{Diff}$ is within the range of 0-255, then d_{modified} is $d + \text{Diff}$.

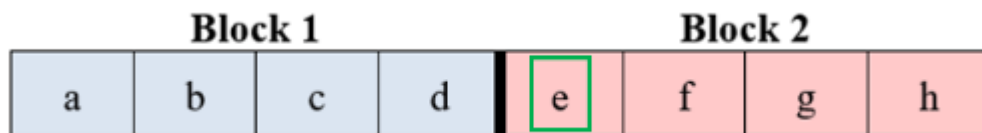
Bjontegaard, 20.

497. In the third condition above, the modified value of pixel p_0 (in Bjontegaard, d_{modified}) is the sum of the result of the first offset equation and pixel value d.

- d. **Element 1[c]: modifying said pixel value q_0 of said pixel closest to said block boundary in said corresponding line of pixels by subtracting said first offset from said pixel value q_0 of said pixel closest to said block boundary in said corresponding line of pixels to form a modified pixel value q_0' of said pixel closest to said block boundary in said corresponding line of pixels.**

498. In my opinion, Bjontegaard in view of a POSITA's knowledge discloses or suggests element 1[c].

499. Bjontegaard discloses that the modified pixel value of e (i.e., q_0'), which is shown below (boxed in green), is the pixel closest to the block boundary in the line of pixels d, c, b, a , and that e_{modified} is derived by subtracting the result of the first offset to the value of pixel e .



500. Bjontegaard's equation to calculate the modified value of e (i.e., e_{modified} ; q_0' in terms of the '659 Patent) is reproduced below:

$$e_{\text{modified}} = e - \text{Diff} \text{ (clipped to stay in the range 0-255)}$$

Bjontegaard, 20.

501. As discussed above in Section X.B.1.b, Diff is the result of Bjontegaard's first offset equation, which is subtracted from the value of pixel e as shown in the equation above. The result is e_{modified} , which is the modified pixel value

of e (i.e., q_0' in terms of element $1[c]$), which is the pixel value closest to the block boundary in the corresponding line of pixels e, f, g, h in the neighboring block of pixels.

502. Bjontegaard discloses that the result of $e - \text{Diff}$ is clipped to stay within the range of 0 to 255.

503. Applying the logic of Bjontegaard's clipping function, the following conditions are recognized:

- If the result of $e - \text{Diff}$ is less than 0; then e_{modified} (i.e., the modified value of e) is 0;
- If the result of $e - \text{Diff}$ is greater than 255, then e_{modified} is 255;
- If the result of $e - \text{Diff}$ is within the range of 0-255, then e_{modified} is $e - \text{Diff}$.

504. In the third condition above, the modified value of pixel q_0 (in Bjontegaard, e_{modified}) is the difference of the result of the first offset equation and pixel value e .

2. Claim 2

505. As discussed above in Section X.B.1, Bjontegaard in view of the knowledge of a POSITA renders claim 1 obvious.

- a. **Element 2[a]: The method according to claim 1, further comprising clipping said first offset to be within an interval of $-t_c$ and t_c by setting said first offset to be equal to $-t_c$ if said first offset is smaller than $-t_c$ and by setting said first offset to be equal to t_c if said first offset is larger than t_c ,**

506. In my opinion, Bjontegaard discloses or suggests element 2[a].

507. As discussed above in Section X.B.1.b, Bjontegaard teaches that its first offset is clipped using Bjontegaard's $Clip_{edge}$ equation. Bjontegaard's equation to derive its first offset, $Diff$, is reproduced below, along with the $Clip_{edge}$ function:

$$Diff = (3(c - f) - 8(d - e))/16$$

$$Diff = \max(-Clip_{edge}, \min(Clip_{edge}, Diff))$$

Bjontegaard, 20.

508. Bjontegaard teaches that $Diff$ is clipped to remain with a minimum and maximum value (i.e., an interval of $-t_c$ and t_c in terms of element 2[a]. Bjontegaard's $Clip_{edge}$ function is reproduced below:

$$Clip_{edge} = (Clip(qp, Strength_1) + Clip(qp, Strength_2) + Act_1 + Act_2 - 2)/2$$

Bjontegaard, 19-20.

509. Bjontegaard's $Clip_{edge}$ function, in turn, relies on Bjontegaard's $Clip(qp, Strength)$ function. Specifically, Bjontegaard's $Clip_{edge}$ function sums the value of (1) its $Clip$ function for the value of $strength_1$, (2) its $Clip$ function for the

value of strength_2 , (3) Act_1 , (4) Act_2 , and (5) negative 2, before dividing that value by 2 to arrive at the result. Bjontegaard, 19-20.

510. Bjontegaard's $\text{Clip}(\text{qp}, \text{Strength})$ function is reproduced below:

$$\text{Clip}(\text{qp}, \text{Strength})$$

511. The value of Bjontegaard's $\text{Clip}(\text{qp}, \text{Strength})$ function that is used to derive the first two values used in its $\text{Clip}_{\text{edge}}$ function can be taken from the following table, dependent on the qp value. Bjontegaard, 19-20.

QP	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	29	30	31
$\text{Clip}(\text{qp}, 1)$	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	2	2	2	2	2	3	3
$\text{Clip}(\text{qp}, 2)$	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	2	2	2	2	3	3	3	3	4	5	5
$\text{Clip}(\text{qp}, 3)$	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	2	2	2	2	3	3	3	4	4	5	5	5	7	8	9

512. Bjontegaard's Act values are also dependent on qp, where they are derived using the following set of equations. Bjontegaard teaches that these steps are used to get the value for both Act_1 and Act_2 .

4.5.4 Activity on each side – Act_1 , Act_2

- Act_1 is initially set to 1
- If $S0_1 < \text{Lim}(\text{qp})$, Act_1 is incremented by 1
- If $S1_1 < \text{Lim}(\text{qp})$, Act_1 is incremented by 1

The same procedure is made for Act_2 .

513. As discussed in Section VII.F, the variables “ $S0_1$ ” and “ $S1_1$ ” then represent different measures of activity in block 1, with $S0_1$ being the second-order derivative (or difference-of-difference) of three pixels in block 1, and the variable

“S1₁” represents a weighted measure of activity that includes an additional pixel in block 1. The same is true for variables “S0₂” and “S1₂,” except these variables use pixels from block 2 instead of block 1.

514. Finally, Bjontegaard discloses that “Lim(qp) is a table defining threshold values for the filtering process depending on qp. As pixels a to h belong to two blocks, qp of the two blocks may differ. In this context qp means the maximum qp for the two blocks.” Bjontegaard, 19. The table to derive the Lim(qp) value is:

QP	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Lim	7	8	9	10	11	12	14	15	17	20	22	25	28	31	35	39	44	49	55	62	69	78	88	98	110	124	139	156	175	197	221	248

Bjontegaard, 19.

515. After the value for Clip_{edge} is derived by the above steps, the first offset is clipped to remain within an interval of -Clip_{edge} and Clip_{edge}, where the following conditions are applied:

- First, take the minimum of values Clip_{edge} and Diff using the equation $\min(\text{Clip}_{\text{edge}}, \text{Diff})$:
 - If Diff is greater than Clip_{edge}, then the resulting value is Clip_{edge}
 - If Diff is less than Clip_{edge}, then the resulting value is Diff

- Second, take the maximum of the values $Clip_{edge}$ and the result of minimum equation (discussed above) using the equation $\max(-Clip_{edge}, \min(Clip_{edge}, Diff))$:
 - If the result of $\min(Clip_{edge}, Diff)$ is less than $-Clip_{edge}$, the resulting value is $-Clip_{edge}$
 - If the result of $\min(Clip_{edge}, Diff)$ is greater than $Clip_{edge}$, the resulting value is the result of $\min(Clip_{edge}, Diff)$ (i.e., either $Diff$ or $Clip_{edge}$)

Bjontegaard, 20.

516. Turning again to Bjontegaard's equation to derive $Diff$, $Diff$ is clipped to remain within the bounds of $-Clip_{edge}$ (i.e., $-t_C$ in terms of element 2[a]) and $Clip_{edge}$ (i.e., t_C in terms of element 2[a]) as discussed above.

$$Diff = \max(-Clip_{edge}, \min(Clip_{edge}, Diff))$$

Id.

- b. Element 2[b]: wherein t_C is a threshold value that is dependent on a quantization parameter value assigned to said block.**

517. In my opinion, Bjontegaard discloses or suggests element 2[b].

518. Bjontegaard teaches that "[t]he quantization/dequantization process shall perform 'normal' quantization/dequantization as well as take care of the fact

that transform operations are kept very simple and therefore do not contain normalization of transform coefficients.” Bjontegaard, 17. Bjontegaard further teaches that the QP value indicates step size, where, for example, an “[i]ncrease of QP by 6 means that the step size is about doubled.” Bjontegaard, 17. Bjontegaard’s quantization process, however, “is designed so that no overflow will occur with the use of 32 bit arithmetic.” Bjontegaard, 18.

519. For its deblocking filter, Bjontegaard teaches that its quantization parameter value, qp, is a value assigned to the two blocks used as an example of Bjontegaard. Bjontegaard, 19. Bjontegaard, however, teaches that “qp of the two blocks may differ” so in the context of its Lim(qp) table, “qp means the maximum qp for the two blocks.” Bjontegaard, 19.

520. As discussed above in Section X.B.2.a, Bjontegaard’s clipping function, Clip_{edge}, depends on activity calculations that, in turn, depend on the quantization parameter, qp. Those equations are reproduced below:

If $S0_1 < \text{Lim}(\text{qp})$, Act₁ is incremented by 1
If $S1_1 < \text{Lim}(\text{qp})$, Act₁ is incremented by 1

where the value of Lim(qp) is derived from the table below:

QP	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Lim	7	8	9	10	11	12	14	15	17	20	22	25	28	31	35	39	44	49	55	62	69	78	88	98	110	124	139	156	175	197	221	248

Bjontegaard, 19.

521. Bjontegaard's $\text{Clip}_{\text{edge}}$ function depends on five values, including the values of Act_1 and Act_2 that, in turn, are dependent on the maximum qp assigned to blocks 1 and 2. Bjontegaard, 17-20. If block 1's qp value is greater than block 2's qp value, then Bjontegaard's $\text{Clip}_{\text{edge}}$ function depends on the value of block 1, (i.e., "said block" recited in element 2[b]).

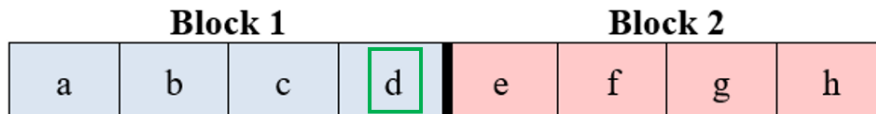
3. Claim 3

522. As discussed above in Section X.B.1, Bjontegaard in view of the knowledge of a POSITA renders claim 1 obvious.

- a. **Element 3[a]: The method according to claim 1, further comprising: clipping said modified pixel value $p0'$ of said pixel closest to said block boundary in said line of pixels to be within an interval of zero and a defined maximum value by setting said modified pixel value $p0'$ of said pixel closest to said block boundary in said line of pixels to be equal to zero if said modified pixel value $p0'$ of said pixel closest to said block boundary in said line of pixels is smaller than zero and by setting said modified pixel value $p0'$ of said pixel closest to said block boundary in said line of pixels to be equal to said defined maximum value if said modified pixel value $p0'$ of said pixel closest to said block boundary in said line of pixels is larger than said defined maximum value; and**

523. In my opinion, Bjontegaard in view of a POSITA's knowledge discloses or suggests element 3[a].

524. Bjontegaard teaches that its modified pixel value d_{modified} (i.e., $p0'$ in terms of element 3[a]) is clipped. As discussed in Section X.B.1 and shown in the figure below, d_{modified} (i.e., $p0'$) is the modified pixel value of d , which is the pixel closest to the block boundary in the line of pixels d, c, b, a .



525. Bjontegaard teaches that d_{modified} is the sum of the value of pixel d and the result of Bjontegaard's first offset equation, Diff . Bjontegaard further teaches that this sum is "clipped to stay in the range 0-255" as shown in the equation below:

$$d_{\text{modified}} = d + \text{Diff} \text{ (clipped to stay in the range 0-255)}$$

Bjontegaard, 20.

526. Applying the logic of Bjontegaard's clipping disclosure, the following conditions are recognized:

- If the result of $d + \text{Diff}$ is less than 0; then d_{modified} (i.e., the modified value of d) is 0;
- If the result of $d + \text{Diff}$ is greater than 255, then d_{modified} is 255;
- If the result of $d + \text{Diff}$ is within the range of 0-255, then d_{modified} is $d + \text{Diff}$.

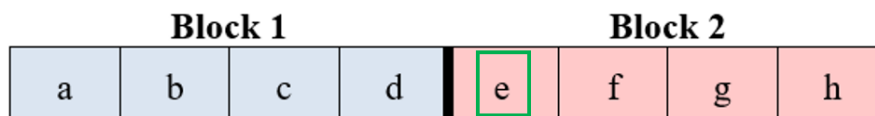
527. Here, as discussed in Section V.B.1, 255 is defined by the maximum value that an 8-bit pixel can be (2^8-1). Bjontegaard, therefore, discloses to a POSITA that 255 is a defined maximum.

528. For example, B205 teaches that 255 is a common defined maximum value of a clip function for pixels. Ex-1001, 2:58-3:3; Section V.B.1 (discussing pixel values). Similarly, the '659 Patent discloses that for its clipping function: “The parameter M denotes a defined maximum value that the modified pixel values can assume. In a particular embodiment, this maximum value is defined based on the number of bits that are spent on the pixel values. Thus, assume that the pixel values are in the form of a m-bit value, then M is preferably equal to 2^m-1 . For instance, if each pixel value is a 8-bit value, i.e., $m=8$, then $M=255$.” Ex-1001, 11:64-12:17.

- b. **Element 3[b]: clipping said modified pixel value $q0'$ of said pixel closest to said block boundary in said corresponding line of pixels to be within an interval of zero and said defined maximum value by setting said modified pixel value $q0'$ of said pixel closest to said block boundary in said corresponding line of pixels to be equal to zero if said modified pixel value $q0'$ of said pixel closest to said block boundary in said corresponding line of pixels is smaller than zero and by setting said modified pixel value $q0'$ of said pixel closest to said block boundary in said corresponding line of pixels to be equal to said defined maximum if said modified pixel value $q0'$ of said pixel closest to said block boundary in said corresponding line of pixels is larger than said defined maximum value.**

529. In my opinion, Bjontegaard in view of a POSITA's knowledge discloses or suggests element 3[b].

530. Bjontegaard teaches that its modified pixel value e_{modified} (i.e., $q0'$ in terms of element 3[b]) is clipped. As discussed in Section X.B.1 and shown in the figure below, e_{modified} (i.e., $q0'$) is the modified pixel value of e, which is the pixel closest to the block boundary in the corresponding line of pixels e, f, g, h in the neighboring block of pixels.



531. Bjontegaard teaches that e_{modified} is the result of the value of pixel e minus the result of Bjontegaard's first offset equation, Diff. Bjontegaard further

teaches that this result is “clipped to stay in the range 0-255” as shown in the equation below:

$$e_{modified} = e - Diff \text{ (clipped to stay in the range 0-255)}$$

Bjontegaard, 20.

532. Applying the logic of Bjontegaard’s clipping disclosure, the following conditions are recognized:

- If the result of $e - Diff$ is less than 0; then $e_{modified}$ (i.e., the modified value of e) is 0;
- If the result of $e - Diff$ is greater than 255, then $e_{modified}$ is 255;
- If the result of $e - Diff$ is within the range of 0-255, then $e_{modified}$ is $e - Diff$.

533. Here, as discussed in Section V.B.1, 255 is defined by the maximum value that an 8-bit pixel can be (2^8-1). Bjontegaard, therefore, discloses to a POSITA that 255 is a defined maximum.

534. For example, B205 teaches that 255 is a common defined maximum value of a clip function for pixels. Ex-1001, 2:58-3:3; Section V.B.1 (discussing pixel values). Similarly, the ’659 Patent discloses that for its clipping function: “The parameter M denotes a defined maximum value that the modified pixel values can assume. In a particular embodiment, this maximum value is defined based on the number of bits that are spent on the pixel values. Thus, assume that the pixel values

are in the form of a m-bit value, then M is preferably equal to 2^m-1 . For instance, if each pixel value is a 8-bit value, i.e., m=8, then M=255.” Ex-1001, 11:64-12:17.

4. Claim 4

535. As discussed above in Section X.B.1, Bjontegaard in view of the knowledge of a POSITA renders claim 1 obvious.

- a. **Element 4[a]: The method according to claim 1, further comprising: calculating a second offset to be based on $(p_2+p_0[-]2\times p_1+2\times\Delta)/4$, wherein p_2 denotes a pixel value of a pixel second next closest to said block boundary in said line of pixels and Δ denotes said first offset;**

536. In my opinion, Bjontegaard in view of Bjontegaard’s Code and a POSITA’s knowledge discloses or suggests element 4[a].

537. Bjontegaard teaches that a second offset can be calculated to derive a modified value of pixel c (i.e., pixel p1 in the ’659 Patent)—the next closest pixel to the block boundary in the line of pixels d, c, b, a.

538. Bjontegaard discloses that after d and e are filtered, a decision is made whether to filter c in pixel block 1 and/or f in pixel block 2. Bjontegaard, 20. As with pixel d, Bjontegaard’s decision on whether to filter pixel c and how much to filter pixel c depends on activity value, here Act_{11} .

539. When Act_{11} equals 2, Bjontegaard teaches that a second offset should be calculated using the following equation:

$$Diff = 5(b - 2c + d)/16$$

Bjontegaard, 20.

540. As shown above, Bjontegaard's second offset equation relies on consecutive pixel values d, c, and b, where pixel b is the second next closest pixel to the block boundary in the line of pixels d, c, b, a, as shown in the figure below:



541. However, in my opinion, a POSITA would have understood that Bjontegaard's equation would be read as: $\text{Diff}_{[1]} = 5(b - 2c + d_{\text{modified}})/16$, because Bjontegaard provides a logical walkthrough of the deblocking filter's operation, in which each section logically follows those that precede it. The reason that Bjontegaard re-initializes the variable "Diff" to equal zero in Sections §§ 4.5.6.1 and 4.5.6.2 is because Bjontegaard uses the same variable "Diff" (for efficiency) for multiple calculations. Without re-initializing "Diff" at the start of each section, the value of "Diff" previously stored in memory from calculations in a preceding section would be used.

542. The same reasoning therefore applies to the value of pixels d and e calculated in Section § 4.5.5. Although Bjontegaard labels the modified value of d as d_{modified} , logically it is just a redefined value of d (*i.e.*, $d = d + \text{Diff}$) because d_{modified} will replace d in the image.

543. Bjontegaard's Code would have confirmed this understanding, because Bjontegaard discloses modifying pixel d *before* then filtering pixel c. *See* Ex-1014, 11. Specifically, Bjontegaard's Code labels the first offset using variable "id" and pixel d using variable "li[3]." *Id.* The offset "id" is then applied to "li[3]" using equations: "il1=img->li[3]+id" (to calculate the new pixel value) and "img->li[3] = il1" (to apply the new pixel value to li[3] (pixel d)). Only then does filtering proceed to pixel c (li[2]) using the new value of pixel d (li[3]) *Id.* (identified by "/* center left */" section of code). In my opinion, therefore, Bjontegaard discloses using the *modified* value of d in the equation $\text{Diff}_{[1]} = 5(b - 2c + d_{\text{modified}}) / 16$.

544. Even if Bjontegaard did not expressly disclose this, in my opinion, it would have been obvious to modify Bjontegaard to use d_{modified} instead of d. That is because, as was common in the art, filtering pixels further from the boundary should depend on how much pixels closer to the boundary are filtered. *See, e.g.,* H.241, 29 (modifying p0 using Δ , and p1 using $\Delta/2$); B205, 131 (same); Ex-1039, 215 (applying $\Delta/2$, $\Delta/4$, $\Delta/8$ when moving further from boundary). If the offset applied to pixels further from the boundary did not depend on the offset applied to pixels closer to the boundary, that could result in new unwanted discontinuities in the vicinity of the boundary as a result of the deblocking filter, which would be contrary to the intended purpose of deblocking to eliminate such discontinuities. By using d_{modified} instead of d, therefore, Bjontegaard would ensure the new pixel value for

pixel c is determined based on the modified pixel d_{modified} and thus has some relationship to the offset applied to pixel d.

545. Applying this reasoning, Bjontegaard's equation either was or would obviously be re-written as follows:

$$\text{Diff}_c = \frac{5(b - 2c + d_{\text{modified}})}{16}$$

where $d_{\text{modified}} = d + \text{Diff}_1$:

$$= \frac{5(b - 2c + d + \text{Diff}_1)}{16}$$

$$= \frac{5(b - 2c + d)}{16} + \frac{5 * \text{Diff}_1}{16}$$

And re-writing this equation with delta and p, consistent with the '659 Patent:

$$\Delta_{p1} = \frac{5(p2 - 2p1 + p0)}{16} + \frac{5\Delta_{p0}}{16}$$

546. In my opinion, a POSITA would have therefore recognized that the second-offset equation Bjontegaard discloses has two distinct components: (1) a high-pass filter component ($\frac{5(p2-2p1+p0)}{16}$), evident from the Laplacian identified in blue; and (2) a first-offset component ($\frac{5\Delta}{16}$) identified by the first offset (Δ). The only difference between Bjontegaard's second-offset equation and the one found in 4[a] is the coefficients applied to each of these two components:

Claimed Equation	Bjontegaard's Equation
$\Delta = \frac{1(p_2 - 2p_1 + p_0)}{4} + \frac{1\Delta}{2}$	$\Delta = \frac{5(p_2 - 2p_1 + p_0)}{16} + \frac{5\Delta}{16}$

547. In my opinion, for the same reasons I discussed above with respect to the coefficients applied to each pixel difference in the first-offset equations, *see* Section X.A.1.b (element 1[a], Ground 1) and Section X.B.1.b (element 1[a], Ground 2), a POSITA would have also recognized that the coefficients applied to each of these components in the second-offset equation are variables¹⁷ subject to experimentation and optimization because they too control the frequency response of the second-offset filter. In fact, several other prior-art references combined these same two components but applied different coefficients to each of them to achieve their desired level of filtering.

548. Fu, for instance, discloses the following equation to modify the p1 pixel:

$$P_1 = \text{Clip1} (p_1 + \text{Clip3} (-C, C ((P_0 - p_1) \times 3 + (p_2 - Q_0) + 4) >> 3))$$

Fu, 8:17; Section X.A.4 (claim 4, Ground 1). This equation can be re-written (without clipping, for simplicity) as follows:

$$\Delta = \{3(q_0 - p_0) + (p_1 - q_1) + 4\} >> 3\}$$

¹⁷ As I discuss in further detail, another variable subject to routine experimentation was the exact high-pass filter equation used for the high-pass component.

$$P_1 = p_1 + \{[3(\textcolor{red}{P0} - p_1) + (p_2 - \textcolor{red}{Q0}) + 4] \gg 3\}$$

$$\text{where } \textcolor{red}{P0} = p_0 + \Delta$$

$$\textcolor{red}{Q0} = q_0 - \Delta$$

$$= p_1 + \{[3(\textcolor{red}{p0} + \Delta - p_1) + (p_2 - \textcolor{red}{q0} + \Delta) + 4] \gg 3\}$$

$$= p_1 + (p_2 - 3p_1 + 3p_0 - q_0 + 4\Delta + 4) \gg 3$$

$$= p_1 + \frac{p_2 - 3p_1 + 3p_0 - q_0 + 4\Delta}{8}$$

$$= p_1 + \frac{p_2 - 3p_1 + 3p_0 - q_0}{8} + \frac{\Delta}{2}$$

See generally Section V.B.3.b (describing conversion of bit-shift form to division form). Fu therefore discloses a second-offset equation of: $\frac{p_2 - 3p_1 + 3p_0 - q_0}{8} + \frac{\Delta}{2}$.

549. The only difference between Bjontegaard and Fu is which high-pass filter—the other variable subject to experimentation that I noted above—is used to characterize additional high frequencies for filtering ($p_2 - 3p_1 + 3p_0 - q_0$ for Fu, compared to $p_2 - 2p_1 + p_0$ for Bjontegaard), and how much weight is applied to each component ($1/8$ for Fu’s high-pass component and $1/2$ for Fu’s first-offset component, compared to $5/16$ for both of Bjontegaard’s components).

550. Similarly, during the Joint Video Team (“JVT”) proceedings to develop the H.264 standard, the JVT proposed as part of the draft standard using a p_1 offset equation that combined a scaled Laplacian high-pass filter with an offset. B118, 48.

B118 presented the following equation, using “p1” as a notation for the pixel closest to the block boundary, which I have converted to p0 notation for consistency:

$$\begin{aligned} P0 &= p0 + \Delta \\ P1 &= p1 + (p2 + \textcolor{red}{P0} - p1 \ll 1) \gg 1) \\ &= p1 + (p2 + \textcolor{red}{p0} + \Delta - p1 \ll 1) \gg 1) \\ &= p1 + \frac{p2 - 2p1 + p2 + \Delta}{2} \\ &= p1 + \frac{p2 - 2p1 + p2}{2} + \frac{\Delta}{2} \end{aligned}$$

B118 therefore discloses a second-offset equation of $\frac{p2 - 2p1 + p0}{2} + \frac{\Delta}{2}$.

551. The only difference between Bjontegaard and B118 is how much weight is applied to the Laplacian and offset components (1/2 for B118, compared to 5/16 for Bjontegaard, for each).

552. Because other prior art proposed second-offset equations that included the same two components as Bjontegaard, with different coefficients applied to each, in my opinion, a POSITA would have been motivated to experiment and optimize Bjontegaard’s coefficients to achieve a desired balance between filtering blocking artifacts and maintaining overall image quality, just like with the first-offset equation. And, in my opinion, a POSITA would have found the claimed second-offset equation obvious because the coefficients are close enough—the claimed weights (1/4 and 1/2) can be obtained by adjusting each of Bjontegaard’s component weights

(5/16) by a few fractions—that a POSITA would have expected only slight differences between their filtering performance, as shown below through some example applications of the two equations (using each of the first-offset values calculated above, Section X.B.1.b (element 1[a]), for each respective offset):

p2	p1	p0	q0	q1	Type	$\frac{1(p2 - 2p1 + p0)}{4} + \frac{1\Delta}{2}$	$\frac{5(p2 - 2p1 + p0)}{16} + \frac{5\Delta}{16}$
200	150	150	150	150	Step (p1)	13	16
200	200	150	150	150	Step (p0)	-8	-13
200	200	200	150	150	Step (q0)	-10	-5
150	160	170	180	190	Ramp (10)	0	0
125	150	175	200	225	Ramp (25)	0	-1
200	250	200	250	200	Alternating	-6	-21
255	50	50	255	255	Line	90	84
255	255	50	50	255	Line	-51	-64
175	200	150	150	175	Random	-16	-22
50	125	100	150	125	Random	-11	-23
75	100	150	75	125	Random	-17	-5

553. Specifically, a POSITA would have understood that the high-pass filter component is intended to characterize high-frequency pixel changes present near the block boundary for additional filtering, and adjusting the weight of this component would control how much those high frequencies would be filtered. A POSITA

would have therefore expected that modifying the coefficient from $5/16$ to $1/4$ —which is just a difference of $1/16^{\text{th}}$ —would have meant that *slightly* less additional high-frequency filtering would be applied by the second offset, and therefore *slightly* less blurring would occur (at least due to this component).

554. A POSITA would have also understood that the first-offset component is intended to apply some fraction of the same offset applied to pixels closer to the block boundary to pixels further from the boundary to create a smoother transition, and adjusting the weight of this component would control how much of the first offset is carried over to the second offset to further blur the blocking artifact. *See* Ex-1039, 215 (applying $\Delta/2$, $\Delta/4$, $\Delta/8$ when moving further from boundary). A POSITA would have therefore expected that modifying the weight from $5/16$ to $1/2$ —which is just the difference of $3/16^{\text{ths}}$ —would have meant that *slightly* more of the first offset would have applied as part of the second offset, and therefore *slightly* more blurring would occur (at least due to this component). In sum, weighting each of these components relative to one another controls the *overall* amount of filtering performed by the second offset.

555. For these reasons, in my opinion, a POSITA would have been motivated to experiment with the coefficients applied to each of the high-pass component and first-offset component—as Bjontegaard, Fu, and B118 demonstrate—to achieve different amounts of filtering, knowing that they controlled

how much each component contributed to the second offset, and would have found applying a weight of $\frac{1}{4}$ to the high-pass Laplacian component, and $\frac{1}{2}$ to the first-offset component obvious.

556. Moreover, a POSITA would have recognized that the p1 (and q1 addressed in further detail below) offset equations do not produce any unexpected results for the reasons I discussed previously. *See* Section X.A.1.b (element 1[a], Ground 1); Section X.B.1.b (element 1[a], Ground 2).

557. In my opinion, a POSITA also would have had a reasonable expectation of success in designing a filter equation for the p1 and q1 pixels because the prior art was replete with examples, the design considerations were known, and it was largely understood to be an empirical exercise of testing different filter coefficients to arrive at a desired outcome (*e.g.*, overall image quality and deblocking performance, which depends on the amount of the offset).

b. Element 4[b]: calculating a third offset to be based on $(q2+q0-2\times q1-2\times \Delta)/4$, wherein q2 denotes a pixel value of a pixel second next closest to said block boundary in said corresponding line of pixels;

558. In my opinion, Bjontegaard in view of Bjontegaard's Code and a POSITA's knowledge discloses or suggests element 4[b].

559. Bjontegaard teaches that its third offset can be calculated to derive the value of pixel f (i.e., q1)—the next closest pixel to the block boundary in the corresponding line of pixels d, e, f, g.

560. Bjontegaard discloses that after d and e are filtered, a decision is made whether to filter c in pixel block 1 and/or f in pixel block 2. Bjontegaard, 20. Similar to pixel c, Bjontegaard's decision on whether to filter pixel f and how much to filter pixel f depends on activity value, here Act_{22} .

561. When Act_{22} equals 2, Bjontegaard teaches that a third offset should be calculated using the following formula:

$$Diff = 5(e - 2f + g)/16$$

Bjontegaard, 20.

562. As shown above, Bjontegaard's third offset equation relies on consecutive pixel values e, f, g, where pixel g is the second next closest pixel to the block boundary in the corresponding line of pixels e, f, g, h, as shown in the figure below:



563. As I discussed previously, however, a POSITA would have known or found it obvious that Bjontegaard's equation uses pixel e_{modified} . See Section X.B.4.a

(element 4[a]). Like with pixel d, Bjontegaard's Code would have confirmed this understanding, because Bjontegaard discloses modifying pixel e *before* then filtering pixel f. *See* Ex-1012, 11. Specifically, Bjontegaard's Code labels the first offset using variable "id" and pixel e using variable "li[4]." *Id.* The offset "id" is then applied to "li[4]" using equations: "ir1=img->li[4]-id" (to calculate the new pixel value) and "img->li[4] = ir1" (to apply the new pixel value to li[4] (pixel e)). Only then does filtering proceed to pixel f (li[5]) using the new value of pixel e (li[4]). *Id.* (identified by "/* center left */" section of code). Applying this reasoning, Bjontegaard's equation either was or would obviously be re-written as follows:

$$\text{Diff}_f = \frac{5(e_{\text{modified}} - 2f + g)}{16}$$

where $e_{\text{modified}} = e - \text{Diff}_2$:

$$\begin{aligned} &= \frac{5(e - \text{Diff}_2 - 2f + g)}{16} \\ &= \frac{5(e - 2f + g)}{16} - \frac{5 * \text{Diff}_2}{16} \end{aligned}$$

564. And re-writing this equation with delta and q, consistent with the '659

Patent:

$$\Delta_{q1} = \frac{5(q2 - 2q1 + q0)}{16} - \frac{5\Delta_{q0}}{16}$$

565. However, like the second offset, similar equations had been used in the prior art, which differed only as to: (1) the specific high-pass filter component used; and (2) the coefficients assigned to each component.

Claimed Equation	Bjontegaard's Equation
$\Delta = \frac{1(q_2 - 2q_1 + q_0)}{4} - \frac{1\Delta}{2}$	$\Delta = \frac{5(q_2 - 2q_1 + q_0)}{16} - \frac{5\Delta}{16}$

566. For instance, Fu discloses the following filter equation incorporating a third offset:

$$Q_1 = \text{Clip1}(q_1 - \text{Clip3}(-C, C((q_1 - Q_0) \times 3 + (P_0 - q_2) + 4) \gg 3))$$

Fu, 8:20; *see also* Section X.A.4.b (element 4[b], Ground 1). Just like with the P1 equation, the Q1 equation can be written as:

$$\begin{aligned}
 \Delta &= \{3(q_1 - Q_0) + (P_0 - q_2) + 4\} \gg 3\} \\
 Q_1 &= q_1 - \{[3(q_1 - Q_0) + (P_0 - q_2) + 4] \gg 3\} \\
 &\text{where } P_0 = p_0 + \Delta \\
 Q_0 &= q_0 - \Delta \\
 &= q_1 - \{[3(q_1 - q_0 + \Delta) + (p_0 + \Delta - q_2) + 4] \gg 3\} \\
 &= q_1 - (p_0 - 3q_0 + 3q_1 - q_2 + 4\Delta + 4) \gg 3 \\
 &= q_1 - \frac{p_0 - 3q_0 + 3q_1 - q_2 + 4\Delta}{8} \\
 &= q_1 - \frac{p_0 - 3q_0 + 3q_1 - q_2}{8} + \frac{\Delta}{2}
 \end{aligned}$$

$$= q1 + \frac{q2-3q1+3q0-p0}{8} - \frac{\Delta}{2}$$

567. Similarly, B118 proposed a third-offset equation that combined a scaled Laplacian high-pass filter into the offset with “ $\Delta/2$ ”:

$$Q1 = q1 + (q2 + Q0 - q1 \ll 1) \gg 1)$$

$$\text{where } Q0 = q0 - \Delta$$

$$= q1 + (q2 + q0 - \Delta - q1 \ll 1) \gg 1)$$

$$= q1 + \frac{q2-2q1+q2-\Delta}{2}$$

$$= q1 + \frac{q2-2q1+q2}{2} - \frac{\Delta}{2}$$

568. As I also discussed previously, a POSITA would have experimented with the coefficients used in the offset equation, as the coefficients were known variables that affect the amount that pixels near block boundaries were filtered without degrading overall image quality. A POSITA would have found the offset equation slightly different than what already existed in the prior art and would have been a matter of routine experimentation and optimization. *See* Section X.B.4.a.

- c. **Element 4[c]: modifying said pixel value p1 of said pixel next closest to said block boundary in said line of pixels by adding said second offset to said pixel value p1 of said pixel next closest to said block boundary in said line of pixels to form a modified pixel value p1' of said pixel next closest to said block boundary in said line of pixels; and**

569. In my opinion, Bjontegaard in view of Bjontegaard's Code and a POSITA's knowledge discloses or suggests element 4[c].

570. Bjontegaard discloses that the modified value of pixel c, which as shown in the below figure is the pixel next closest to the block boundary in the line of pixels d, c, b, a, is derived by adding the result of the second offset to the value of pixel c.



571. Bjontegaard's equation to calculate the modified value of c (i.e., c_{modified} or p1') is reproduced below:

$$c_{\text{modified}} = c + \text{Diff} \text{ (clipped to stay in the range 0-255)}$$

$$\text{Diff} = 5(b - 2c + d)/16$$

Bjontegaard, 20.

572. Bjontegaard discloses that the result of its second offset equation is clipped before it is added to the value of pixel c to derive c_{modified} . Bjontegaard's clipping equation is reproduced below:

$$Diff = \max(-Clip(qp, Strength_1), \min(Clip(qp, Strength_1), Diff))$$

Bjontegaard, 20.

573. Bjontegaard's $Clip(qp, Strength)$ function is reproduced below, along with "tables depending on qp and $Strength$ for clipping modifications":

$$Clip(qp, Strength)$$

QP	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	29	30	31
$Clip(qp, 1)$	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	2	2	2	2	2	3	3
$Clip(qp, 2)$	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	2	2	2	2	3	3	3	3	4	5	5
$Clip(qp, 3)$	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	2	2	2	2	3	3	3	4	4	5	5	5	7	8	9

Bjontegaard, 19-20.

574. After the value for $Clip(qp, Strength_1)$ is derived, the second offset is clipped to remain within an interval of $-Clip(qp, Strength_1)$, $Clip(qp, Strength_1)$, where the following conditions are applied:

- First, take the minimum of values $Clip(qp, Strength_1)$ and $Diff$ using the equation $\min(Clip(qp, Strength_1), Diff)$:
 - If $Diff$ is greater than $Clip(qp, Strength_1)$, then the resulting value is $Clip(qp, Strength_1)$

- If Diff is less than $\text{Clip}(qp, \text{Strength}_1)$, then the resulting value is Diff
- Second, take the maximum of the values $\text{Clip}(qp, \text{Strength}_1)$ and the result of the minimum equation (discussed above) using the equation $\max(-\text{Clip}(qp, \text{Strength}_1), \min(\text{Clip}(qp, \text{Strength}_1), \text{Diff}))$:
 - If the result of $\min(\text{Clip}(qp, \text{Strength}_1), \text{Diff})$ is less than $-\text{Clip}(qp, \text{Strength}_1)$, then the resulting value is $(-\text{Clip}(qp, \text{Strength}_1))$
 - If the result of $\min(\text{Clip}(qp, \text{Strength}_1), \text{Diff})$ is greater than $-\text{Clip}(qp, \text{Strength}_1)$, then the resulting value is $\min(\text{Clip}(qp, \text{Strength}_1), \text{Diff})$ (i.e., either Diff or $\text{Clip}(qp, \text{Strength}_1)$)

575. If Diff is less than $\text{Clip}(qp, \text{Strength}_1)$ and greater than $-\text{Clip}(qp, \text{Strength}_1)$, then Diff is added to the value of pixel c , which will result in c_{modified} .

- d. **Element 4[d]: modifying said pixel value q1 of said pixel next closest to said block boundary in said corresponding line of pixels by adding said third offset to said pixel value q1 of said pixel next closest to said block boundary in said corresponding line of pixels to form a modified pixel value q1' of said pixel next closest to said block boundary in said corresponding line of pixels.**

576. In my opinion, Bjontegaard in view of Bjontegaard's Code and a POSITA's knowledge discloses or suggests element 4[d].

577. Bjontegaard discloses that the modified value of pixel f, which as shown in the figure below is the pixel next closest to the block boundary in the corresponding line of pixels e, f, g, h, is derived by adding the result of the third offset value to the value of pixel f.



578. Bjontegaard's equation to calculate the modified value of f (i.e., f_{modified} or f_1') is reproduced below.

$$f_{\text{modified}} = f + \text{Diff} \text{ (clipped to stay in the range 0-255)}$$

$$\text{Diff} = 5(e - 2f + g)/16$$

Bjontegaard, 20.

579. Bjontegaard discloses that the result of its third offset equation is clipped before it is added to the value of pixel f to derive f_{modified} . Bjontegaard's clipping equation is reproduced below:

$$Diff = \max(-Clip(qp, Strength_2), \min(Clip(qp, Strength_2), Diff))$$

Bjontegaard, 20.

580. Bjontegaard's $Clip(qp, Strength)$ function is reproduced below, along with "tables depending on qp and $Strength$ for clipping modifications":

$$Clip(qp, Strength)$$

QP	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	29	30	31
$Clip(qp, 1)$	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	2	2	2	2	2	3	3
$Clip(qp, 2)$	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	2	2	2	2	3	3	3	3	4	5	5
$Clip(qp, 3)$	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	2	2	2	2	3	3	3	4	4	5	5	5	7	8	9

581. After the value for $Clip(qp, Strength_1)$ is derived, the third offset is clipped to remain within an interval of $-Clip(qp, Strength_2)$, $Clip(qp, Strength_2)$, where the following conditions are applied:

- First, take the minimum of values $Clip(qp, Strength_2)$ and $Diff$ using the equation $\min(Clip(qp, Strength_2), Diff)$:
 - If $Diff$ is greater than $Clip(qp, Strength_2)$, then the resulting value is $Clip(qp, Strength_2)$
 - If $Diff$ is less than $Clip(qp, Strength_2)$, then the resulting value is $Diff$

- Second, take the maximum of the values $Clip(qp, Strength_2)$ and the result of the minimum equation (discussed above) using the equation $\max(-Clip(qp, Strength_2), \min(Clip(qp, Strength_2), Diff))$:
 - If the result of $\min(Clip(qp, Strength_2), Diff)$ is less than $-Clip(qp, Strength_2)$, then the resulting value is $(-Clip(qp, Strength_2))$
 - If the result of $\min(Clip(qp, Strength_2), Diff)$ is greater than $-Clip(qp, Strength_2)$, then the resulting value is $\min(Clip(qp, Strength_2), Diff)$ (i.e., either $Diff$ or $Clip(qp, Strength_2)$)

582. If $Diff$ is less than $Clip(qp, Strength_2)$ and greater than $-Clip(qp, Strength_2)$, then $Diff$ is added to the value of pixel f , which will result in f_{modified} .

5. Claim 5

583. As discussed above in Section X.B.4, Bjontegaard in view of the knowledge of a POSITA renders claim 4 obvious.

- a. **Element 5[a]: The method according to claim 4, wherein calculating said second offset comprises calculating said second offset to be equal to $((p_2 + p_0 + 1) \gg 1) - p_1 + \Delta \gg 1$, wherein $\gg$ denotes a right shift operation; and**

584. In my opinion, Bjontegaard in view of a POSITA's knowledge discloses or suggests element 5[a].

585. The '659 Patent discloses that the second offset can be calculated using a traditional division form, or can alternatively be calculated in a "hardware-adapted implementation" using a right shift function instead of division. Ex-1001, 15:41-51.

586. As I discuss above in Section X.B.4.a, in my opinion Bjontegaard renders the second-offset equation from element 4[a] obvious. The second-offset equation recited here uses a right-shift function instead of division but that is the obvious substitute for a hardware implementation. The second-offset from element 4[a] is reproduced below as a starting point:

$$= [(p_2 + p_0 - 2 \times p_1 + 2 \times \Delta)]/4$$

587. The mathematical equivalent of the second-offset expression from element 4[a] is reproduced below, where the entire expression is divided by two:

$$= \left[\left(\frac{1}{2} p_2 + \frac{1}{2} p_0 - p_1 + \Delta \right) \right] / 2$$

588. If the like $\frac{1}{2}$ terms, p_2 and p_0 , are combined, then that results in the mathematically equivalent expression:

$$= \left\lceil \left(\frac{1}{2} (p_2 + p_0) - p_1 + \Delta \right) \right\rceil / 2$$

589. From here, a POSITA would have recognized that a right-shift function can be substituted for the division functions in the expression above. That would result in the following:

$$= \left(((p_2 + p_0) \gg 1) - p_1 + \Delta \right) \gg 1$$

590. As discussed in Section V.B.2.b above, however, the output of a right-shift function is always an integer. As such, the right-shift function truncates the value of the result obtained after the division function to the greatest integer whose value is less than or equal to that result, and thus introduces some inaccuracies to the result. It is well known to a POSITA that those inaccuracies can be reduced by adding an offset to the value prior to the right-shift function whose value is equal to half of the value of the denominator. If you consider a denominator of 4 (i.e., a right-shift of 2), then the rounding value that is added to offset is 2. This approach results in an approximation of the result obtained by the division function value by producing the integer whose value is closest to that result.

591. Here, the $p_2 + p_0$ term is right shifted by 1 bit, which is the hardware alternative to dividing by 2 (i.e., 2^1). To compensate for the inaccuracies introduced by this operation, an offset value of $2/2$ (or 1) is added to the $p_2 + p_0$ term prior to the right-shift function, which would result in:

$$= ((p_2 + p_0 + 1) \gg 1) - p_1 + \Delta \gg 1$$

592. As explained above, the output of a right-shift operation is always rounded down to the integer value that is less than or equal to the value obtained by the division function. To round the output to the nearest integer, an offset value is added prior to the right-shift function, which has a value that is half of the value of the denominator. A POSITA would have understood that whether or not to include an offset value is a design choice of whether the result of the right shift is rounded to the nearest integer value or truncated to the greatest integer whose value is less than or equal to that result. Here, to round the output of $(p_2 + p_0) \gg 1$ to the nearest integer, an offset value of 1 is added prior to the right-shift function.

593. Applying the hardware alternative to the second-offset expression of element 4[a] arrives at the second-offset expression recited in element 5[a]. Bjontegaard renders element 5[a] obvious for the same reasons discussed above in Section X.B.4.a.

b. Element 5[b]: calculating said third offset comprises calculating said third offset to be equal to $((q_2 + q_0 + 1) \gg 1) - q_1 + \Delta \gg 1$

594. In my opinion, Bjontegaard in view of a POSITA's knowledge discloses or suggests element 5[b].

595. As I discuss above in Section X.B.4.b, in my opinion Bjontegaard renders the third-offset equation from element 4[b] obvious. The third-offset

expression recited here uses a right-shift function instead of division but that is the obvious substitute for a hardware implementation. The third-offset from element 4[b] is reproduced below as a starting point:

$$= [(q_2 + q_0 - 2 \times q_1 - 2 \times \Delta)]/4$$

596. The mathematical equivalent of the third-offset expression from element 4[b] is reproduced below, where the entire expression is divided by two:

$$= \left[\left(\frac{1}{2}q_2 + \frac{1}{2}q_0 - q_1 - \Delta \right) \right] / 2$$

597. If the like $\frac{1}{2}$ terms, q_2 and q_0 , are combined, then that results in the mathematically equivalent expression:

$$= \left[\left(\frac{1}{2}(q_2 + q_0) - q_1 - \Delta \right) \right] / 2$$

598. From here, a POSITA would have recognized that a right-shift function can be substituted for the division functions in the equation above. That would result in the following:

$$= \left(((q_2 + q_0) \gg 1) - q_1 - \Delta \right) \gg 1$$

599. As discussed in Section V.B.2.b above, however, the output of a right-shift function is always an integer. As such, the right-shift function truncates the value of the result obtained after the division function to the greatest integer whose value is less than or equal to that result, and thus introduces some inaccuracies to the

result. It is well known to a POSITA that those inaccuracies can be reduced by adding an offset to the value prior to the right-shift function whose value is equal to half of the value of the denominator. If you consider a denominator of 4 (i.e., a right-shift of 2), then the rounding value that is added to offset is 2. This approach results in an approximation of the result obtained by the division function value by producing the integer whose value is closest to that result.

600. Here, the $q_2 + q_0$ term is right shifted by 1 bit, which is the hardware alternative to dividing by 2 (i.e., 2^1). To compensate the inaccuracies introduced by this operation, an offset value of $2/2$ (or 1) is added to the $q_2 + q_0$ term prior to the right-shift function, which would result in:

$$= ((q_2 + q_0 + 1) \gg 1) - q_1 - \Delta \gg 1$$

601. As explained above, the output of a right-shift operation is always rounded down to the integer value that is less than or equal to the value obtained by the division function. To round the output to the nearest integer, an offset value is added prior to the right-shift function, which has a value that is half of the value of the denominator. A POSITA would have understood that whether or not to include an offset value is a design choice of whether the result of the right shift is rounded to the nearest integer value or truncated to the greatest integer whose value is less than or equal to that result. Here, to round the output of $(q_2 + q_0) \gg 1$ to the nearest integer, an offset value of 1 is added prior to the right-shift function.

602. Applying the hardware alternative to the third-offset expression of element 4[b] arrives at the third-offset expression recited in element 5[b]. Bjontegaard renders element 5[b] obvious for the same reasons discussed above in Section X.B.4.b.

6. Claim 6

603. As discussed above in Section X.B.4, Bjontegaard in view of the knowledge of a POSITA renders claim 4 obvious.

- a. **Element 6[a]: The method according to claim 4, further comprising: clipping said second offset to be within an interval of $-t_{C2}$ and t_{C2} by setting said second offset to be equal to $-t_{C2}$ if said second offset is smaller than $-t_{C2}$ and by setting said second offset to be equal to t_{C2} if said second offset is larger than t_{C2} ,**

604. In my opinion, Bjontegaard discloses or suggests element 6[a].

605. Bjontegaard discloses that the result of its second offset equation is clipped before it is added to the value of pixel c to derive c_{modified} . Bjontegaard's clipping equation is reproduced below:

$$Diff = \max(-Clip(qp, Strength_1), \min(Clip(qp, Strength_1), Diff))$$

Bjontegaard, 20.

606. Bjontegaard's $Clip(qp, Strength)$ function is reproduced below, along with "tables depending on qp and $Strength$ for clipping modifications":

$$Clip(qp, Strength)$$

qp	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	29	30	31
Clip(qp,1)	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	2	2	2	2	2	3	3
Clip(qp,2)	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	2	2	2	2	3	3	3	3	4	5	5
Clip(qp,3)	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	2	2	2	2	3	3	3	4	4	5	5	5	7	8	9

607. After the value for $\text{Clip}(qp, \text{Strength}_1)$ is derived, the second offset is clipped to remain within an interval of $-\text{Clip}(qp, \text{Strength}_1)$, $\text{Clip}(qp, \text{Strength}_1)$, where the following conditions are applied:

- First, take the minimum of values $\text{Clip}(qp, \text{Strength}_1)$ and Diff using the equation $\min(\text{Clip}(qp, \text{Strength}_1), \text{Diff})$:
 - If Diff is greater than $\text{Clip}(qp, \text{Strength}_1)$, then the resulting value is $\text{Clip}(qp, \text{Strength}_1)$
 - If Diff is less than $\text{Clip}(qp, \text{Strength}_1)$, then the resulting value is Diff
- Second, take the maximum of the values $\text{Clip}(qp, \text{Strength}_1)$ and the result of the minimum equation (discussed above) using the equation $\max(-\text{Clip}(qp, \text{Strength}_1), \min(\text{Clip}(qp, \text{Strength}_1), \text{Diff}))$:
 - If the result of $\min(\text{Clip}(qp, \text{Strength}_1), \text{Diff})$ is less than $-\text{Clip}(qp, \text{Strength}_1)$, then the resulting value is $(-\text{Clip}(qp, \text{Strength}_1))$

- If the result of $\min(\text{Clip}(qp, \text{Strength}_1), \text{Diff})$ is greater than $-\text{Clip}(qp, \text{Strength}_1)$, then the resulting value is $\min(\text{Clip}(qp, \text{Strength}_1), \text{Diff})$ (i.e., either Diff or $\text{Clip}(qp, \text{Strength}_1)$)

608. Applying the logic of Bjontegaard's clipping equation, if the result of Bjontegaard's second offset expression is less than $-\text{Clip}(qp, \text{Strength}_1)$ (or $-t_{c2}$ in terms of element 6[a]), then the result of the second offset expression is set to $-\text{Clip}(qp, \text{Strength}_1)$. Conversely, if the result of Bjontegaard's second offset expression is greater than $\text{Clip}(qp, \text{Strength}_1)$ (or t_{c2} in terms of element 6[a]), then the result of the second offset expression is set to $\text{Clip}(qp, \text{Strength}_1)$.

b. Element 6[b]: wherein $-t_{c2}$ is a threshold value that is dependent on a quantization parameter value assigned to said block; and

609. In my opinion, Bjontegaard discloses or suggests element 6[b].

610. Bjontegaard teaches that “[t]he quantization/dequantization process shall perform ‘normal’ quantization/dequantization as well as take care of the fact that transform operations are kept very simply and therefore do not contain normalization of transform coefficients.” Bjontegaard, 17. Bjontegaard further teaches that the QP value indicates step size, where, for example, an “[i]ncrease of QP by 6 means that the step size is about doubled.” Bjontegaard, 17. Bjontegaard's

quantization process, however, “is designed so that no overflow will occur with the use of 32 bit arithmetic.” Bjontegaard, 18.

611. For its deblocking filter, Bjontegaard teaches that its quantization parameter value, qp , is a value assigned to the two blocks used as an example of Bjontegaard. Bjontegaard, 19. Bjontegaard further teaches that its clipping function for the second offset depends directly on qp , which is assigned to the block 1. Bjontegaard, 19 (teaching that the “ qp of the two blocks may differ”). Bjontegaard clarifies that “ qp ” means the maximum qp for the two blocks. *Id.* The threshold value of $-Clip(qp, Strength_1)$ is, therefore, dependent on the qp value assigned to block 1 when it is greater than the qp value assigned to block 2.

$$Diff = \max(-Clip(qp, Strength_1), \min(Clip(qp, Strength_1), Diff))$$

Bjontegaard, 20.

- c. **Element 6[c]: clipping said third offset within an interval of $-t_{C2}$ and t_{C2} by setting said third offset to be equal to $-t_{C2}$ if said third offset is smaller than $-t_{C2}$ and by setting said third offset to be equal to t_{C2} if said third offset is larger than t_{C2} .**

612. In my opinion, Bjontegaard in view of a POSITA’s knowledge discloses or suggests element 6[c].

613. Bjontegaard discloses that the result of its third offset equation is clipped before it is added to the value of pixel f to derive f_{modified} . Bjontegaard’s clipping equation is reproduced below:

$$Diff = \max(-Clip(qp, Strength_2), \min(Clip(qp, Strength_2), Diff))$$

Bjontegaard, 20.

614. Bjontegaard's $Clip(qp, Strength)$ function is reproduced below, along with "tables depending on qp and Strength for clipping modifications":

$$Clip(qp, Strength)$$

QP	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	29	30	31
$Clip(qp, 1)$	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	2	2	2	2	2	3	3
$Clip(qp, 2)$	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	2	2	2	2	3	3	3	3	4	5	5
$Clip(qp, 3)$	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	2	2	2	2	3	3	3	4	4	5	5	5	7	8	9

615. After the value for $Clip(qp, Strength_1)$ is derived, the third offset is clipped to remain within an interval of $-Clip(qp, Strength_2)$, $Clip(qp, Strength_2)$, where the following conditions are applied:

- First, take the minimum of values $Clip(qp, Strength_2)$ and Diff using the equation $\min(Clip(qp, Strength_2), Diff)$:
 - If Diff is greater than $Clip(qp, Strength_2)$, then the resulting value is $Clip(qp, Strength_2)$
 - If Diff is less than $Clip(qp, Strength_2)$, then the resulting value is Diff
- Second, take the maximum of the values $Clip(qp, Strength_2)$ and the result of the minimum equation (discussed above) using

the equation
 $\max(-\text{Clip}(qp, \text{Strength}_2), \min(\text{Clip}(qp, \text{Strength}_2), \text{Diff})):$

- If the result of $\min(\text{Clip}(qp, \text{Strength}_2), \text{Diff})$ is less than $-\text{Clip}(qp, \text{Strength}_2)$, then the resulting value is $(-\text{Clip}(qp, \text{Strength}_2))$
- If the result of $\min(\text{Clip}(qp, \text{Strength}_2), \text{Diff})$ is greater than $-\text{Clip}(qp, \text{Strength}_2)$, then the resulting value is $\min(\text{Clip}(qp, \text{Strength}_2), \text{Diff})$ (i.e., either Diff or $\text{Clip}(qp, \text{Strength}_2)$)

616. Applying the logic of Bjontegaard's clipping equation, if the result of Bjontegaard's third offset expression is less than $-\text{Clip}(qp, \text{Strength}_2)$ (or $-t_{c2}$ in terms of element 6[c]), then the result of the third offset expression is set to $-\text{Clip}(qp, \text{Strength}_2)$. Conversely, if the result of Bjontegaard's third offset expression is greater than $\text{Clip}(qp, \text{Strength}_2)$ (or t_{c2} in terms of element 6[c]), then the result of the third offset expression is set to $\text{Clip}(qp, \text{Strength}_2)$.

617. Bjontegaard teaches using different strength values to clip the result of its second offset equation and the result of its third offset equation, where Strength_1 and Strength_2 are used respectively.

618. The strength table disclosed in Bjontegaard is reproduced below:

qp	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	29	30	31
Clip(qp,1)	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	2	2	2	2	2	3	3
Clip(qp,2)	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	2	2	2	2	3	3	3	3	4	5	5
Clip(qp,3)	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	2	2	2	2	3	3	3	4	4	5	5	5	7	8	9

619. One can see that in most instances, the result of the function $Clip(qp, Strength_1)$ and $Clip(qp, Strength_2)$ is the same (e.g., when qp is 0-9, 13-19). Bjontegaard, therefore, discloses that, in some instances, the same threshold values will be used to clip the second and third offset values.

620. Moreover, a POSITA would have been motivated to use the same clipping thresholds to clip the result of Bjontegaard's second and third offset equations.

621. For example, as discussed above in Section X.A.6, Fu teaches clipping the second and third offset equations by the same values.

622. As yet another example, B118 teaches clipping its second and third offset equations using the same thresholds—C0, C0—as shown in the equations below (as noted elsewhere, B118 uses a different pixel notation than the '659 Patent, Fu, and B205, where p1/q1 are the pixels closest to the block boundary and p2/q2 are the pixels next closest to the block boundary):

$$P_2 = p_2 + Clip(-C0, C0, (p_3 + P_1 - p_2 \ll 1) \gg 1)$$

$$Q_2 = q_2 + Clip(-C0, C0, (q_3 + Q_1 - q_2 \ll 1) \gg 1)$$

B118, 48 (§5.4.2).

623. As shown in Fu, B205, and B118, it was well-known to use the same clipping thresholds to clip the results of the second and third offset equations. Doing so would allow for a natural ramp for the clipped pixels c_{modified} , d_{modified} , e_{modified} , and f_{modified} that results in a smoother filtered image. See Ex-1039, 214-15, Fig. 4 (below). As discussed above in Section X.B.2, the first offset for pixels d and e in Bjontegaard are clipped using the same threshold. A POSITA would have been motivated to do the same here.

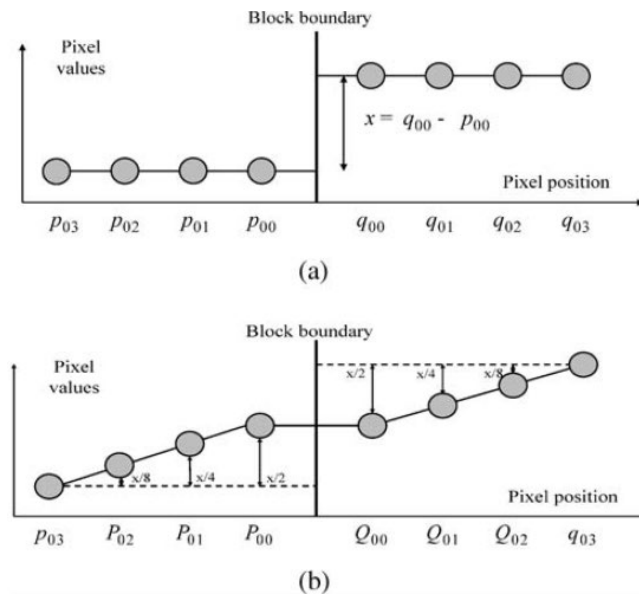


Figure 4. One dimensional view of the block boundary by offset-based filtering mode. a Before filtering. b After filtering.

7. Claim 7

624. As discussed above in Section X.B.4, Bjontegaard in view of the knowledge of a POSITA renders claim 4 obvious.

- a. **Element 7[a]: The method according to claim 4, further comprising: clipping said modified pixel value $p1'$ of said pixel next closest to said block boundary in said line of pixels to be within an interval of zero and a defined maximum value by setting said modified pixel value $p1'$ of said pixel next closest to said block boundary in said line of pixels to be equal to zero if said modified pixel value $p1'$ of said pixel next closest to said block boundary in said line of pixels is smaller than zero and by setting said modified pixel value $p1'$ of said pixel next closest to said block boundary in said line of pixels to be equal to said defined maximum value if said modified pixel value $p1'$ of said pixel next closest to said block boundary in said line of pixels is larger than said defined maximum value; and**

625. In my opinion, Bjontegaard discloses or suggests element 7[a].

626. Bjontegaard teaches that its modified pixel value c_{modified} (i.e., $p1'$ in terms of element 7[a]) is clipped. As shown in the figure below, c_{modified} is the modified value of c , which is the pixel next closest to the block boundary in the line of pixels d, c, b, a .



627. Bjontegaard teaches that c_{modified} is clipped to remain within the bounds of 0-255.

$$c_{\text{modified}} = c + \text{Diff} \text{ (clipped to stay in the range 0-255)}$$

Bjontegaard, 20.

628. Applying the logic of Bjontegaard's clipping disclosure, the following conditions are recognized:

- If the result of $c + \text{Diff}$ is less than 0; then c_{modified} (i.e., the modified value of c) is 0;
- If the result of $c + \text{Diff}$ is greater than 255, then c_{modified} is 255;
- If the result of $c + \text{Diff}$ is within the range of 0-255, then c_{modified} is $c + \text{Diff}$.

629. Here, as discussed in Section V.B.1, 255 is defined by the maximum value that an 8-bit pixel can be (2^8-1). Bjontegaard, therefore, discloses to a POSITA that 255 is a defined maximum.

630. For example, B205 teaches that 255 is a common defined maximum value of a clip function for pixels. Ex-1001, 2:58-3:3; Section V.B.1 (discussing pixel values). Similarly, the '659 Patent discloses that for its clipping function: "The parameter M denotes a defined maximum value that the modified pixel values can assume. In a particular embodiment, this maximum value is defined based on the number of bits that are spent on the pixel values. Thus, assume that the pixel values are in the form of a m -bit value, then M is preferably equal to 2^m-1 . For instance, if each pixel value is a 8-bit value, i.e., $m=8$, then $M=255$." Ex-1001, 11:64-12:17.

- b. **Element 7[b]: clipping said modified pixel value $q1'$ of said pixel next closest to said block boundary in said corresponding line of pixels to be within an interval of zero and said defined maximum value by setting said modified pixel value $q1'$ of said pixel next closest to said block boundary in said corresponding line of pixels to be equal to zero if said modified pixel value $q1'$ of said pixel next closest to said block boundary in said corresponding line of pixels is smaller than zero and by setting said modified pixel value $q1'$ of said pixel next closest to said block boundary in said corresponding line of pixels to be equal to said defined maximum if said modified pixel value $q1'$ of said pixel next closest to said block boundary in said corresponding line of pixels is larger than said defined maximum value.**

631. In my opinion, Bjontegaard discloses or suggests element 7[b].

632. Bjontegaard teaches that its modified pixel value f_{modified} (i.e., $q1'$ in terms of element 7[b]) is clipped. As shown in the figure below, f_{modified} is the modified value of f , which is the pixel next closest to the block boundary in the corresponding line of pixels e. f. g. h.



633. Bjontegaard teaches that f_{modified} is clipped to remain within the bounds of 0-255.

$$f_{\text{modified}} = f + \text{Diff} \text{ (clipped to stay in the range 0-255)}$$

Bjontegaard, 20.

634. Applying the logic of Bjontegaard's clipping disclosure, the following conditions are recognized:

- If the result of $f + \text{Diff}$ is less than 0; then f_{modified} (i.e., the modified value of f) is 0;
- If the result of $f + \text{Diff}$ is greater than 255, then f_{modified} is 255;
- If the result of $f + \text{Diff}$ is within the range of 0-255, then f_{modified} is $f + \text{Diff}$.

635. Here, as discussed in Section V.B.1, 255 is defined by the maximum value that an 8-bit pixel can be (2^8-1). Bjontegaard, therefore, discloses to a POSITA that 255 is a defined maximum.

636. For example, B205 teaches that 255 is a common defined maximum value of a clip function for pixels. Ex-1001, 2:58-3:3; Section V.B.1 (discussing pixel values). Similarly, the '659 Patent discloses that for its clipping function: "The parameter M denotes a defined maximum value that the modified pixel values can assume. In a particular embodiment, this maximum value is defined based on the number of bits that are spent on the pixel values. Thus, assume that the pixel values are in the form of a m -bit value, then M is preferably equal to 2^m-1 . For instance, if each pixel value is a 8-bit value, i.e., $m=8$, then $M=255$." Ex-1001, 11:64-12:17.

8. Claim 10

- a. Element 10[pre]: A user equipment comprising a deblocking filter unit for reducing blocking artifacts at a block boundary between a block of multiple pixels and a neighboring block of multiple pixels in a video frame, wherein the deblocking filter unit comprises:**

637. In my opinion, Bjontegaard in view of Bjontegaard's Code and a POSITA's knowledge discloses or suggests element 10[pre] under Patent Owner's proposed construction at the ITC—that the entire preamble is limiting. It follows, that 10[pre] would also be obvious under Lenovo's construction, which agrees that the preamble is limiting with the exception of “user equipment comprising.”

638. Although Bjontegaard does not detail the exact device that operates the coding model, a POSITA would have understood that it would be run on a “device”—such as a computer or other digital processing devices—because video codecs are conventionally run on such devices. *See, e.g.*, Ex-1061, 9:49-57 (describing “[a]n electronic device for deblocking and edge enhancement”), 10:9-13 (“Those of ordinary skill would further appreciate that the various illustrative logical blocks, modules, and algorithm steps ... may be implemented as electronic hardware, computer software, or combinations of both.”); Ex-1062, 18:17-18 (“a deblocking filter is utilized for loop filtering”), 22:23-24 (providing “a computer system with which some embodiments of the invention is implemented”).

639. In fact, as part of Bjontegaard's disclosure, Bjontegaard provided the computer code by which to operate the coding model on a computer. Ex-1010. A POSITA would have also understood that Bjontegaard's Code was intended to operate the coding model on a computer because the code included executable computer files (.exe) as well as the underlying source code files (.c) to operate on a computer. *See, e.g.*, Ex-1010. Bjontegaard's Code also includes instructions for how to operate the code on a computer. *See* Ex-1015, 1 (describing usage of encoder and decoder by running "DOS command prompt[s]"). A POSITA would have, therefore, understood that Bjontegaard would be run on a "device."

640. Moreover, video codecs with deblocking filters are conventionally run on devices such as computers and other digital processing devices (*e.g.*, smartphones, physical-media players, etc.). *See, e.g.*, Ex-1040, 1-5 (describing importance of video codecs, and particularly H.264, to various consumer electronics); Ex-1061, 9:49-57 (describing "[a]n electronic device for deblocking and edge enhancement"), 10:9-13 ("Those of ordinary skill would further appreciate that the various illustrative logical blocks, modules, and algorithm steps ... may be implemented as electronic hardware, computer software, or combinations of both."); Ex-1062, 18:17-18 ("a deblocking filter is utilized for loop filtering"), 22:23-24 (providing "a computer system with which some embodiments of the invention is implemented").

641. For this reason, even if Bjontegaard does not expressly disclose a “filtering control device,” a POSITA would have known that Bjontegaard was intended to be implemented in a device, and would have been motivated to implement Bjontegaard accordingly.

- b. Element 10[a]: a first offset calculator configured to calculate a first offset to be equal to $(9x(q_0 - p_0) - 3x(q_1 - p_1) + 8) \gg 4$, wherein $\gg$ denotes a right shift operation, p_0 denotes a pixel value of a pixel closest to said block boundary in a line of pixels in said block, wherein said line of pixels is perpendicular to said block boundary, p_1 denotes a pixel value of a pixel next closest to said block boundary in said line of pixels, q_0 denotes a pixel value of a pixel closest to said block boundary in a corresponding line of pixels in said neighboring block, wherein said corresponding line of pixels is perpendicular to said block boundary, and q_1 denotes a pixel value of a pixel next closest to said block boundary in said corresponding line of pixels; and**

642. In my opinion, Bjontegaard in view of a POSITA’s knowledge discloses or suggests element 10[a] for the same reasons I discuss above in Section X.B.1.b (element 1[a]). Element 10[a] is the apparatus equivalent of element 1[a].

- c. **Element 10[b]: a pixel value modifier configured to modify said pixel value p0 of said pixel closest to said block boundary in said line of pixels by adding said first offset to said pixel value p0 of said pixel closest to said block boundary in said line of pixels to form a modified pixel value p0' of said pixel closest to said block boundary in said line of pixels**

643. In my opinion, Bjontegaard in view of a POSITA's knowledge discloses or suggests element 10[b] for the same reasons I discuss above in Section X.B.1.c (element 1[b]). Element 10[b] is the apparatus equivalent of element 1[b].

- d. **Element 10[c]: and to modify said pixel value q0 of said pixel value q0 of said pixel closest to said block boundary in said corresponding line of pixels to form a modified pixel value q0' of said pixel closest to said block boundary in said corresponding line of pixels.**

644. In my opinion, Bjontegaard in view of a POSITA's knowledge discloses or suggests element 10[c] for the same reasons I discuss above in Section X.B.1.d (element 1[c]). Element 10[c] is the apparatus equivalent of element 1[c].

9. Claim 11

645. As discussed above in Section X.B.8, Bjontegaard in view of the knowledge of a POSITA renders claim 10 obvious.

- a. **Element 11[a]: The user equipment according to claim 10, wherein the deblocking filter unit further comprises a first clipping unit configured to clip said first offset within an interval of $-t_c$ and t_c by setting a first offset to be equal to $-t_c$ if said first offset is smaller than $-t_c$ and by setting a first offset to be equal to t_c if said first offset is larger than t_c ,**

646. In my opinion, Bjontegaard discloses or suggests element 11[a] for the same reasons I discuss above in Section X.B.2.a (element 2[a]). Element 11[a] is the apparatus equivalent of element 2[a].

- b. **Element 11[b]: wherein t_c is a threshold value that is dependent on a quantization parameter value assigned to said block.**

647. In my opinion, Bjontegaard discloses or suggests element 11[b] for the same reasons I discuss above in Section X.B.2.b (element 2[b]). Element 11[b] is the apparatus equivalent of element 2[b].

10. Claim 12

648. As discussed above in Section X.B.8, Bjontegaard in view of the knowledge of a POSITA renders claim 10 obvious.

- a. **Element 12[a]: The user equipment according to claim 10, wherein the deblocking filter unit further comprises a second clipping unit configured to clip said modified pixel value p0' of said pixel closest to said block boundary in said line of pixels to be within an interval of zero and a defined maximum value by setting said modified pixel value p0' of said pixel closest to said block boundary in said line of pixels to be equal to zero if said modified pixel value p0' of said pixel closest to said block boundary in said line of pixels is smaller than zero and by setting said modified pixel value p0' of said pixel closest to said block boundary in said line of pixels to be equal to said defined maximum value if said modified pixel value of p0' of said pixel closest to said block boundary in said line of pixels is larger than said defined maximum value and**

649. In my opinion, Bjontegaard in view of a POSITA's knowledge discloses or suggests element 12[a] for the same reasons I discuss above in Section X.B.3.a (element 3[a]). Element 12[a] is the apparatus equivalent of element 3[a].

- b. Element 12[b]: to clip said modified pixel value q0' of said pixel closest to said block boundary in said corresponding line of pixels to be within an interval of zero and said defined maximum value by setting said modified pixel value q0' of said pixel closest to said block boundary in said corresponding line of pixels to be equal to zero if said modified pixel value q0' of said pixel closest to said block boundary in said corresponding line of pixels is smaller than zero and by setting said modified pixel value q0' of said pixel closest to said block boundary in said corresponding line of pixels to be equal to said defined maximum if said modified pixel value q0' of said pixel closest to said block boundary in said corresponding line of pixels is larger than said defined maximum value.**

650. In my opinion, Bjontegaard in view of a POSITA's knowledge discloses or suggests element 12[b] for the same reasons I discuss above in Section X.B.3.b (element 3[b]). Element 12[b] is the apparatus equivalent of element 3[b].

XI. SUMMARY AND ADDITIONAL REMARKS

651. It is my opinion that Grounds 1 and 2 each renders the Challenged Claims unpatentable as obvious, as I explained in the paragraphs preceding.

652. I currently hold the opinions expressed in this Declaration based on the information available to me at this time. I may, however, form additional or new opinions should additional information or insights be made available, or upon further consideration.