

FACIAL FEATURE BASED HEAD TRACKING AND POSE ESTIMATION

Jari Hannuksela

Hannuksela J. (2003) Facial feature based head tracking and pose estimation. Department of Electrical and Information Engineering, University of Oulu, Oulu, Finland. Diploma Thesis, 60 p.

ABSTRACT

Main purpose of this thesis is to study a facial feature based human head pose estimation. Especially the facial feature extraction methods suitable for a real-time system are considered. Based on the literature survey, an experimental head tracking and pose estimation system is implemented.

An efficient method based on skin detection, gray-scale morphology and a geometrical face model are applied to detect roughly the face region and extract facial features. The facial features extracted include the eyes, mouth, nostrils and eyebrows. The system automatically initializes whenever the face of a user is detected. After successful initialization, the tracker is started. The extended Kalman filtering (EKF) framework is utilized for tracking and estimation of the 3-D pose of the moving head in an image sequence. During tracking the correspondences between a rigid head model and three extracted facial features (eyes and mouth) are used to solve the pose. To demonstrate the practicality of the approach the user's gaze direction is detected and applied to a cursor control on the computer screen.

Experiments with real image sequences and synthetic data were carried out. Tests with real sequences showed that the system is able to extract facial features reliably from a near frontal view of the face under proper lighting conditions. Furthermore, high real-time performance was achieved. Pose estimation and gaze detection accuracies were tested with synthetic data and good preliminary results were obtained. The fast implementation proves the basic premise that the proposed approach can be applied in platforms like mobile devices where high computational resources are not available. However, there are some restrictions in the current implementation. A user with glasses, occlusion and shadows will cause problems for the system. In future work the observed problems will be considered more closely.

Keywords: proactive computing, machine vision, face detection, extended Kalman filtering.

Hannuksela J. (2003) Kasvonpiirteisiin perustuva pään seuranta sekä paikan ja orientaation estimointi. Oulun yliopisto, sähkö- ja tietotekniikan osasto. Diplomityö, 60 s.

TIIVISTELMÄ

Tämän diplomityön päätarkoituksena oli tutkia kasvonpiirteiden avulla suoritettavaa ihmisen pään paikan ja orientaation estimointia. Eri-tyisesti tarkasteltiin reaaliaikaisiin järjestelmiin sopivia kasvonpiirteiden irrotusmenetelmiä. Kirjallisuuskatsauksen pohjalta toteutettiin kokeellinen järjestelmä pään seurantaan sekä paikan ja orientaation estimointiin.

Tehokasta menetelmää, joka perustuu ihon ilmaisuun, harmaasävy-morfologiaan ja geometriseen kasvomalliin, sovellettiin kasvoalueen kar-keaan ilmaisuun ja kasvonpiirteiden irrottamiseen. Irrotettavia kasvon-piirteitä ovat silmät, suu, nenänreiät ja kulmakarvat. Aina, kun käyt-täjän kasvot ilmaistaan, järjestelmä alustuu. Seuranta käynnistyy on-nistuneen alustuksen jälkeen. Laajennettua Kalman suodatusta käy-tettiin liikkuvan pään seurantaan sekä 3-D paikan ja orientaation es-timointiin kuvasekvenssistä. Seurannan aikana paikka ja orientaatio ratkaistiin jäykän päämallin ja irrotettujen kolmen kasvonpiirteen (sil-mät ja suu) vastaavuuksien avulla. Lähestymistavan käytännöllisyyden demonstroimiseksi käyttäjän katseen suunta ilmaistiin ja sitä sovellet-tiin cursorin ohjaukseen tietokoneen näytöllä.

Kokeita suoritettiin todellisilla kuvasekvensseillä ja synteettisellä da-talla. Todellisilla kuvasekvensseillä suoritettut testit osoittivat, että järjestelmällä voidaan irrottaa kasvonpiirteet luotettavasti lähes kohti-suorasti edestäpäin otetusta kasvokuvasta, jos valaistusolosuhteet ovat sopivat. Menetelmän reaaliaikainen suorituskky todettiin korkeaksi. Paikan ja orientaation estimoinnin ja katseen ilmaisun tarkkuus tes-tattiin synteettisellä datalla. Tehtyjen kokeiden tulokset olivat hyviä. Nopea toteutus antaa viitteen siitä, että ehdotettua lähestymistapaa voidaan soveltaa esimerkiksi liikkuvissa laitteissa, joissa laskennalliset resurssit ovat rajalliset. Toteutetussa järjestelmässä on myös rajoit-teita. Silmälasit, piirteiden peittyminen ja varjot aiheuttavat järjestel-mälle ongelmia. Havaittuja ongelmia tullaan tarkastelemaan lähemmin jatkotutkimuksissa.

Avainsanat: proaktiivinen tietotekniikka, konenäkö, kasvojen ilmaisu, laajennettu Kalman suodatus.

CONTENTS

ABSTRACT
 TIIVISTELMÄ
 CONTENTS
 PREFACE

LIST OF SYMBOLS AND ABBREVIATIONS

1. INTRODUCTION	8
2. FACIAL FEATURE EXTRACTION	10
2.1. Background	10
2.2. Facial feature extraction methods	11
2.3. Low level methods	12
2.3.1. Color	12
2.3.2. Motion	13
2.3.3. Edge detection	13
2.3.4. Projections	14
2.3.5. Morphology	15
2.3.6. Symmetry transform	16
2.3.7. Geometrical face model	16
2.3.8. Gabor filters	17
2.4. Template matching methods	17
2.4.1. Correlation	17
2.4.2. Hough transform	18
2.4.3. Snakes	19
2.4.4. Deformable templates	19
2.4.5. Point distribution models	20
2.5. Discussion	20
3. HEAD POSE ESTIMATION AND TRACKING	22
3.1. Related work	22
3.2. Visual tracking	23
3.3. Pose estimation	25
3.3.1. Camera model	26
3.3.2. Perspective-three-point algorithm	27
3.3.3. Dynamic model	29
3.3.4. Measurement model	30
3.3.5. Extended Kalman filter algorithm	30
3.4. Tracking with Kalman filtering	31
4. IMPLEMENTATION	32
4.1. System initialization	33
4.1.1. Face candidate detection	33
4.1.2. Facial feature extraction	34
4.2. Head tracker	37
4.2.1. Tracker initialization	37
4.2.2. Measurements	39
4.3. Gaze detection	40
5. EXPERIMENTAL RESULTS	42
5.1. Tests with real image sequences	42
5.1.1. Initialization	42
5.1.2. Facial feature tracking	44

5.2. Tests with synthesized data	47
5.2.1. Pose estimation	47
5.2.2. Gaze detection	49
5.3. Real-time performance	51
6. DISCUSSION	53
7. SUMMARY	55
8. BIBLIOGRAPHY	56
9. APPENDICES	60

PREFACE

This thesis was carried out in the Machine Vision Group at the Department of Electrical and Information Engineering of the University of Oulu. The work was done in a project called "Machine vision for sensing and understanding human actions (ProVision)".

I am grateful to my supervisors Professor Janne Heikkilä and Professor Matti Pietikäinen for their guidance and advice throughout work on this thesis. I would like to thank the Academy of Finland for the financial support obtained for this work. I wish also to thank Mr. Gordon Roberts for the language revision of this manuscript. Finally, I would like to thank the personnel in the Machine Vision Group who have helped me in my work.

Oulu, Finland 27.10.2003

Jari Hannuksela

LIST OF SYMBOLS AND ABBREVIATIONS

D_u, D_v	conversion factors from metric units to pixels
f	focal length
$\mathbf{R}$	rotation matrix
$\mathbf{t}$	translation vector
u, v	image coordinates
$\mathbf{x}_k$	state vector at time instant k
Φ_k	state transition matrix at time instant k
σ	standard deviation
ω, φ, κ	Euler angles
2-D	two-dimensional
3-D	three-dimensional
ASM	active shape model
EKF	extended Kalman filter
GUI	graphical user interface
HCI	human-computer interaction
NCC	normalized color coordinates, normalized RGB, color model
P3P	perspective-three-point, pose estimation algorithm
PCA	principal component analysis
PDM	point distribution model
PTZ	pan-tilt-zoom
PUI	perceptual user interface
RGB	red green blue, color model
RMS	root mean square

3. HEAD POSE ESTIMATION AND TRACKING

The position and orientation (pose) of a human head can be estimated by extracting the facial feature correspondences between successive images. However, the 2-D image measurements of facial features are usually affected by significant noise and disturbances due to various error sources in the imaging. These errors may produce large pose estimation uncertainties. Visual tracking of the facial features in a video sequence with a proper estimation technique can improve the pose estimation result significantly.

In this chapter representative studies of the human head tracking are presented. The visual tracking is introduced in general. The focus is on human movement tracking. The pose estimation problem is also introduced. The needed camera model and a simple pose estimation algorithm for a single image are described. The extended Kalman filter (EKF) is used to estimate head pose in this thesis. The adoption of the EKF represents a good trade-off between computational complexity and estimation accuracy. Facial feature tracking with EKF is described in Section 3.4.

3.1. Related work

In this section, some representative work in the area of visual tracking of the human head is presented. According to Pentland [4], most of the current head tracking systems track facial features in 2-D and then use these points to estimate the 3-D pose of the head. Several systems also allow gaze detection.

Azarbayejani et al. [37, 38] have used recursive estimation to recover head structure and motion from image sequences of rigid motion. In their later research, the focal length is also recovered. Small facial features such as corners of the eyes and nostrils are tracked using the extended Kalman filter framework. The use of the system is applicable if the same points are visible in most of the frames.

Jebara and Pentland [39] presented a real-time system for detecting, modelling and tracking human faces. They used extended Kalman filter (EKF) to recover 3-D structure, pose, motion and focal length based on 2-D feature tracking. The system automatically detects faces and facial features. The face detection is based on skin color information. After face localization facial features including eyes, nose and mouth are detected using symmetry transform and an image intensity gradient. The face and facial feature detection is not performed in real time. After initialization the features are tracked by using EKF that recovers also 3-D structure and the pose. In the tracking stage features are measured using 2-D image correlation. If the feature is occluded due to a large amount of head movement the tracking fails.

Ström and Pentland [40] developed a head tracker based on the work of Jebara and Pentland [39]. The main change was in the feature points used in the tracking. The head tracker system tries to find the best features to track for each face instead of using predefined features; the eyes, nose and the mouth. The input image Hessians are used to rank features. The system is expected to be more robust against head movements. For example, if the eye is occluded a point from the ear is selected for tracking.

Black and Yacoob [41] developed a regularized optical flow method that uses an eight parameter 2-D model. However, the use of the 2-D planar model limits accurate tracking to medium size head motions. The large head motion causes problems. Based on the work of Black and Yacoob, Basu et al. [42] presented a system to track heads in different conditions such as a large amount of head motion. Instead of using a planar model they used a 3-D ellipsoidal model of the head. The algorithm starts by computing optical flow for the image and then it estimates 3-D motion of the rigid model. The estimated motion parameters are used to modify the location and rotation of the model and are used for the next frame. The motion estimation for each frame depends on the accuracy of the estimation in the previous frame and therefore the errors tend to propagate and grow. The system is not real-time due to slow computation of the optical flow.

Ballard and Stockman [43] proposed a system to estimate gaze direction. They assumed that distances between facial features are not changed with facial rotation and translation. The eyes and nose of the user are tracked using parameterized templates. Using these three points, the pose and gaze is computed with the P3P algorithm. A disadvantage is that the distance between the camera plane and features has to be measured manually in the initialization phase. Also the gaze direction is not computed in real-time.

Yang et al. [44] proposed a region based visual tracking system for face and facial feature tracking in real-time. They used a statistical skin color model for facial region detection and face tracking. After face location, the facial features such as eyes, nostrils and lips are extracted within the facial region. The facial features are tracked and the head pose is estimated based on a perspective camera model. They reported a frame rate of over 30 frames per second using a low-end workstation. A drawback is that the region-based trackers are not usually very accurate.

3.2. Visual tracking

The basic idea behind visual tracking is to determine the measurable state of a target object from a sequence of images [45]. The complexity of the tracking problem depends on the application. In some cases, the position of an object in image coordinates can be enough, e.g. computation of the object's center of mass. In other cases, different parts of the object should be tracked. For example visual tracking of human motion involves tracking of different body parts. Visual tracking can be understood as a process of repeated estimation and search steps. It should be noted that the estimation is performed in the state-space, not in the image. Typically, the parameters for the state are based on a structural model of the object. If the dynamics for the object can be modelled, the tracking can be based on motion estimation.

Human motion can be rigid, articulated, or deformable [46]. Rigid motion means that there is no relative motion between different parts of the object with respect to each other. In the case of articulated motion, the structure of the object can be characterized as comprising rigid components connected by simple constraints such as hinges, slides etc. These different types of motion can exist simultaneously, such as facial expressions during head rotations.

Visual tracking of rigid objects is typically based on searching feature corre-

spondences in the image sequence. Using feature correspondences the motion estimation is a quite straightforward task and the 3-D pose estimation is possible. The deformable objects are much more difficult to be tracked due to the motion caused by deformations of the object.

Many criteria can be used to classify visual tracking systems. Black et al. [46] categorized approaches to track humans as follows:

- motion-based vs. static
- 3-D model based vs. 2-D image-based
- region-based vs. boundary-based

The motion-based approaches track objects by using motion information between consecutive images. The methods to recover motion parameters can be further divided into feature correspondence based methods and optical flow methods. The former methods extract image features and track their movement from image to image. The features extracted from an image can be for example regions, edges, and point features. For head tracking, point features such as eyes, nostrils and mouth are widely used. Noisy measurements make feature based motion estimation difficult. Methods based on optical-flow use spatio-temporal derivatives to estimate image flow at each location in the image. The advantage of optical flow is that there is no need to model the movement of the object. Furthermore, if the correspondences are difficult to determine optical flow based methods may be appropriate. However, optical flow is typically hard to use in real time systems due to its high computational cost. The static approaches simply try to localize the object in a single image.

The 3-D model-based trackers assume a 3-D model of the object and estimate pose based on feature correspondences. A general model of the object is difficult to achieve due to variations in appearance. The point correspondences are extracted if the rigidity for object can be assumed. The 2-D image or appearance-based methods use intensity or color information on the images to track object parts. Mapping 2-D images of the object and corresponding poses can solve the 3-D pose. The mapping is based on learning from appearance examples. The 2-D image based tracking performance is decreased by the articulation and deformation [46].

Region-based or blob trackers typically use information such as color to segment object of interest from the image. There are two groups of region-based methods: view-based and parametric. The view-based method finds the best match of a region in a search area with a reference template. The parametric method assumes a parametric model of changes in an image and computes an optimal fitting of the model to the pixel data in a region. Region based tracking can be robust but accurate results are difficult to achieve. Moreover, there may be multiple regions. The boundary-based approaches use features based on discontinuities in the image such as edges and contours. Tracking and describing arbitrarily curving edges and contours can be a difficult task due to changes in imaging conditions, certain motions and occlusion.

3.3. Pose estimation

The process of finding the position and orientation (pose) of an object is known as the pose estimation problem. 3-D pose estimation from 2-D images can be solved in many ways. The classical approach is stereovision, where two cameras with two different views are used to capture 2-D images of the given 3-D object. Both cameras observe the object points. Based on the disparities of the point locations in the 2-D images the 3-D pose of the object can be estimated. Another approach is to use only one stationary camera and a sequence of monocular 2-D images of moving objects [47]. Equivalently the camera can be moving and the object is stationary. Another possibility is that both can be moving. A 2-D image sequence is then used to compute disparities. The 3-D pose of the object can be estimated also from a single image if some constraints on the imaging geometry are set. All of these methods need correspondence pairs in order to estimate the pose. In this study, the correspondences are computed between facial features extracted and a model of the human head.

In order to extract the pose of the head, the relation between a 3-D object coordinate system and 2-D image coordinates must be known. This relation can be modelled using the pinhole camera model described in Section 3.3.1. A simple pose estimation procedure called the perspective-three-point (P3P) algorithm is presented in Section 3.3.2. The advantage of the P3P algorithm is that the 3-D pose of an object can be computed from three points if the geometry of the object is known and only one image is needed. It is therefore very suitable for initial pose estimation in the starting phase of the tracking.

Visual tracking of the object throughout a long video sequence will improve the pose estimation accuracy if the information from the previous frames is used. The tracking and pose estimation of a known object in 3-D space using the 2-D feature correspondences from images can be formulated as a nonlinear estimation problem due to the pinhole camera model. This formulation allows the use of nonlinear estimation techniques, such as extended Kalman filtering (EKF) and the Condensation [48] algorithm. Other methods for tracking include optical flow based methods. One popular technique used in such methods is called the KLT (Kanade-Lucas-Tomasi) tracking algorithm proposed by Lucas and Kanade [49].

Kalman filtering [50] is a widely applied technique for pose estimation and tracking in computer vision. It has for example been applied to 3-D motion estimation of a known rigid object and the structure estimation of an unknown scene [51]. The main features of the Kalman filter formulation and solution are modelling the random process under consideration using a state-space model and recursive processing of the noisy measurement data. It is an optimal filter because it generates the best linear mean squared solution. The corresponding mean-square error estimate is also provided. The recursive nature of the algorithm makes it convenient to use in real-time systems where the data can be integrated into the state estimate and there is no need to store previous measurements. Furthermore, it is based on physical dynamics which allow for predictive estimation.

The measurement equation is usually non-linear in computer vision systems because a perspective camera model is used. For non-linear systems a sub-optimal variant of the filter, the extended Kalman filter (EKF) is typically used. The EKF requires a physical dynamic model of the motion and the measurement model to

relate the 3-D pose parameters to the 2-D image feature positions. The dynamic (Section 3.3.3) and measurement models (Section 3.3.4) used are given for the head pose estimation problem. The EKF algorithm is presented in Section 3.3.5.

3.3.1. Camera model

The image formation process can be modelled via perspective projection. The pinhole camera model uses perspective projection to relate the object point in the 3-D scene to the corresponding observations in the 2-D image, as shown in Fig. 6.

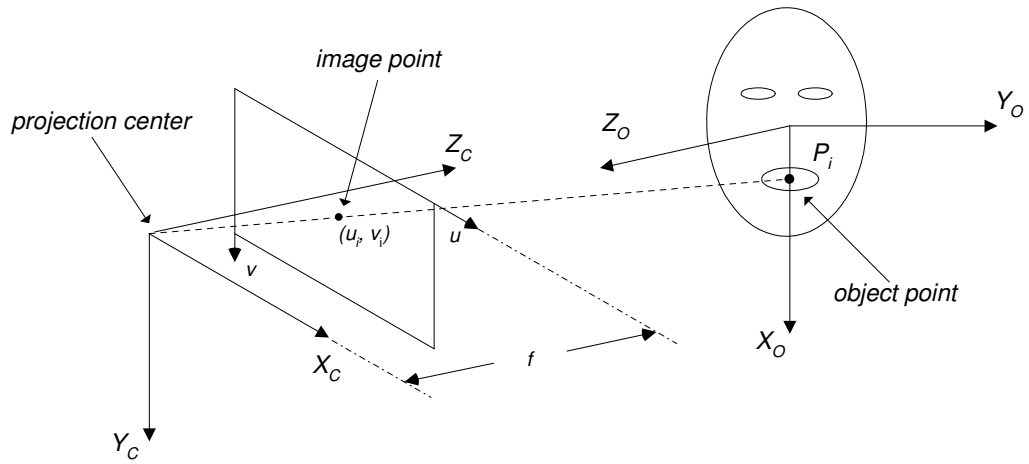


Figure 6. Pinhole camera model.

The origin of the camera coordinate system is at the projection center and the Z_c -axis coincides with the optical axis. The image plane is parallel to the (X_c, Y_c) plane and the distance from the projection center is f . The transformation from the 3-D object coordinates $\mathbf{P}_i$ to 2-D image observations (u_i, v_i) can be decomposed into the following steps. First the object coordinates $\mathbf{P}_i$ are mapped to camera coordinates $\mathbf{p}_i$ as follows

$$\mathbf{p}_i = \mathbf{R}\mathbf{P}_i + \mathbf{t}, \quad (8)$$

where $\mathbf{R}$ is an orthonormal rotation matrix and $\mathbf{t}$ is a translation vector. The rotation matrix describes the orientation of the camera coordinate system related to the object coordinate system. The translation vector describes the position of the origin of the object coordinate system in the camera coordinate system.

The rotations are defined by Euler angles: ω rotating around the X_c -axis, φ rotating around the Y_c -axis and κ rotating around the Z_c -axis. The rotations are performed counter clockwise. The rotation matrices can be concatenated into a single matrix

$$\mathbf{R} = \mathbf{R}_\kappa \mathbf{R}_\varphi \mathbf{R}_\omega, \quad (9)$$

where

$$\mathbf{R}_\omega = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\omega & -\sin\omega \\ 0 & \sin\omega & \cos\omega \end{bmatrix} \mathbf{R}_\varphi = \begin{bmatrix} \cos\varphi & 0 & \sin\varphi \\ 0 & 1 & 0 \\ -\sin\varphi & 0 & \cos\varphi \end{bmatrix} \mathbf{R}_\kappa = \begin{bmatrix} \cos\kappa & -\sin\kappa & 0 \\ \sin\kappa & \cos\kappa & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

The following convention can be established

$$\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}.$$

Thus, the homogenous transformation from the object coordinates to the camera coordinates can be represented by

$$\begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}. \quad (10)$$

The camera coordinates $\mathbf{p}_i(x_i, y_i, z_i)$ are projected into the image plane (u_i, v_i)

$$\begin{bmatrix} u_i \\ v_i \end{bmatrix} = \frac{f}{z_i} \begin{bmatrix} x_i \\ y_i \end{bmatrix}, \quad (11)$$

where f is the effective focal length.

The image coordinates (u_i, v_i) are transformed to computer image coordinates (u'_i, v'_i) in pixels as follows

$$\begin{bmatrix} u'_i \\ v'_i \end{bmatrix} = \begin{bmatrix} s_u D_u u_i \\ D_v v_i \end{bmatrix} + \begin{bmatrix} u'_0 \\ v'_0 \end{bmatrix}, \quad (12)$$

where s_u is the scale factor, which adjusts the image aspect ratio and (u'_0, v'_0) is the image center in pixels. The conversion factors D_u and D_v are used to convert metric units into pixels. The origin of the image coordinate system is in the upper left corner of the image array.

3.3.2. Perspective-three-point algorithm

The pose of the object can be computed from the 3-D coordinates of three feature points seen in a 2-D image if the three 3-D interpoint distances are known. This is referred to as the perspective-three-point (P3P) problem [52]. It is assumed that the object is rigid and the known distances do not vary as the object moves.

The imaging model used is the perspective camera model, as shown in Fig. 7. It is assumed that the world coordinate system is the same as the camera coordinate system. The focal length f is the distance from the center of projection to the image plane. The coordinates of image points are $\mathbf{Q}_i = [u_i, v_i, f]$ relative to the camera coordinate system. The locations of the unknown feature points $\mathbf{P}_i$ in the 3-D scene can be solved from the corresponding observations $\mathbf{Q}_i$ in the image

plane (u, v) if the distances a_i between $\mathbf{Q}_i$ and $\mathbf{P}_i$ can be computed from the following equations

$$\begin{cases} a_1^2 + a_2^2 - 2(\mathbf{q}_1, \mathbf{q}_2)a_1a_2 = d_{12}^2 \\ a_2^2 + a_3^2 - 2(\mathbf{q}_2, \mathbf{q}_3)a_2a_3 = d_{23}^2 \\ a_3^2 + a_1^2 - 2(\mathbf{q}_3, \mathbf{q}_1)a_3a_1 = d_{31}^2 \end{cases}, \quad (13)$$

where the d_{mn} is the known distance between object points P_m and P_n . The $\mathbf{q}_i$ is the unit vector in the direction of $\mathbf{P}_i$ and $\mathbf{Q}_i$. It is defined as follows

$$\mathbf{q}_i = \frac{1}{\sqrt{x_i^2 + y_i^2 + f^2}} [x_i, y_i, f]^T. \quad (14)$$

The system of equations (13) can be solved by finding the roots of the following functions

$$f(a_m, a_n) = a_m^2 + a_n^2 - 2(\mathbf{q}_m, \mathbf{q}_n)a_ma_n - d_{mn}^2, \quad (15)$$

which represents the difference between the known and computed distance for a pair of feature points. The algorithm to solve this problem is presented by Shapiro and Stockman [53].

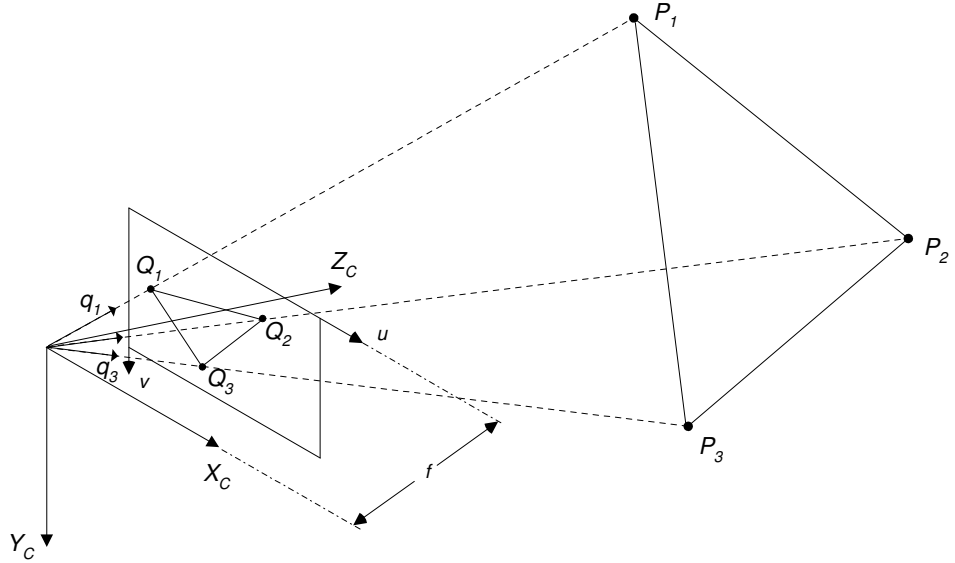


Figure 7. Geometry for the perspective 3 point problem.

After the solution has been found, the position of the 3-D feature points can be computed from $\mathbf{p}_i = a_i\mathbf{q}_i$. These 3-D points can be used further to estimate the pose of the object. For example, the obtained three points could be three facial features; the eyes and mouth. It can be assumed that the facial features are located in the same plane and the origin of the object coordinate system is at the center of gravity of these features. In this case, the pose of the object can be solved by using simple vector calculations.

3.3.3. Dynamic model

A dynamic model allows prediction and filtering of the object pose and motion from a sequence of images. In this thesis, the term pose estimation is used even when the motion is also estimated. For pose estimation, several dynamic models can be applied. Constant velocity or constant acceleration can be assumed or more accurate mechanics can be considered. In this thesis, the dynamic model used is a discrete time state-space model. The head is treated as a rigid object that moves with constant velocity and the pose of the head changes in respect to the camera. Accelerations are modelled as the process noise. The motion model is

$$\mathbf{x}_{k+1} = \Phi_k \mathbf{x}_k + \Gamma_k \varepsilon_k, \quad (16)$$

where the parameters to be estimated are presented by the state vector $\mathbf{x}_k$ at time instant k , and Φ_k is the state transition matrix. The state transition matrix relates the state at time instant k to the state at time instant $k + 1$.

The state vector $\mathbf{x}_k$ of the 3-D pose estimation problem consists of the position (x_k, y_k, z_k) and orientation $(\omega_k, \varphi_k, \kappa_k)$ of the head and the corresponding velocities at time instant k . It is defined as follows

$$\mathbf{x}_k = [x_k, \dot{x}_k, y_k, \dot{y}_k, z_k, \dot{z}_k, \omega_k, \dot{\omega}_k, \varphi_k, \dot{\varphi}_k, \kappa_k, \dot{\kappa}_k].$$

The state transition matrix Φ_k is given by

$$\Phi_k = \begin{bmatrix} 1 & \Delta t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \Delta t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

where Δt is the time difference between two successive images.

$\Gamma_k \varepsilon_k$ models the uncertainty of the motion model. The process noise ε_k is assumed to be Gaussian distributed with an expected value $E\{\varepsilon_k\} = 0$ and the covariance matrix $\mathbf{Q}_k = E\{\varepsilon_k \varepsilon_k^T\}$. A process noise transition matrix Γ_k is defined

as follows

$$\mathbf{\Gamma}_k = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

3.3.4. Measurement model

The measurement model is needed to relate the state vector to the 2-D observations in the image and it is

$$\mathbf{z}_{k+1} = F(\mathbf{p}_i, \mathbf{x}_k) + \eta_k, \quad (17)$$

where $F(\mathbf{p}_i, \mathbf{x}_k)$ is a measurement function that uses the camera model described in Section 3.3.1 to transform object coordinates $\mathbf{p}_i$ to image coordinates $\mathbf{z}_{k+1}$ using state $\mathbf{x}_k$.

The measurement noise η_k models uncertainty in the measurements and it is assumed to be Gaussian distributed with an expected value $E\{\eta_k\} = 0$ and the covariance matrix $\mathbf{R}_k = E\{\eta_k \eta_k^T\}$.

It was mentioned in Section 3.3.1 that the measurement function $F(\mathbf{p}_i, \mathbf{x}_k)$ is nonlinear and therefore it should be linearized. The first order approximation is used to linearize the measurement function around the current state estimate. The partial derivatives of the measurements with respect to the state variables are used to obtain Jacobian matrix $\mathbf{J}_k$ that is the linearized measurement transformation matrix. It is defined as follows

$$\mathbf{J}_k = \left. \frac{\partial F}{\partial \mathbf{x}_k} \right|_{\mathbf{x}_k = \mathbf{x}_{k+1|k}}. \quad (18)$$

3.3.5. Extended Kalman filter algorithm

The EKF algorithm estimates the state using two stages: prediction (time update) Eqs. 19-21 and correction (measurement update) Eqs. 22-24. In the first stage, the state of the system at the next time instant is predicted based on the previous state and the dynamic model of the system. In the correction stage the predicted state is corrected by using measurements. Repeating the stages iteratively performs the filtering cycle. Using the state from the previous cycle always starts the next filtering cycle.

The steps for the EKF algorithm are given as follows:

1. *A priori* estimate of the state vector $\hat{\mathbf{x}}_{k+1|k}$ is obtained by multiplying the previous *a posteriori* state estimate $\hat{\mathbf{x}}_{k|k}$ by the state transition matrix Φ_k .

$$\hat{\mathbf{x}}_{k+1|k} = \Phi_k \hat{\mathbf{x}}_{k|k} \quad (19)$$

2. *A priori* estimation covariance matrix $\mathbf{P}_{k+1|k}$ is obtained using Equation 20. $\mathbf{P}_{k+1|k}$ represents the uncertainty of the state estimate. $\mathbf{P}_{k|k}$ is the previous *a posteriori* estimation covariance matrix.

$$\mathbf{P}_{k+1|k} = \Phi_k \mathbf{P}_{k|k} \Phi_k^T + \Gamma_k \mathbf{Q} \Gamma_k^T \quad (20)$$

3. The Kalman gain matrix $\mathbf{K}_k$ is updated.

$$\mathbf{K}_k = \mathbf{P}_{k+1|k} \mathbf{J}_k^T (\mathbf{R}_k + \mathbf{J}_k \mathbf{P}_{k+1|k} \mathbf{J}_k^T)^{-1} \quad (21)$$

4. The prediction for the measurement in the image coordinates $\mathbf{z}_{k+1|k}$ is obtained using measurement function $F(\mathbf{p}_i, \mathbf{x}_{k+1|k})$.

$$\mathbf{z}_{k+1|k} = F(\mathbf{p}_i, \mathbf{x}_{k+1|k}) \quad (22)$$

5. The state estimate $\hat{\mathbf{x}}_{k+1|k+1}$ is corrected. The innovation term $(\mathbf{z}_k - \mathbf{z}_{k+1|k})$ is weighted by the Kalman gain matrix $\mathbf{K}_k$. $\mathbf{z}_k$ is the actual measurement and the $\mathbf{z}_{k+1|k}$ is the predicted measurement.

$$\hat{\mathbf{x}}_{k+1|k+1} = \hat{\mathbf{x}}_{k+1|k} + \mathbf{K}_k (\mathbf{z}_k - \mathbf{z}_{k+1|k}) \quad (23)$$

6. The estimation covariance matrix $\mathbf{P}_{k+1|k+1}$ is updated.

$$\mathbf{P}_{k+1|k+1} = (\mathbf{I} - \mathbf{K}_k \mathbf{J}_k) \mathbf{P}_{k+1|k} \quad (24)$$

3.4. Tracking with Kalman filtering

The features of interest can be efficiently tracked with EKF. Based on the previous state and the dynamic model, the location of the feature point in the next frame is predicted using the EKF algorithm. The measurement can be performed in a small region around the expected location of the feature [47]. To speed up the image processing algorithms, the search area for measurement is adjusted dynamically according to the uncertainty of the prediction. The adjustment can significantly reduce computational time.

The size for the search region can be defined from the estimation covariance matrix $\mathbf{P}_{k|k}$. The uncertainty is projected to the image coordinates using the Jacobian matrix $\mathbf{J}_k$ of the measurement function. The uncertainty covariance matrix $\mathbf{S}_{k+1|k}$ is given below

$$\mathbf{S}_{k+1|k} = \mathbf{J}_k \mathbf{P}_{k+1|k} \mathbf{J}_k^T + \mathbf{R}_k, \quad (25)$$

where $\mathbf{P}_{k+1|k}$ is *a priori* estimation covariance matrix and $\mathbf{R}_k$ is the measurement noise matrix. The diagonal elements of the matrix $\mathbf{S}_{k+1|k}$ represent the variance of uncertainty in the image coordinates. The search window is enlarged based on these variances.

4. IMPLEMENTATION

A system to estimate head pose in real-time is proposed in this chapter. The approach is mainly based on the literature review that was presented in the previous chapters. The overview of the system for head pose estimation is depicted in Fig. 8. The system consists of system initialization and head tracker stages. In the first stage, initialization is performed automatically whenever the human face is detected. For proper face candidate detection, a near-frontal view is needed. Within the possible face regions the facial feature candidates are extracted. These features are analyzed to establish the best constellation that forms a face. If the facial features are found, the initialization stage is performed.

After successful initialization, the head tracker stage is started. The tracker is initialized by using information extracted in the system initialization stage. The head pose is initialized by the simple pose estimation algorithm. During tracking, the extended Kalman filter estimates the head pose. The pose is estimated with three facial features including the eyes and the mouth. These features are tracked in a sequence of images. The predictive estimation also allows search region adjustment for the facial features in the tracking stage that decreases computational load. After the head pose is estimated, the gaze is detected in the monitor screen and used for cursor control. If the tracking fails, the initialization stage is started again. In the following sections, the different blocks of the system are introduced in more detail.

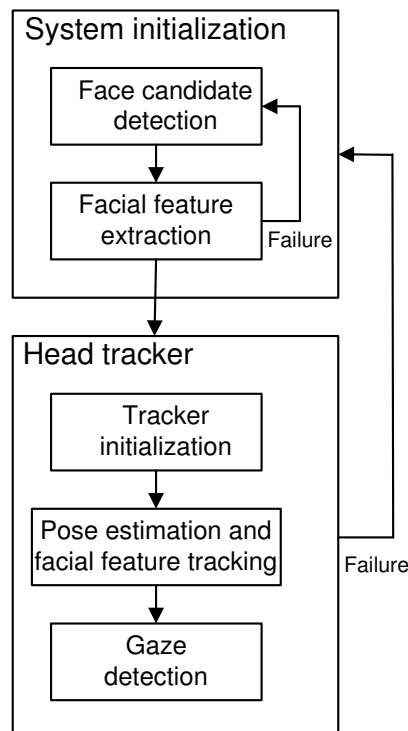


Figure 8. Block diagram of the system.

4.1. System initialization

The system is initialized in two steps. First, the face candidates are detected. Face-like regions are found by skin color analysis and then the shapes of the candidates are verified. Next, the facial feature candidates are searched for within the face candidate regions. After that, the feature candidate constellations are evaluated to find facial features and to initialize the head tracker.

4.1.1. Face candidate detection

In order to detect possible face areas in an image, the skin is detected in the image. Skin detection has been very popular in real-time applications due to its low computational cost as discussed in Section 2.5. In this thesis, skin detection is based on the skin locus model introduced in Section 2.3.1 [18]. First, the input image presented in RGB color space is converted to NCC color coordinates. Then, the skin color analysis for the image is performed. In the present implementation, only the red and blue color coordinates are used in skin locus. If the color coordinates of a pixel fall into the area of the skin locus, the pixel belongs to skin. After skin detection, the result is enhanced using a morphological closing operator with a mask size of 3x3 pixels. Fig. 9 shows an example result. The original color image is shown on the left and on the right is the result.

After the skin-like regions have been detected, these regions are possible face candidates. However, not all detected skin-regions contain a face. Some regions correspond to skin areas like hands and others to exposed parts of the body. Also, there might be false detections to objects with skin-like color. To distinguish faces and other skin-like areas, the shapes of the regions are verified next. A connected component analysis is performed, and only large connected components of the skin color are kept. Isolated pixels corresponding to the background are removed. Also, the bounding box for the skin area should be over 10% of the image size. The shape of the bounding box is also checked.

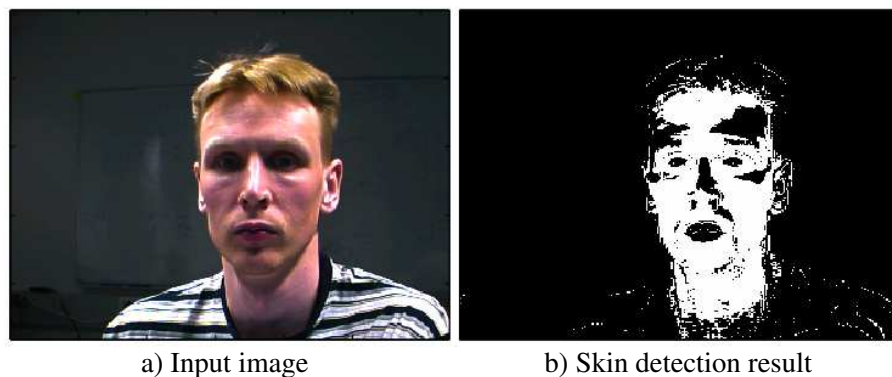


Figure 9. An example result for skin detection.

4.1.2. Facial feature extraction

Once a possible face is detected, the feature candidates are searched for within the face candidate area. Based on the observation that facial feature areas are usually darker than other face area, a morphological valley detector is used. The morphological operations are widely used in facial feature extraction and are highly suitable for real-time implementation as noted in Section 2.5. The algorithm used is based on the work of Wong et al. [25] and it is described in Section 2.3.5. In Fig. 10 an example result for valley detection within a face candidate region is shown. After detection, the morphological closing operation with a mask size of 3x3 pixels is utilized. Then, the connected component analysis is performed and centers of mass of the connected components are used for coordinates of facial feature candidates. An example result of facial feature candidate detection is shown on the left in Fig. 12.

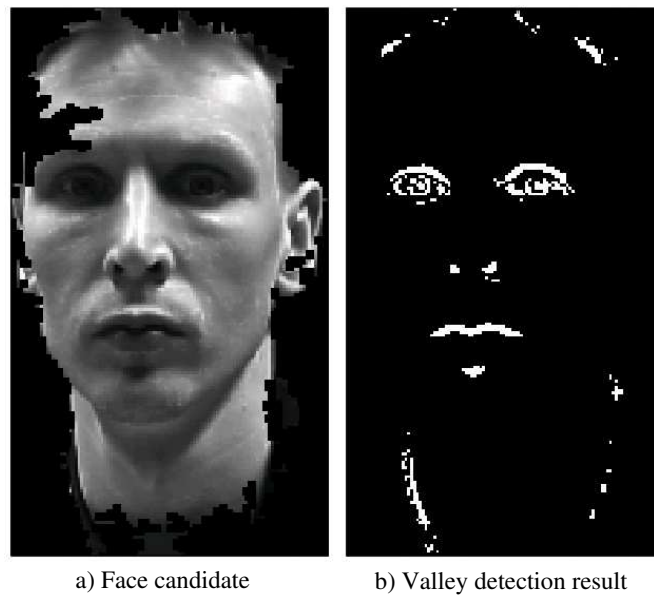


Figure 10. An example result for valley detection.

After the possible facial features are detected, the method proposed by Jeng et al. [16] is applied to establish the best feature constellation that forms a face. The method is based on a geometrical face model, which is shown in Fig. 11. The model includes facial features like eyes, eyebrows, nostrils and mouth.

The line passing through the centers of the eyes is called a base line. The coefficients for the base line $ax + by + c = 0$ and the angle θ between the base line and x-axis are calculated with following equations

$$\begin{cases} a = y_2 - y_1 \\ b = x_1 - x_2 \\ c = x_2 y_1 - x_1 y_2 \\ \theta = \arctan(-\frac{a}{b}), -\frac{\pi}{2} \leq \theta < \frac{\pi}{2} \end{cases},$$

where (x_1, y_1) are coordinates for the left eye and (x_2, y_2) are coordinates for the right eye.

The distances d_{mouth} , $d_{nostril}$ and $d_{eyebrow}$ indicate the distance from the mouth, the nostril and the eyebrow to the base line, correspondingly. The c_i $i = 1, 2, 3$ represent the boundary lines $bx - ay - c_i = 0$ for the facial feature search regions.

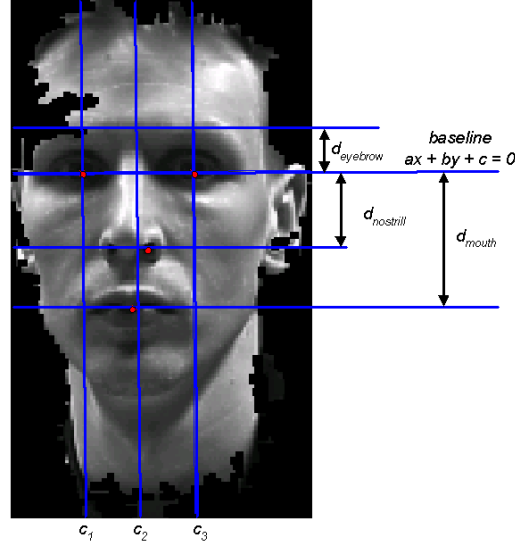


Figure 11. Geometry of the face model.

The best constellation search starts by constructing the possible eye pairs from the feature candidates detected. The feature candidate is a possible eye if the features are located at the upper half of the face candidate area. Two potential eye candidates are a possible eye pair if the distance D between them is under 80% of the width of the bounding box surrounding the face candidate. Also, the absolute value of the eye pairs base line angle θ should be under 30 degrees. These conditions reduce significantly the amount of eye candidate pairs. The distance D between a possible eye pair is used as a reference distance in the geometrical face model and it is calculated as follows

$$D = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}. \quad (26)$$

Based on the reference distance D , the reference distances from the mouth, nostrils and eyebrows to the base line in the geometrical model are defined. The $D_{nostril}$, $D_{eyebrow}$ and D_{mouth} are respectively $0.7D$, $0.35D$, $1.1D$. The distances of the model are constructed from measurements with several real faces from the database.

For each eye pair, the other facial features are searched for from on the geometrical face model. For example the possible mouth candidates for an eye pair candidate are searched for within the small region that is located in the distance $1.1D$ from the base line. If that region contains a feature candidate, it is treated as a mouth candidate for this possible eye pair. The distance from the base line is calculated for the mouth candidate as follows

$$d_{mouth} = \frac{|ax_k + by_k + c|}{\sqrt{a^2 + b^2}}, \quad (27)$$

where x_k and y_k are the image coordinates for the feature candidate. Because the center of the mouth must be located between lines $bx - ay - c_1 = 0$ and

$bx - ay - c_3 = 0$, the condition $(bx - ay - c_1)(bx - ay - c_3) < 0$ is required. If the mouth candidate is located outside these boundaries the feature is rejected.

Similarly the candidates for the other facial features like nostrils and eyebrows are detected and the distances calculated. However, there might be many candidates for certain features and every candidate found increases the amount of the feature constellations. In order to evaluate the candidates, the evaluation is performed using a special evaluation function for each facial feature. First the evaluation value E_{eye} for the eye pair is calculated by using the equation below

$$E_{eye} = \exp(-10(\frac{D - 0.4BB_{width}}{D})^2), \quad (28)$$

where D is the measured reference distance between eye candidates. BB_{width} is the width of the bounding box surrounding the possible facial region. The constant value ten is used to scale the result.

Then the following evaluation function is employed for the mouth candidates

$$E_{mouth} = \exp(-10(\frac{d_{mouth} - D_{mouth}}{D})^2), \quad (29)$$

where d_{mouth} is the measured distance from the baseline to the mouth and D_{mouth} is the corresponding reference distance in the geometrical face model.

The evaluation functions for eyebrows and nostrils are computed in a similar way, given in Eqs. 30-33.

$$E_{lefteyebrow} = \exp(-10(\frac{d_{lefteyebrow} - D_{lefteyebrow}}{D})^2) \quad (30)$$

$$E_{righteyebrow} = \exp(-10(\frac{d_{righteyebrow} - D_{righteyebrow}}{D})^2) \quad (31)$$

$$E_{leftnostril} = \exp(-10(\frac{d_{leftnostril} - D_{leftnostril}}{D})^2) \quad (32)$$

$$E_{rightnostril} = \exp(-10(\frac{d_{rightnostril} - D_{rightnostril}}{D})^2) \quad (33)$$

After the evaluation is performed for each eye pair candidate and for the other facial features, the total evaluation value is computed for the constellation. The total evaluation value is a weighted sum of the evaluation value for each facial feature. The weight is based on the importance of a given feature for face detection. The eyes are considered the most important features. The constellation which has largest evaluation value, is a face. However, a there is condition that the total evaluation value should be over the limit of 0.6. The total evaluation value is

$$E = 0.4E_{eye} + 0.3E_{mouth} + 0.1E_{leftnostril} + 0.1E_{rightnostril} + 0.05E_{lefteyebrow} + 0.05E_{righteyebrow}. \quad (34)$$

Fig. 12 shows an example result for the best constellation search. On the left there are candidates for features and on the right detected facial features. Also the search areas are marked on the image. These areas are also used in the tracking phase.

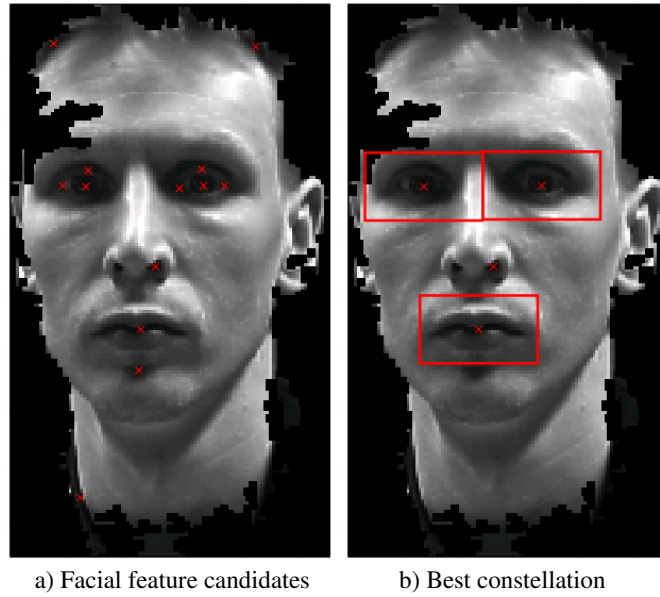


Figure 12. An example result for facial features extracted.

4.2. Head tracker

The EKF algorithm described in Section 3.3.5 is applied to head pose estimation in a video sequence. The EKF estimates the pose of a human head and predicts facial feature locations in the next image. After actual measurements, the pose estimate of the head is corrected. Since the estimates from previous images are taken into account a smooth result is obtained if the tracker works properly. To achieve proper operation the initialization and the measurements should be done carefully. In the following sections, implementation issues for the tracker are described.

4.2.1. Tracker initialization

In order to track a rigid object, the model of the object is needed. The proposed head model is shown in Fig. 13. Three facial features including the eyes and the mouth are included in the model. The center of the head coordinate system is set to the center of gravity of these three facial feature points. The coordinates for the mouth $\mathbf{P}_1$, the left eye $\mathbf{P}_2$ and the right eye $\mathbf{P}_3$ in the model are defined as follows

$$\mathbf{P}_1 = \begin{bmatrix} 1s & 0 & 0 \end{bmatrix}^T$$

$$\mathbf{P}_2 = \begin{bmatrix} -0.5s & -0.75s\alpha & 0 \end{bmatrix}^T$$

$$\mathbf{P}_3 = \begin{bmatrix} -0.5s & 0.75s\alpha & 0 \end{bmatrix}^T$$

where the α is the distance between eyes divided by the distance from the baseline to the mouth center. The distances are measured in the initialization phase. The s is the scaling factor that is used to scale model dimensions into the dimensions of the real face.

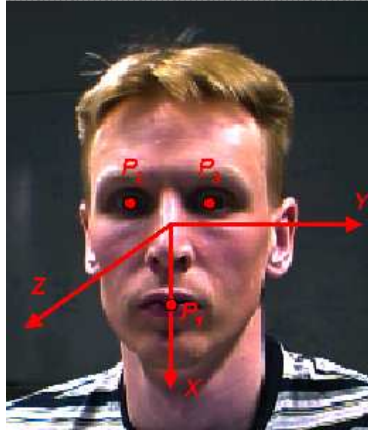


Figure 13. Head model used in tracking.

The pose parameters of the state vector are initialized by using the simple pose estimation algorithm described in Section 3.3.2. At the beginning, the velocities are set to zero. The time difference between two successive images is set to 1. This is needed in the state transition matrix Φ_k .

The estimation covariance matrix $\mathbf{P}_{k|k}$ needed in Eq. 20 should be initialized before tracking. The initial values for the matrix will have a critical impact on the starting phase of the tracking. In this implementation, large values along the main diagonal of the matrix are set. The initial estimation covariance matrix $\mathbf{P}_{0|0}$ is defined as follows

$$\mathbf{P}_{0|0} = \text{diag}(\sigma_{x_k}^2),$$

where the elements $\sigma_{x_k}^2$ of the diagonal matrix represent uncertainties for the corresponding parameters in the state vector.

The process noise matrix $\mathbf{Q}_k$ and measurement noise covariance matrix $\mathbf{R}_k$ should also be initialized based on assumed uncertainty in motion and measurements. The values for these matrices have a critical impact on the tracking and should be set as realistically as possible. The uncertainty in motion can be approximated from assumed maximum accelerations. In this implementation these matrices are not updated during tracking and are defined as follows

$$\mathbf{Q}_k = \text{diag}(\sigma_i^2),$$

where the elements σ_i^2 of the diagonal matrix represent uncertainty for the velocities, and

$$\mathbf{R}_k = \text{diag}(\sigma_q^2),$$

where the elements σ_q^2 of the diagonal matrix represent uncertainty for measurements in image pixels.

For this system, it is assumed that the noise associated with the image measurement in both x- and y-axis directions has a standard deviation of 2 pixels giving it a variance of 4.

4.2.2. Measurements

The three facial features are tracked including the eyes and the mouth. The features are tracked using small search windows around a corresponding feature in the image. The position of the window is based on the prediction of the EKF algorithm. The size of the window is based on the distances between facial features detected in the initialization stage. The window width is half of the distance between the eyes and the window height is one fourth of the distance between the mouth and the baseline. The size is adjusted on the basis of the uncertainty of the prediction derived from the covariance matrix, as described in Section 3.4. Using gray-scale morphological operators similarly to the initialization stage performs the measurement within a window. Sometimes there are several connected areas within a search window. Then, the center of mass of the area nearest to the prediction is chosen. The obtained feature points are then used to correct the predicted state (Eq. 23). In Fig. 14 is shown an example measurement for each feature in the tracking stage.

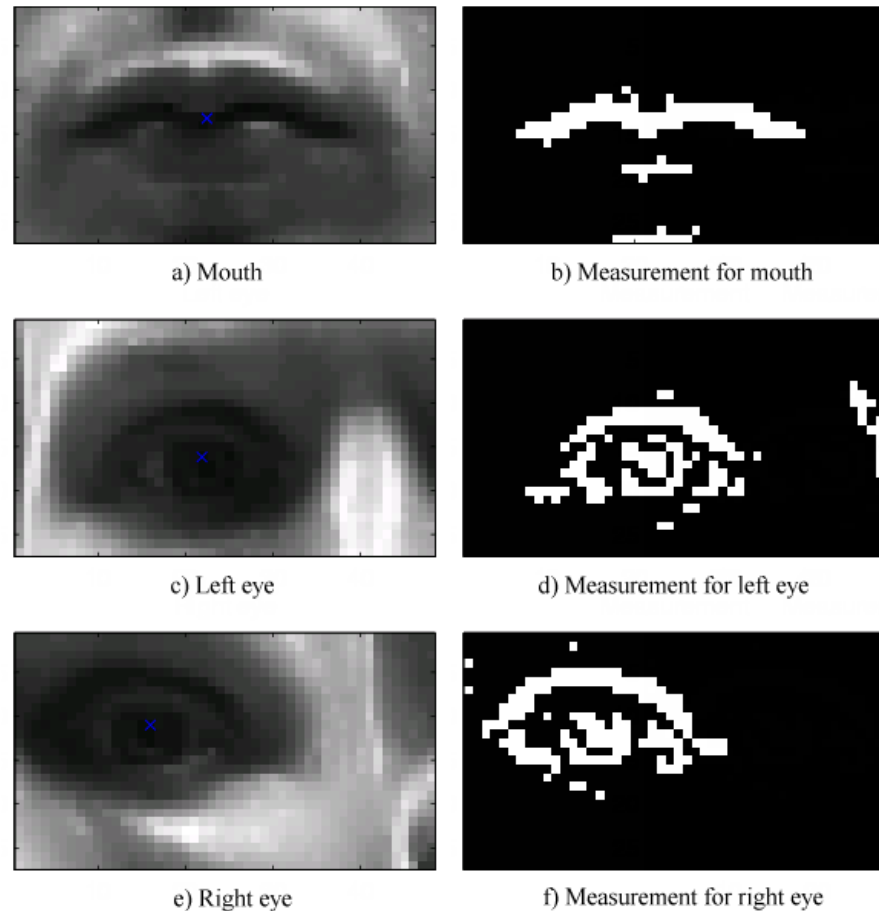


Figure 14. Example measurements for each facial feature in tracking phase.

The result of the feature measurements can occasionally cause errors that make the pose estimation fail and the tracking may become unreliable. To overcome this problem some constraints are set. The position of the measurement for a particular feature should be within the uncertainty limits derived from the prediction. These uncertainty limits can be extracted projecting state uncertainty to the image plane as described in Section 3.4. If the feature is not within the limits then the prediction is used instead of measurement. The error cannot exist for a long time and a limit is set to three frames. The feature can be also lost due to the occlusion. In this case the initialization is started again.

4.3. Gaze detection

Gaze detection can be used to locate the position on a monitor screen where a user is looking. Gaze direction is determined by orientation of the head and the orientation of the eyes. The eye orientation determines the exact gaze direction and the head orientation determines the overall gaze direction [44]. In this thesis the direction of the gaze is a normal of the plane defined by three facial features; the centers of the eyes and the mouth. This provides a rough estimate for the gaze direction. The points used in gaze detection are corrected measurements derived from the EKF algorithm.

Fig. 15 shows the geometry for the gaze detection problem. The direction of the gaze is defined by using the direction of the gaze vector $\mathbf{N}_H = [0, 0, 1, 0]^T$. The gaze vector starts from the origin $\mathbf{O}_H = [0, 0, 0, 1]^T$ of the head coordinate system. The gaze position in the monitor screen is described by point $\mathbf{P}_M$.

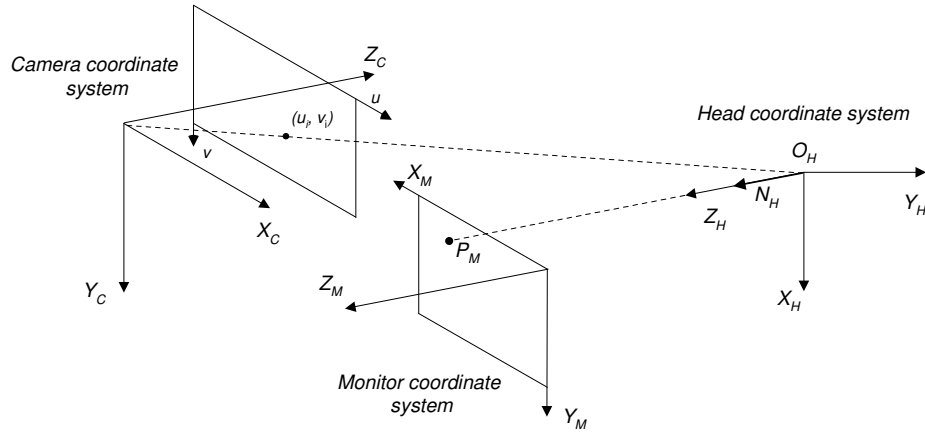


Figure 15. Geometry for the gaze detection problem.

In order to solve the gaze position in the monitor screen, the gaze vector has to be transformed from the head coordinate system to the monitor coordinate system. First, the origin $\mathbf{O}_H$ and the direction $\mathbf{N}_H$ of the gaze vector are transformed to the camera coordinate system using Eq. 10. After that, transformation to monitor coordinates is performed. It is assumed that the position and the ori-

entation are known in advance. The transformation is defined as follows

$$\begin{bmatrix} X_M \\ Y_M \\ Z_M \\ 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & -1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_C \\ Y_C \\ Z_C \\ 1 \end{bmatrix}, \quad (35)$$

where the (X_M, Y_M, Z_M) are the coordinates in the monitor coordinate system and (X_C, Y_C, Z_C) are the coordinates in the camera coordinate system. The (t_x, t_y, t_z) represent the position of the camera coordinate system in the monitor coordinate system.

The intersection of the gaze vector with the monitor plane can be solved by finding λ that satisfies the following equation

$$\mathbf{O}_M + \lambda \mathbf{N}_M = \mathbf{P}_M, \quad (36)$$

where the point $\mathbf{P}_M = [X_M, Y_M, 0, 1]^T$ is the gaze position on the monitor screen. $\mathbf{O}_H$ is the origin and $\mathbf{N}_H$ is the direction of the gaze vector in the monitor coordinate system.

It is assumed that the user is looking into the center of the monitor screen at the beginning. The offset is calculated on the basis of this assumption. The offset value is the distance between the actual gaze position and the center of the monitor screen. This value is used to ensure the correct operation at the beginning. Otherwise there is a possibility that gaze direction goes beyond the monitor screen. After the gaze position on the monitor screen is obtained; the user can control the cursor in the graphical user interface (GUI).

5. EXPERIMENTAL RESULTS

The experiments of this thesis are divided into three parts in which the performance of the system is evaluated. To demonstrate the correct operation of the proposed approach, the initialization and facial feature tracking were tested with real image sequences. The head pose estimation and gaze detection were tested with synthetic data. This makes a quantitative analysis possible because the actual pose of the head is not known with the real image sequences. Finally, the real-time performance of the system is verified. The experimental results are summarized in the following sections.

5.1. Tests with real image sequences

Two different test sets of image sequences were collected. The first set was used to evaluate automatic initialization of the system. This set consists of three sequences where the user does not move his head. A frontal view of the face is needed for proper initialization. The second set consists of four recorded test image sequences where the user moves his head. Based on these sequences the facial feature tracking ability is verified. All the sequences are natural with large head motions and pose changes. The user in the images tries to control the cursor on the monitor screen.

The first test sequence for facial feature tracking was recorded by a Sony DFW-X700 digital video camera and the others were recorded by a Sony DFW-VL500 digital video camera. The test data was collected under typical office illumination with 320x240 pixel resolution and 24-bit color resolution. The frame rate was 15 frames per second. The imaging geometry is basically the same as that shown in Fig. 15. The user is looking at the monitor and the camera is placed above the monitor. The face of the user is in the center of the image acquired by the camera. The arrangement is typical for the proposed system. The computations were performed using Matlab-software.

5.1.1. Initialization

In this experiment, the automatic initialization capability of the system was tested. The three test sequences used were named A1, B1, and C1. To test the robustness against bad lighting, the test sequences were captured in different lighting conditions. Typically glasses introduce a serious problem for face detection systems. Therefore the first sequence contains a man with no glasses and the other two men with glasses. All the sequences consist of 100 images. For each image in the sequence the initialization procedure was run. First the skin-like regions are detected and possible facial features are detected within these regions. The presence of the face is finally verified by using the geometrical face model. The model consists of seven facial features like eyes, mouth, nostrils and eyebrows. The initialization was successful if the evaluation value was more than 0.6. The procedure is described in more detail in Section 4.1 and more examples of initialization are presented in Appendix 1.

For test sequence A1, 97 successful initializations were performed. Only 3 attempts were unsuccessful, mainly caused by small movement of the eye. The errors were not so critical and these false cases could even be accepted. The reason for a very good result is almost ideal lighting conditions for the algorithm. Moreover, the view of the face is near frontal and the facial features are easily separable as can be seen in Fig. 16. One possible disadvantage was observed with sequence A1. The eyebrows are hard to extract because of low contrast in the image. If the eyebrow is missing, the evaluation value for the face is decreased. However, the significance of the eyebrows is quite small in the geometrical face model and its influence is negligible.

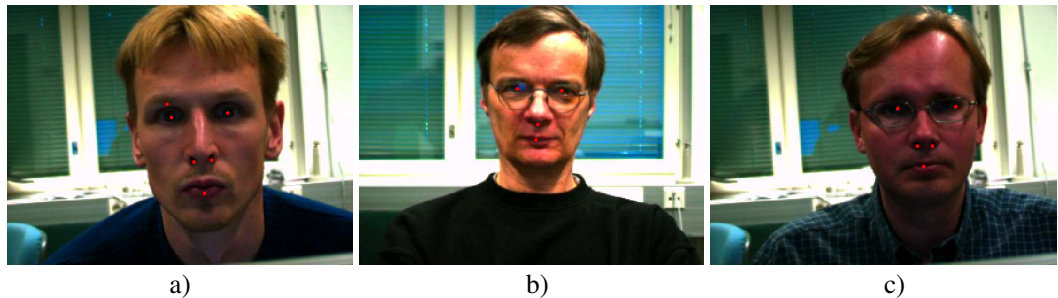


Figure 16. Initialization examples for test sequences.

With test sequence B1, 95 correct initializations were achieved. In the four false cases, the mouth was not found at all. The problem of the missing mouth probably originates from the inadequate lighting conditions. Nevertheless, the result is very good because the man has glasses. In Fig. 16 it can be noticed that only one nostril is detected and the location is between the true locations of the nostrils. The problem is shadows caused by lighting. In addition, the same phenomenon was observed as with the previous sequence. The eyebrows were not found at all. The reason is different this time. The dark frames of the glasses partly occlude the eyebrows. Nevertheless, the missing features do not deteriorate the initialization result.

Test sequence C1 seems to be more problematic for the system due to the man having glasses and bad lighting conditions. Only 67 successful initializations were obtained. The failures are mainly caused by the frame of the glasses. The eye cannot be distinguished from the frame due to the large dark area caused by the frames. Typically the eyes themselves also construct a dark area on the image. The fifteen failures fall into this category. In 10 cases, the facial features were not found at all due to bad lighting and the issues mentioned before. For the sequence C1 the result is quite poor. It should be remembered that after failure the real system tries to re-initialize and only one successful initialization is needed to start tracking.

The results for the initialization test are summarized in Table 1. It can be stated that the initialization will give a good result if a near frontal image of the face is captured under adequate lighting. On the other hand, the user with glasses in certain lighting conditions will cause major problems for the system. In the future research this should be studied more carefully.

Table 1. Result for initialization test

Test sequence	Successful initializations
A1	97
B1	95
C1	67

5.1.2. Facial feature tracking

In order to test the system's ability to track the locations of the facial features, an experiment with four real image sequences was run. The sequences are named A2, B2, C2 and D2. In this experiment, the full functionality of the proposed system was used. The system will be automatically initialized using the procedure tested in previous section. After that, the tracking of facial features is started and the head pose is computed for each image in the sequence. Because the ground truth values for the facial feature locations or head poses were not known, the quantitative evaluation of the tracker was impossible. However, it was visually verified that the system could properly track facial features.

Fig. 17 shows the facial feature locations during tracking for test sequence A1. The length of the video was 100 frames and only the selected frames are shown. The mouth and the eyes are marked with red dots. It was observed that the tracker successfully follows the positions of the features. The use of EKF provides a smooth trajectory for the features. Fig. 18 shows a trajectory for the right eye x-coordinate. The black line presents the filtered measurement computed by the EKF algorithm and the blue dots present actual measurements. It can be seen that a smoother result is obtained by using EKF.

Successful operation was also achieved with sequences B2 and C2. The results are shown in Figs. 19 and 20, respectively. The length of both videos is 200 frames. With sequence B2 it can be seen that the position of the mouth is slightly misplaced in frames 120 and 130. This is due to the shadows caused by a large amount of head rotation. The current implementation of the system cannot handle shadows. There are also some small deviations in the position of the eye, as in frame 20. Pupils that are not in the center of the eye region cause this. Typically the errors occur in the x-axis direction and the position of the eye seems to be more accurate in the y-axis direction. The sequence C2 contains a man with a beard and it should be noticed that the system still works correctly. However, also in this case there are small errors in the feature locations. During testing it was observed that a proper setting for model and measurement uncertainties of the EKF algorithm has a great impact on the tracking result. The uncertainties should be as realistic as possible and proper approximations can improve the tracking result significantly. A shortcoming is the difficulty of approximation. Moreover, the changing imaging conditions can alter the magnitude of the measurement error.

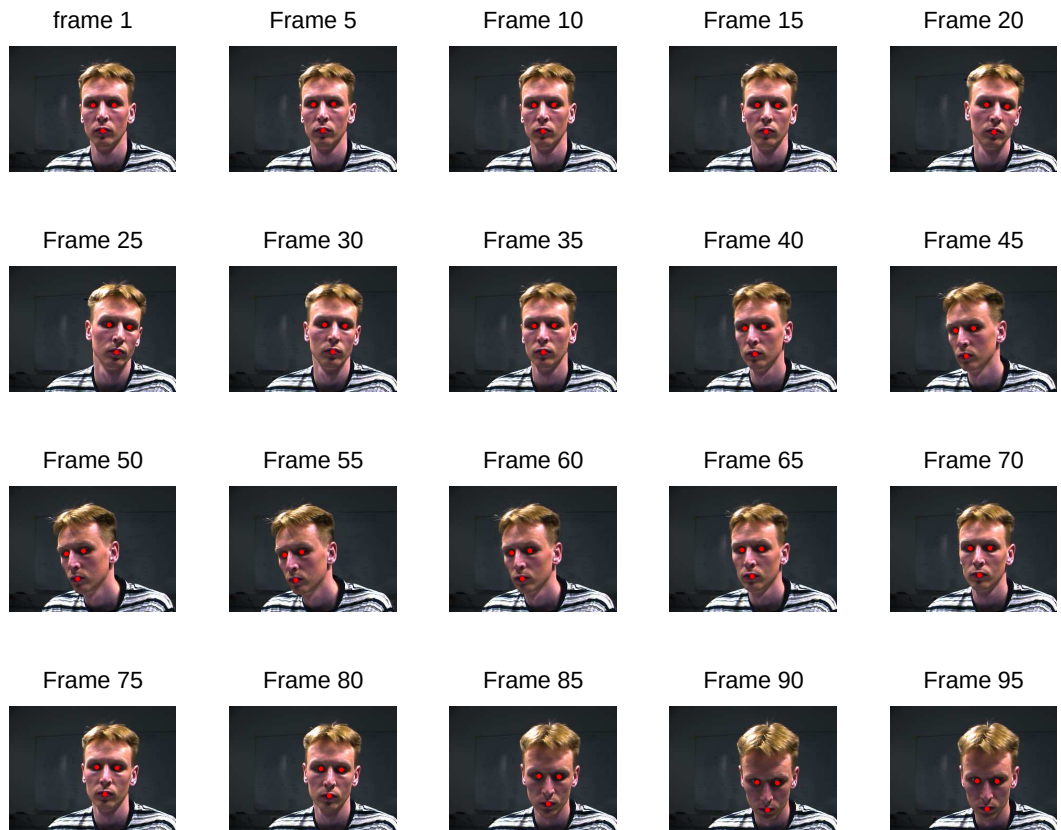


Figure 17. Tracking result for test sequence A2.

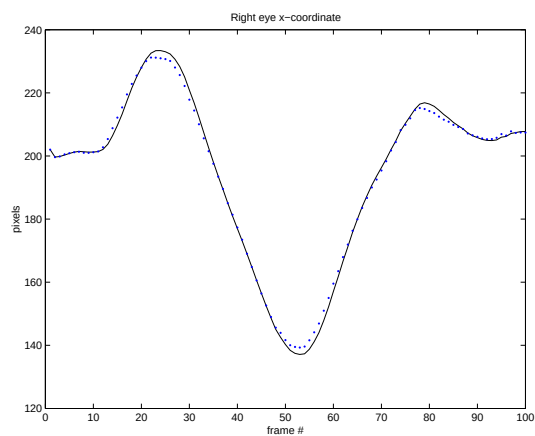


Figure 18. Trajectory for right eye x-coordinate with test sequence A2.

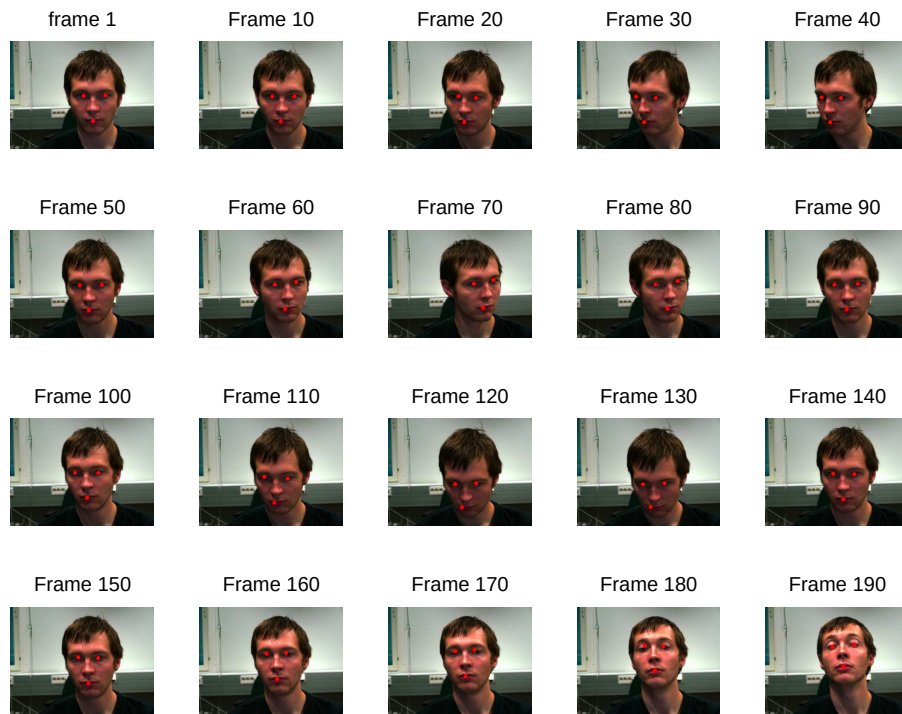


Figure 19. Tracking result for test sequence B2.



Figure 20. Tracking result for test sequence C2.

The test sequence D2 is an example where facial feature extraction will fail. A couple of frames after the successful initialization of the tracker an error occurs in the extraction of the left eye. The reason can be seen in Fig. 21. The upper row shows input images for the algorithm, and in the middle row the left eye area is zoomed. In the bottom row, the feature extraction result is shown for each frame. It can be seen that in frame 2 the feature extraction detects the frame of the glasses and the location of the eye starts to slide towards the right eye. Finally, in frame 5 the positions of both eyes are in the same position and the tracking and pose estimation totally fails. Of course in the practical system there are certain geometrical rules which make this kind of situation impossible. However, the problem with glasses is shown here because it has been observed to be a major problem for the current system. Certain types of glasses and bad lighting conditions can cause the system to fail continuously.

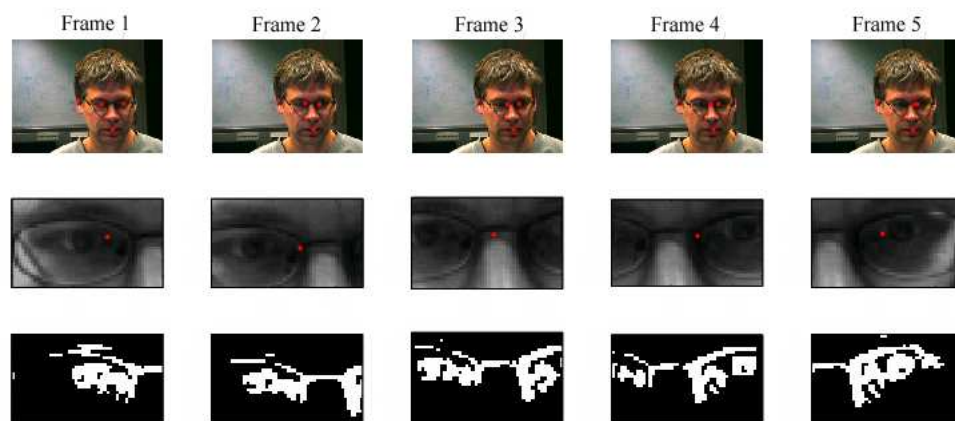


Figure 21. Error in tracking with test sequence D2.

5.2. Tests with synthesized data

It is difficult to obtain a ground truth for the head pose if there are no external measurement sensors. In this study, to evaluate the pose estimation, synthesized ground truth-values for the head pose were generated. Construction of synthetic data makes the comparison between estimated and true values possible. The first synthesized data of 100 frames contains head poses corresponding to a real situation where the user is trying to control the cursor in the monitor screen. For the gaze detection tests, another synthetic data of 90 poses was generated. This sequence contains certain movement of the cursor on the monitor screen.

5.2.1. Pose estimation

In this experiment, the theoretical head pose estimation accuracy is evaluated. The system applies the extended Kalman filter (EKF) for head pose estimation. The EKF algorithm and implementation issues are described in Sections 3.3.5 and 4.2. In this test, the difference compared to the previous experiment is that the synthesized sequence of poses is used instead of real data. The initial state

vector for this experiment was $\hat{\mathbf{x}}_{0|0} = [19.11, 8.65, 685.09, -2.34, 0.02, 1.59]$. The vector contains the position in millimeters and orientation in radians of the head. The focal length used in the simulations was $f = 15$ mm. The conversion factors D_u and D_v were both 50. These factors are used to convert metric units into pixels as mentioned in Section 3.3.1.

The measurements needed by the EKF algorithm are generated by projecting the head model points to the pixel coordinates in the image plane using ground truth pose values. In order to test the performance of the algorithm, random noise with a variance of 4 pixels is added to the measurements. This means that the assumed measurement accuracy is described with a variance $\sigma_q^2 = 4$.

Fig. 22 shows an example of the estimated translation and rotation parameters during tracking compared to the ground truth. In Fig. 23 the corresponding error is shown. It can be clearly seen that the error in z-translation is bigger than in other translations. The obvious reason is that the depth information is more uncertain due to the projective camera model used.

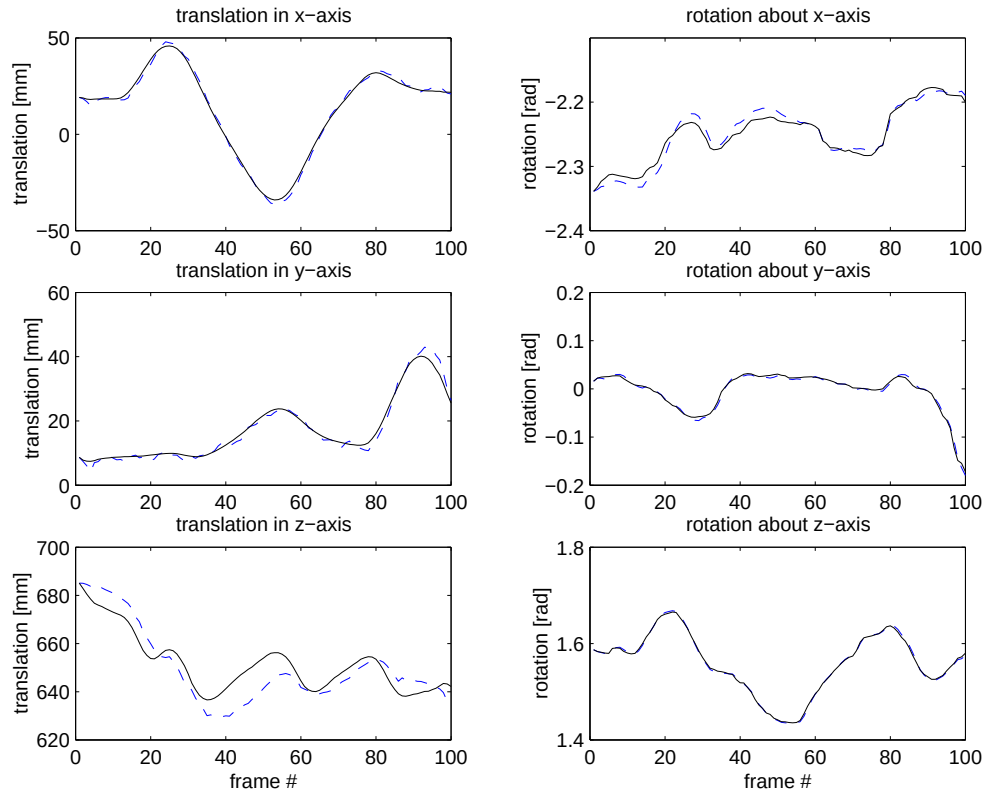


Figure 22. An example head pose estimation experiment. The ground truth is drawn with a solid black line and the estimated pose with a dashed blue line.

The root mean square error is used as a measure for the pose estimation accuracy. In order to get more reliable results the mean value of 100 simulations is reported. The obtained root mean square (RMS) errors for translation in x-, y-, and z-axes directions and rotation about x-, y-, and z-axis are 1.61 mm, 1.30 mm, 6.23 mm, 0.008 rad, 0.004 rad and 0.003 rad. It can be noticed that the error in the z-axis direction is larger than in the x- and y-axis direction. Because the size of the test data is so small, widespread conclusions cannot be made. However, good preliminary results were obtained. Furthermore, the obtained results can be

considered to be ideal because they were achieved through simulation. It should be also remembered that the obtained result depends on the measurement error. If the real measurement error is bigger, the accuracy of the pose estimation is worse.

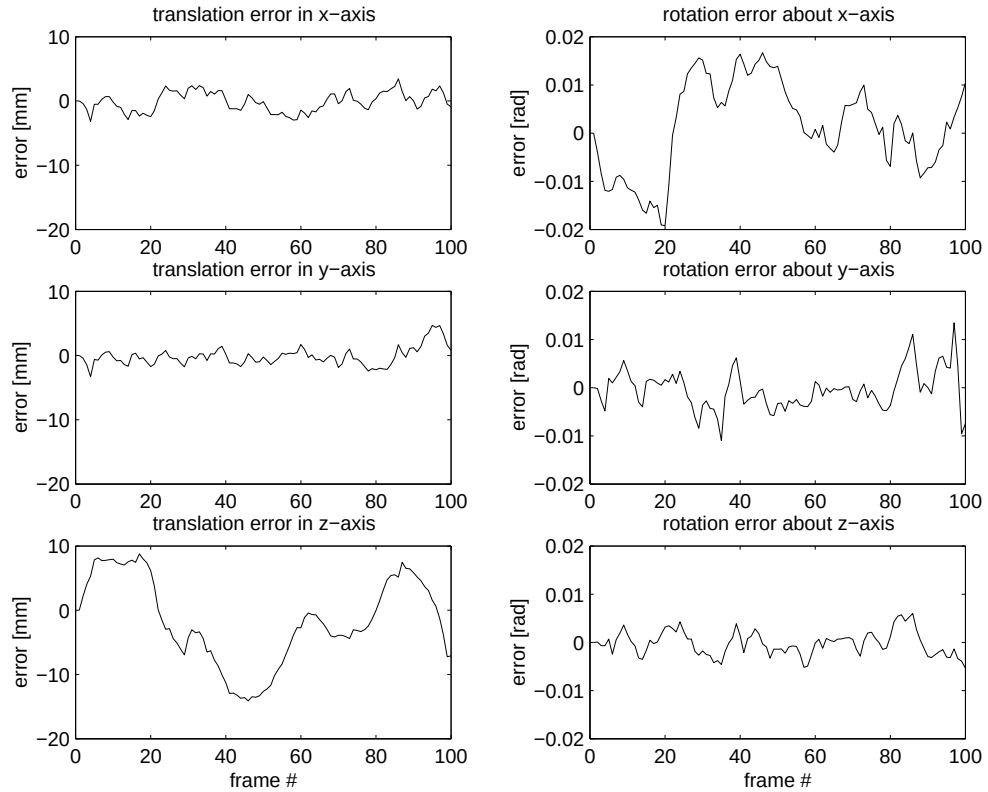


Figure 23. An example of the pose estimation error for each parameter.

5.2.2. Gaze detection

The goal of this experiment was to ensure the correctness of the gaze detection and to study the detection accuracy achievable using the developed system. In order to evaluate the accuracy analytically, simulated data was generated. At first, a trajectory for the gaze on the monitor screen was specified. The coordinates of the points determining the trajectory are the ground truth points. The trajectory used in the experiment is shown with a black line in Fig. 24. Also, an example of estimated gaze is depicted with a blue line in the same figure. The measurement noise with standard deviation of 2 pixels was used in this particular case. It can be seen that significant deviations are caused by abrupt changes in the trajectory. The initial state vector for this experiment was $\hat{\mathbf{x}}_{0|0} = [0, 250, 600, -0.39, 0.29, 1.45]$. Other parameters were the same as those defined in previous section. This gives the frame for the conditions where these simulations were performed.

Next, the ground truth points are projected to the corresponding head pose values. This is done by using geometrical constraints and assuming that there is

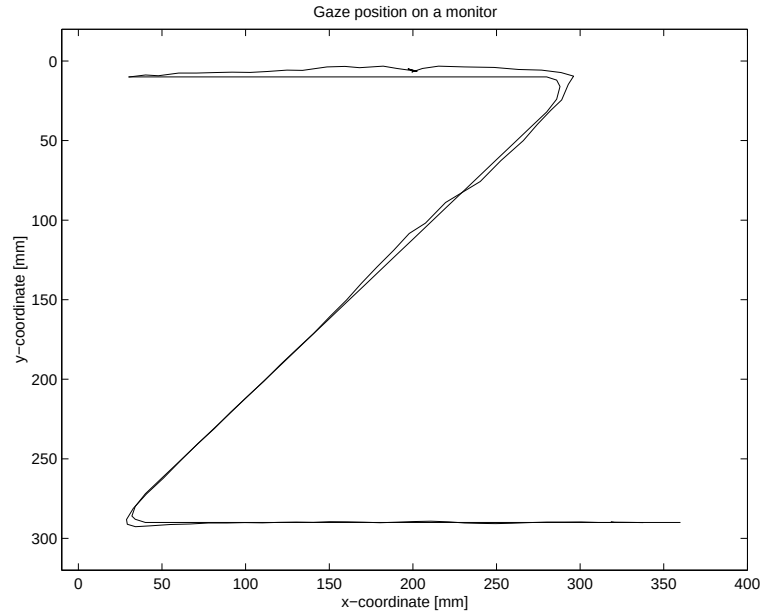


Figure 24. Gaze position on the monitor screen. The ground truth position is marked with a black line and the estimated gaze position is marked with a blue line.

only rotational movement of the head. The measurements needed by the EKF algorithm are produced using these head poses. The noise added to the measurements is varied between the standard deviation of 1 to 8 pixels. The EKF algorithm is then applied to head pose estimation and the gaze is estimated as described in Section 4.3. For each run with a certain noise level, the error between the estimated gaze and the ground truth gaze is computed. In order to measure gaze detection precision, the standard deviation for the error is calculated. The EKF algorithm was applied 100 times to the same noise level in order to improve statistical confidence. The final result is the mean value for a standard deviation σ of 100 measurements. The result is shown in Fig. 25 where the standard deviation of the position error on the monitor plane is presented as a function of the standard deviation of the measurement noise.

It is obvious that the accuracy of the gaze detection decreases if more noise is added to the measurements. In the current implementation, the assumed measurement accuracy is 2 pixels, which will correspond to a gaze detection accuracy of 6 mm on the monitor screen. If the Gaussian distribution is assumed, the maximum error would be about 3σ , resulting in 18 mm error on the monitor plane. This means that the location of the cursor varies between -18 mm to 18 mm from the correct location. In the real situation, the abrupt changes in the cursor position are avoided by filtering. The filter could take into account N previous locations to smooth the cursor trajectory. In theory, the standard deviation of the error should decrease based on the equation $\frac{\sigma}{\sqrt{N}}$. For example, if the gaze detection accuracy σ is 6 mm and the number of locations N is 9, the improved accuracy would be 2 mm. Actually, this means a position range between -6 mm to 6mm, which is a satisfactory result. If the gaze detection accuracy would be within these limits, cursor control using head movements would be possible. It should be remembered that this is the theoretical gaze detection accuracy of the

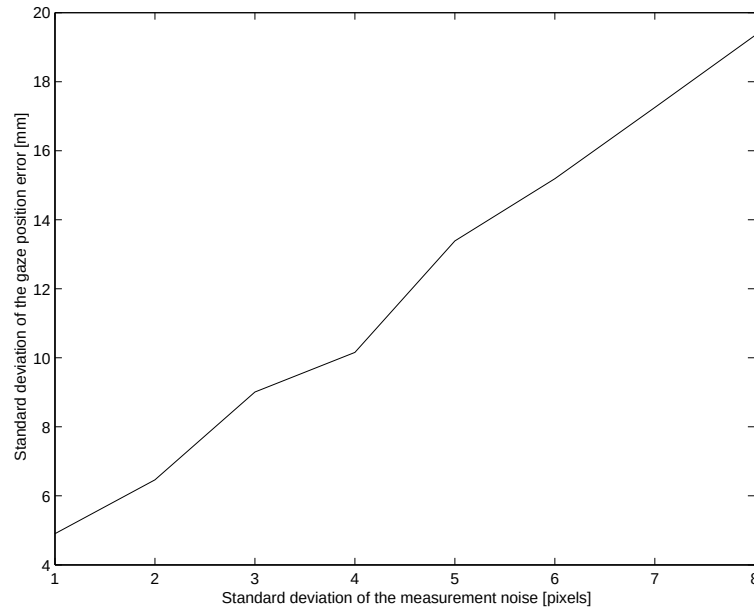


Figure 25. Gaze position estimation error.

proposed system. However, in practise the error can be more significant and then the accuracy is worse. It has been observed with real image sequences that in bad lighting situations the measurement accuracy can be 4 pixels, for example. In this case, the gaze detection accuracy would be about 13 mm. In this case the error range is between -39 mm to 39 mm. Thus, the overall variation is 78 mm which is unacceptable. In this case, more filtering is needed. A shortcoming is the possible delays that occur if the trajectory is extensively filtered.

5.3. Real-time performance

In addition to the Matlab implementation, the proposed system was also implemented using C++ under the MS Windows 2000 operating system to validate the real-time performance. No external machine vision libraries were used. The system runs in real-time on a PC with a 2.2 GHz AMD Athlon XP CPU and 512 Mb memory. The processing speed in the different steps of the system was measured using the operating system's timer. The results are shown in Table 2. The presented time values are mean values of the 100 measured execution times. It should be remembered that the values are approximate values due to the limited resolution of the system timer.

The total time for initialization was 10 ms and for tracking 2 ms. The video acquisition time is not included in the previous table because it does not depend on the implementation. The overall cycle time for the system is 66 ms, including the video acquisition. This provides a 15 frames per second frame rate for the system. The video acquisition frame rate can be easily increased if necessary. It should be also remembered that the processing speed also depends on the sizes of the image, faces and facial features to be processed. The reported execution times are mean values of typical situations.

It can be seen that the feature extraction time in the tracking phase is faster

Table 2. Execution times for different steps

	time [ms]
System Initialization	
Skin Detection	1
Binary Morphology	2
Connected Component Analysis	2
Valley Detection	4
Best Constellation search	1
Tracking	
Tracker Initialization	< 1
EKF algorithm	< 1
Feature Measurement	1
Gaze Detection	< 1

than in the initialization phase. This is due to the prediction and search window capability of the EKF algorithm, as mentioned in Section 3.4. Based on the measurements, the time consuming task is the initialization phase. On the other hand, this phase fulfills the real-time demands of the system and no optimizations are needed. It can be concluded that although further development is in progress and the implementation is non-optimized, the current implementation of the system is fast enough to run on a standard PC.

6. DISCUSSION

Implementation of a real-time system to track facial features and estimate head pose sets hard requirements on the algorithms utilized. To accomplish this demanding task, many different algorithms are usually applied. In this study, promising methods were searched for from the literature. Based on the survey the implementation for such a system is proposed.

Skin detection has been a popular method for face candidate region detection. It has been noticed to be highly suitable for real-time systems due to fast processing time and simple implementation. A shortcoming is that changing illumination conditions decrease the performance of the skin detection. Because the initialization stage of the proposed system depends heavily on skin detection, dramatically changing or bad illumination can cause facial features to be undetectable. Also if the skin areas are not solid and proper feature extraction is impossible using this method. Skin detection can be improved, for example, by using information from the previous frames. In the current implementation, skin detection is performed for every image individually. There are possibilities to use other techniques for initial facial region detection like motion-based or texture-based segmentation. However, skin detection provides a good starting point for facial feature extraction.

Morphological image processing techniques have been applied across a broad range of real-time machine vision applications. These operations have proven to be efficient and powerful due to fast processing. In this study morphological methods are employed for facial feature extraction. Usually, all the facial features are detected, but the problem is how to distinguish particular features. A geometrical face model is used for this purpose. In the proposed face model, there are seven features; eyes, mouth, nostrils and eyebrows. The model is achieved by training with several facial images. A more accurate face model could be generated, but any general model that applies for everyone cannot be constructed. It was observed that the feature locations are roughly located by using this approach. Based on the experiments, the expected initialization result for a user without glasses can be quite good. On the other hand, the result for a user with glasses can be quite poor. A user with glasses under bad lighting is definitely the major problem for the system. To overcome problems, more advanced methods should be developed.

In tracking, a typical measurement error is that the connected component labelling process produces many separate connected regions for same feature. If the feature is split to many individual components, the center of mass for the feature is different. Of course, the result is improved by filtering but errors cannot be totally removed. The EKF algorithm can be adjusted to work very well in certain conditions. If the conditions change the tracking result decreases. It should therefore be accepted that the system works under some restrictions or some compromise should be made in adjustment of the EKF algorithm. If there are errors in the facial feature extraction that decreases the pose estimation accuracy, also cursor control suffers dramatically. Special attention should be focused on the development of facial feature extraction methods.

The tracking can totally fail in some situations. It has been observed that a user with glasses will cause most of the failures. In the cursor control there is a drawback if a failure in the tracking occurs. In this case, the cursor is not

controlled at all for some time. The system should recover the tracking as soon as possible. The system should therefore be so robust that the occurrence of a failure is minimized. A lot of additional work remains to achieve this goal.

In future, the features to be extracted and tracked could be the corners of the eyes and the mouth instead of the centers of the eyes and the mouth. The corners are more permanent than the centers of these features due to possible deformations during tracking. The natural deformations of the mouth will cause large deviations in the image measurements and pose estimation. Also, the amount of features is increased from three to six, which allows more robust pose estimation and tracking. The tip of the nose could also be a rigid feature to be extracted, but that is a very difficult task due to the low contrast of skin.

One important issue is to choose an appropriate model for the head in the tracking phase. In this thesis three facial features, the eyes and the mouth, roughly model the head. In the future, more accurate 3-D models could be applied that make it possible to achieve better head pose estimates. The head model could be, for example, a 3-D wire frame model. The model is obtained by training with several different samples. A shortcoming with more advanced models is that they typically increase the computational load. Furthermore, a general model for the head that applies to all the humans is almost impossible to generate.

An alternative is to use Kalman filtering to recover also the structure of the tracked object. In this case, features do not necessarily have to be facial features like eyes or mouth, but only stable features are needed. These feature points can be, for example, discontinuities caused by edges etc. The advantage is that there is no need for some particular feature to be visible all the time in the tracking phase. When the feature is lost, a new one is selected. In this case also the number of features to be tracked could be increased. A difficult problem is how to select good features for tracking. For a large number of feature points, it is crucial to perform pre-selection, for example, by eliminating all the points that are occluded with respect to the camera. There are some automatic pre-selection algorithms for finding out optimal feature points, but it is nevertheless a very difficult task. The disadvantage of those selection algorithms is usually the computational complexity.

The system was basically tested with both real image sequences and synthetic data. Although there are some problems, very good preliminary results were achieved. However, it should be remembered that the accuracy values obtained are ideal values, which apply under some constraints. At this point of development, the conducted testing was enough to prove the correctness of the approach. Especially the low computational cost of the method makes it usable for real-time applications.

7. SUMMARY

In future proactive systems, the user can control computers and other devices using head movements. Machine vision provides an excellent way of realizing such advanced interaction. This work is a study on the techniques needed for implementing such a system.

The 3-D position and orientation (pose) of the head is estimated by extracting facial feature points and utilizing correspondences between rigid face model points. Facial extraction methods are extensively studied based on existing knowledge in the literature. As a result, an efficient extraction method using skin detection, gray-scale morphology and geometrical face model is proposed. The extended Kalman filtering framework (EKF) is applied for the estimation of the 3-D pose of a moving human head.

The system automatically initializes whenever the user is front of a camera located above the monitor screen. The facial region is roughly detected and facial features (eyes, mouth, nostrils and eyebrows) are extracted. The initial 3-D pose of the head is solved by using the perspective-three-point (P3P) algorithm. The eyes and the mouth are tracked throughout the image sequence and the head pose is estimated continuously. The obtained head pose is used to locate where the user is looking, i.e. the gaze is detected. To demonstrate the practicality of the approach, gaze detection is applied to cursor control on a monitor screen.

Experiments were carried out with real image sequences and synthetic data. Tests with real sequences showed that the system is able to extract facial features reliably from a near frontal view of the face under proper lighting conditions. Furthermore, the facial features were successfully tracked. The experiments with synthetic data enabled a quantitative accuracy analysis of the system. The pose estimation accuracy was measured with RMS error between the estimated and ground truth-values. The gaze position accuracy between computed and real ones were measured as a function of facial feature measurement error in the image. Good preliminary results were obtained with these experiments.

The system was implemented with C++ under the MS Windows 2000 operating system to validate the real-time performance. The achieved frame rate for head pose estimation was 15 frames per second using a standard PC. Experiments clearly show that significantly higher frame rates are possible. The fast implementation proves the basic premise that the proposed approach can be applied in platforms like mobile devices where high computational resources are not available.

There are some restrictions in the current implementation. A user with glasses, occlusion, and shadows will cause problems for the system. The extracted features were found to be unstable which will deteriorate pose estimation accuracy. In future work, the feature extraction methods will be considered more closely and more stable features should be found. However, the implemented system provides a good basis for further development.

8. BIBLIOGRAPHY

- [1] Tennenhouse D. (2000) Proactive Computing. *Communications of the ACM* 43, pp. 43–50.
- [2] Weiser M. (1991) The Computer for the 21st Century. *Scientific American* 265, pp. 94–104.
- [3] Satyanarayanan M. (2001) Pervasive Computing: Vision and Challenges. *IEEE Personal Communications* 8, pp. 10–17.
- [4] Pentland A. (2000) Looking at People: Sensing for Ubiquitous and Wearable Computing. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22, pp. 107–119.
- [5] Essa I.A. (2000) Ubiquitous Sensing for Smart and Aware Environments. *IEEE Personal Communications* 7, pp. 47–49.
- [6] Turk M. & Robertson G. (2000) Perceptual User Interfaces. *Communications of the ACM* 43, pp. 33–34.
- [7] Huang K.S. & Trivedi M.M. (2003) Video Arrays for Real-time Tracking of Person, Head, and Face in an Intelligent Room. *Machine Vision and Applications* 14, pp. 103–111.
- [8] Hjelmas E. & Low B.K. (2001) Face Detection: A Survey. *Computer Vision and Image Understanding* 83, pp. 236–274.
- [9] Yang M.H., Kriegman D. & Ahuja N. (2002) Detecting Faces in Images: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24, pp. 34–58.
- [10] Zhao W., Chellappa R., Rosenfeld A. & Phillips P.J. (2002) Face Recognition: A Literature Survey. Technical Report CAR-TR-948, Center for Automation Research, University of Maryland.
- [11] Fasel B. & Luetttin J. (2003) Automatic Facial Expression Analysis: A Survey. *Pattern Recognition* 36, pp. 259–275.
- [12] Pantic M. & Rothkrantz J.M. (2000) Automatic Analysis of Facial Expressions: The State of the Art. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22, pp. 1424–1445.
- [13] Sobottka K. & Pitas I. (1998) A Novel Method for Automatic Face Segmentation, Facial Feature Extraction and Tracking. *Signal Processing: Image Communication* 12, pp. 263–281.
- [14] Kawaguchi T. & Rizon M. (2003) Iris Detection Using Intensity and Edge Information. *Pattern Recognition* 36, pp. 549–562.
- [15] Sirohey S.A. & Rosenfeld A. (2001) Eye Detection in a Face Image Using Linear and Nonlinear Filters. *Pattern Recognition* 34, pp. 1367–1391.

- [16] Jeng S.H., Liao H.Y.M., Han C.C., Chern M.Y. & Liu Y.T. (1998) Facial Feature Detection Using Geometrical Face Model: An Efficient Approach. *Pattern Recognition* 31, pp. 273–282.
- [17] Gonzalez R.C. & Woods R.E. (2002) *Digital Image Processing*. Prentice Hall, Upper Sadler River, New Jersey.
- [18] Soriano M., Martinkauppi B., Huovinen S. & Laaksonen M. (2003) Adaptive Skin Color Modeling Using the Skin Locus for Selecting Training Pixels. *Pattern Recognition* 36, pp. 681–690.
- [19] Hsu R.L., Abdel-Mottaleb M. & Jain A. (2002) Face Detection In Color Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24, pp. 696–706.
- [20] Martinkauppi B., Soriano M. & Pietikäinen M. (2003) Detection of Skin Color under Changing Illumination: A Comparative Study. In: *Proceedings of the 12th International Conference on Image Analysis and Processing*, pp. 652–657, Mantova, Italy.
- [21] Bala L.P., Talmi K. & Liu J. (1997) Automatic Detection and Tracking of Faces and Facial Features in Video Sequences. In: *Proceedings of Picture Coding Symposium 1997*, Berlin, Germany, pp. 251–256.
- [22] Yow K.C. & Cipolla R. (1997) Feature-Based Human Face Detection. *Image and Vision Computing* 15, pp. 713–735.
- [23] Pigeon S. & Vandendorpe L. (1997) The M2VTS Multimodal Face Database. In: Bigun J., Chollet C. & Borgefors G., editors, *Lecture Notes in Computer Science: Audio- and Video-Based Biometric Person Authentication*, volume 1206, pp. 403–409.
- [24] Brunelli R. & Poggio T. (1993) Face Recognition: Features versus Templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 15, pp. 1042–1052.
- [25] Wong K.W., Lam K.M. & Siu W.C. (2003) A Robust Scheme for Live Detection of Human Faces in Color Images. *Signal Processing: Image Communication* 18, pp. 103–114.
- [26] Wong K.W., Lam K.M. & Siu W.C. (2001) An Efficient Algorithm for Human Face Detection and Facial Feature Extraction under Different Conditions. *Pattern Recognition* 34, pp. 1993–2004.
- [27] Reisfeld D. & Yeshurun Y. (1992) Robust Detection of Facial Features by Generalized Symmetry. In: *Proceedings of the 11th International Conference on Pattern Recognition*, pp. 117–120.
- [28] Reisfeld D. & Yeshurun Y. (1998) Preprocessing of Face Images: Detection of Features and Pose Normalization. *Computer Vision and Image Understanding* 71, pp. 413–430.

- [29] Loy G. & Zelinsky A. (2003) Fast Radial Symmetry for Detecting Points of Interest. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25, pp. 959–973.
- [30] Herpers R., Michaelis M., Lichtenauera K.H. & Sommer G. (1996) Edge Keypoint Detection in Facial Regions. In: *Proceedings of the Second International Conference on Automatic Face and Gesture Recognition*, pp. 212–217.
- [31] Haralick R.M. & Shapiro L.G. (1992) *Computer and Robot Vision*. Addison-Wesley, Reading Massachusetts.
- [32] Chow G. & Li X. (1993) Towards a System for Automatic Facial Feature Detection. *Pattern Recognition* 26, pp. 1739–1755.
- [33] Kass M., Witkin A. & Terzopoulos D. (1988) Snakes: Active Contour Models. *International Journal of Computer Vision* 1, pp. 321–331.
- [34] Yuille A., Hallinan P. & Cohen D. (1992) Feature Extraction from Faces Using Deformable Templates. *International Journal of Computer Vision* 8, pp. 99–111.
- [35] Xie X., Sudhakar R. & Zhuang H. (1994) On Improving Eye Feature Extraction Using Deformable Templates. *Pattern Recognition* 27, pp. 791–799.
- [36] Lanitis A., Taylor C.J. & Cootes T.F. (1997) Automatic Interpretation and Coding of Face Images Using Flexible Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19, pp. 743–756.
- [37] Azarbayejani A., Starner T., Horowitz B. & Pentland A. (1993) Visually Controlled Graphics. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 15, pp. 602–605.
- [38] Azarbayejani A. & Pentland A. (1995) Recursive Estimation of Motion, Structure, and Focal Length. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 17, pp. 562–575.
- [39] Jebara T.S. & Pentland A. (1997) Parametrized Structure from Motion for 3-D Adaptive Feedback Tracking of Faces. In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 144–150.
- [40] Ström J., Jebara T., Basu S. & Pentland A. (1999) Real Time Tracking and Modeling of Faces: An EKF-Based Analysis by Synthesis Approach. In: *Proceedings of the International Conference on Computer Vision: Workshop on Modeling People*, pp. 55–61, Kerkyra, Greece.
- [41] Black M.J. & Yacoob Y. (1997) Recognizing Facial Expressions in Image Sequences Using Local Parametrized Models of Image Motion. *International Journal of Computer Vision* 25, pp. 23–48.
- [42] Basu S., Essa I. & Pentland A. (1996) Motion Regularization for Model-Based Head Tracking. In: *Proceedings of the 13th IEEE International Conference on Pattern Recognition*, pp. 611–616, Vienna, Austria.

- [43] Ballard P. & Stockman G. (1995) Controlling a Computer via Facial Aspect. *IEEE Transactions on Systems, Man, and Cybernetics* 25, pp. 669–677.
- [44] Yang J., Stiefelhagen R., Meier U. & Waibel A. (1998) Real-time Face and Facial Feature Tracking and Applications. In: *Proceedings of AVSP'98, Ter-rigal Australia*, pp. 79–84.
- [45] Toyama K. & Hager G. (1999) Incremental Focus of Attention for Robust Vision-Based Tracking. *International Journal of Computer Vision* 35, pp. 45–63.
- [46] Black M.J., Yacoob Y. & Ju S.X. (1997) Motion-Based Recognition. In: *Shah M. & Jain R., editors, Computational Imaging and Vision*, Kluwer Academic Publishers, volume 9, pp. 245–269.
- [47] Repo T. (2002) Modeling of Structured 3-D Environment from Monocular Image Sequences. Ph.D. thesis, University of Oulu, Department of Electrical and Information Engineering.
- [48] Isard M. & Blake A. (1996) Contour Tracking by Stochastic Propagation of Conditional Density. In: *Proceedings of European Conference on Computer Vision*, volume 1, pp. 343–356.
- [49] Lucas B. & Kanade T. (1981) An Iterative Image Registration Technique with an Application to Stereo Vision. In: *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, pp. 674–679.
- [50] Kalman R.E. (1960) A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME-Journal of Basic Engineering Series D*, pp. 35–45.
- [51] Heikkilä J. (1997) Accurate Camera Calibration and Feature Based 3-D Reconstruction from Monocular Image Sequences. Ph.D. thesis, University of Oulu, Department of Electrical Engineering.
- [52] Fischler M.A. & Bolles R.C. (1981) Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM* 24, pp. 381–395.
- [53] Shapiro L.G. & Stockman G.C. (2001) *Computer Vision*. Prentice Hall.

9. APPENDICES

Appendix 1 Facial feature extraction examples

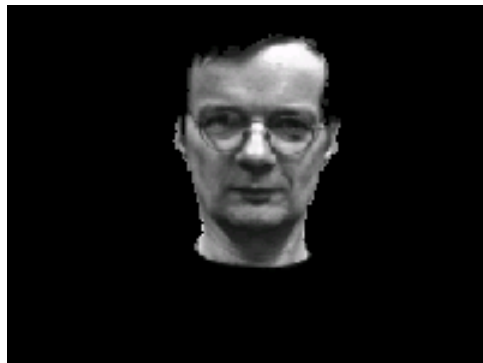
Input image



Skin detection



Face candidate



Valley detection



Feature candidates



Best constellation



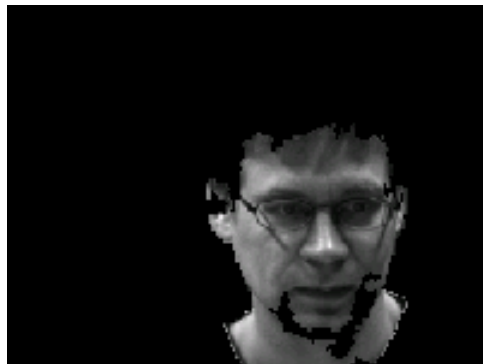
Input image



Skin detection



Face candidate



Valley detection



Feature candidates



Best constellation



Input image



Skin detection



Face candidate



Valley detection



Feature candidates



Best constellation



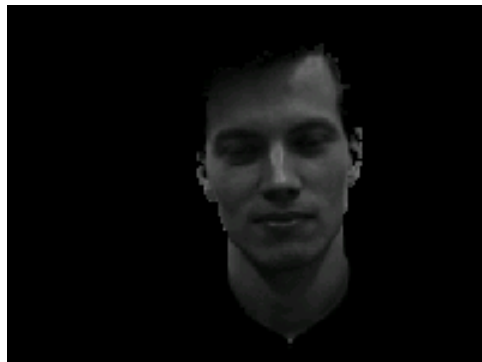
Input image



Skin detection



Face candidate



Valley detection



Feature candidates



Best constellation



Input image



Skin detection



Face candidate



Valley detection



Feature candidates



Best constellation

