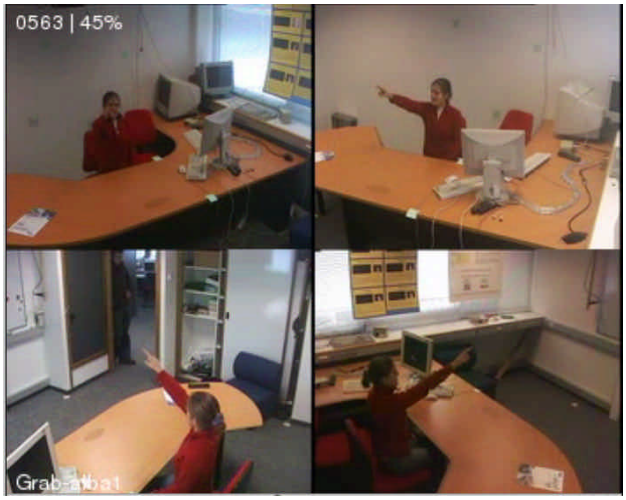
Proceedings

POINTING 2004

International Workshop on
Visual Observation of Deictic Gestures

In association with ICPR 04
Cambridge, UK - 22 August 2004



FGnet

ICPR2004
17th INTERNATIONAL CONFERENCE ON PATTERN RECOGNITION
CAMBRIDGE, UNITED KINGDOM, 23 - 26 AUGUST 2004

Sponsored by : FGnet - (IST-2000-26434) Face and Gesture Recognition Working group

Pointing 2004 Workshop Proceedings

WORKSHOP CHAIRMAN

James L. Crowley, Professeur INP Grenoble

DATA PREPARATION

Julien Letissier and Nicolas Gourier INRIA Rhône Alpes

PROGRAM COMMITEE

James L. Crowley	INP Grenoble
Dr. Tim Cootes	University of Manchester
Prof. Gerhard Rigoll	TUM - Technical University of Munich
Prof. Eric Granum	Aalborg University
Dr. Sebastien Marcel	IDIAP
Dr. Andreas Lanitis	Cyprus College, Nicosia
James Ferryman	Univ. of Reading

Proceedings edited by :

James L. Crowley
Professeur, I.N.P. Grenoble
INRIA Rhône Alpes
655 Ave de l'Europe
38330 Montbonnot
France

Contents

Pointing 2004 Workshop Proceedings

Contents

Introduction to the Pointing 2004 Workshop
James L. Crowley

The Pointing'04 Data Sets
Julien Letissier, Nicolas Gourier

A Two-level Pose Estimation Framework Using Majority Voting of Gabor Wavelets and Bunch Graph Analysis
Junwen Wu, Jens M. Pedersen, Pew Putthividhya, Daniel Norgaard and Mohan M. Trivedi

Estimating Head Pose with Neural Networks
Rainer Stiefelhagen

Estimating Face Orientation from Robust Detection of Salient Facial Structures
Nicolas Gourier, Daniela Hall and James L. Crowley

Pointing Gesture Visual Recognition for Large Display
Sebastien Carbini, Jean Emmanuel Viallet and Olivier Bernier

Skin Colour Cue for Computer Vision (Invited Talk)
Moritz Störring

Pointing and Command Gestures for Augmented Reality
Thomas B. Moeslund, Moritz Störring, and Erik Granum

HMM and IOHMM for the Recognition of Mono- and Bi-Manual 3D Hand Gestures
Agnès Just, Sébastien Marcel, Olivier Bernier, Jean-Emmanuel Viallet

Introduction to the Pointing 2004 Workshop

James L. Crowley

Pointing 2004 has been organized by the European Thematic Network on Face and Gesture Analysis : FGNet (IST-2000-26434) sponsored by the European Commission's IST Programme. Each year, FGnet published a set of benchmark image sequences on some aspect of visual observation of human activity. Research teams from around the world are invited to test their algorithms on the data set. A workshop is then organized at a major conference to showcase the results. Previous FGnet Workshops have included (PETS '02 at ECCV 02, and PETS '03 at ICVS '03).

The theme for the 2004 FGnet workshop is pointing (or deictic) gestures. The network has prepared video images and sequences of people pointing at targets with their hands and faces. Data has been prepared by projecting a target on a wall using a steerable video projector. Subject were seated in a room at a distance of 3 meters from the wall and asked to point at the target with their right hand and with their face. Video sequences were recorded from four cameras placed at different positions around the subject. Still images were also obtained for each gesture. For each gesture a ground truth was automatically recorded in the form of the known target position. Participants have been invited to compete in recognizing or estimating the target position with four categories of data: Stereo image pairs of hand pointing. Monocular images sequences of hand pointing, monocular images of face pointing. A third data set has been included in which pointing gestures are observed from a hat mounted camera. Two sets of sequences and still image sets have been published on the workshop web site. Ground truth data has been provided for the first set, but withheld from the second. For each image or sequence, participants have been invited to determine: precision of target estimation and probability of failure.

The workshop has been organized with three objectives:
1) To provide encourage comparative evaluation of vision techniques.
2) To encourage the definition and use of new metrics for performance evaluation.
3) To provide a publically available benchmark by which future algorithms may be compared.

The data sets will remain available via the Fgnet web site: http://www-prima.inrialpes.fr/FGnet/.

The Pointing'04 Data Sets

Julien Letissier, Nicolas Gourier

The Pointing '04 Head-Pose Image Database

The head pose database consists of 15 sets of images. Each set contains of 2 series of 93 images of the same person at different poses. The first serie is used for learning, the second is for testing. There are 15 people in the database, wearing glasses or not and having various skin color. The pose, or head orientation is determined by 2 angles (h,v), which varies from -90 degrees to +90 degrees. Here is a sample of a serie :

Figure 1. A Sample from the Face Pose Data Set
(http://www-prima.inrialpes.fr/Pointing04/data-face.html)

Each file tar.gz contains one serie of 93 images and has an approximative size of 1MB. For example, the file **Person05-2.tar.gz** is the second serie of the person number 5. All images are in JPEG format. There are a few examples of series in MPEG format. The file Front.tar.gz consists of 30 frontal images of persons of the database. This serie is useful to learn or to test on frontal images.

Filenames are constructed according the following grammar :
personne[*PersonID*][*Serie*][*Number*][*VerticalAngle*][*HorizontalAngle*].jpg
where :
PersonID = {01, ..., 15}: stands for the number of the person.
Serie = {1, 2} stands for the number of the serie.
Number = {00, 01, ..., 92} the number of the file in the directory.
VerticalAngle = {-90, -60, -30, -15, 0, +15, +30, +60, +90}
HorizontalAngle = {-90, -75, -60, -45, -30, -15, 0, +15, +30, +45, +60, +75, +90}

	Negative values	Positive values
Vertical Angle	Bottom	Top
Horizontal Angle	Left	Right

For example, file **personne08123-30+45.jpg** :
PersonID = 08, *Serie* = 1, *Number* = 23, *VerticalAngle* = -30, *HorizontalAngle* = +45

In case the vertical angle is -90 or +90, the person is looking at the bottom or the top, and then the horizontal angle is 0. Each serie contains therefore 7 x 13 + 2 x 1 = 93 images.

Image Acquisition

All images have been taken using the FAME Platform of the PRIMA Team in INRIA Rhone-Alpes. To obtain different poses, we have put markers in the whole room. Each marker corresponds to a pose (h,v). Post-it are used as markers. The whole set of post-it covers a half-sphere in front of the person.

In order to obtain the face in the center of the image, the person is asked to adjust the chair to see the device in front of him. After this initialization phase, we ask the person to stare successively at 93 post-it notes, without moving his eyes. This second phase just takes a few minutes. All images are obtained by using this method.

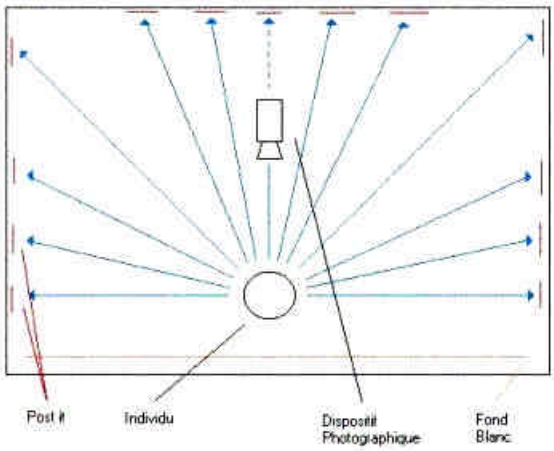


Figure 2a Overhead View

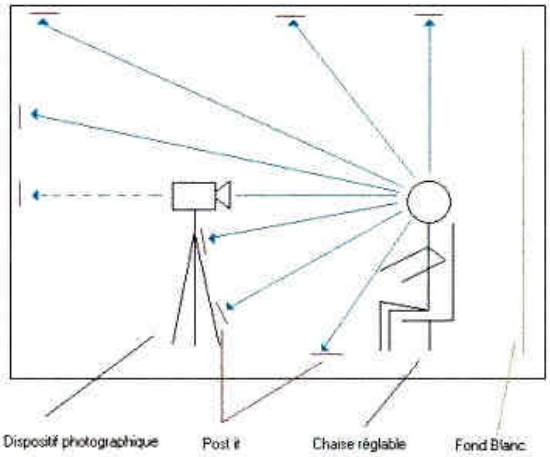


Figure 2b Side View

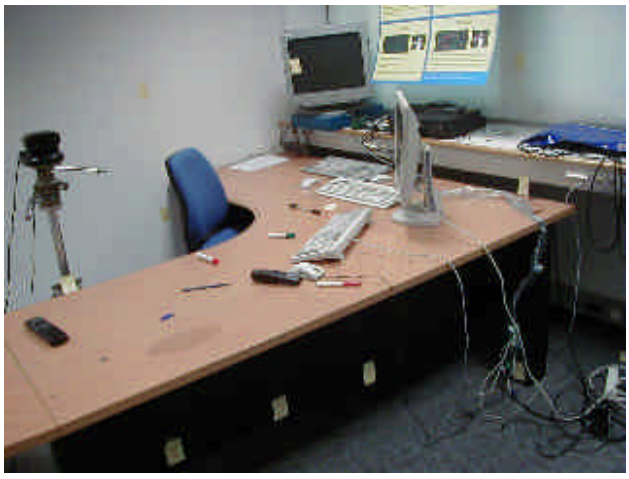


Figure 3. The FAME Augmented Meeting Environment

2. Pointing Gestures: video sequence database

The database consists of 8 video sequences of people successively pointing at different positions on a whiteboard with a finger. Each person is recorded twice, once with a known ground truth of pointed positions, once with a hidden ground truth. The author (Julien Letessier) can be contacted for details not mentioned here.

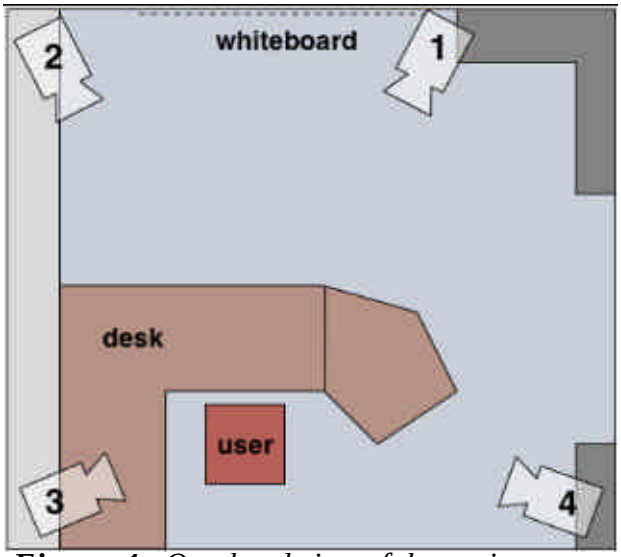


Figure 4a *Overhead view of the environment.*

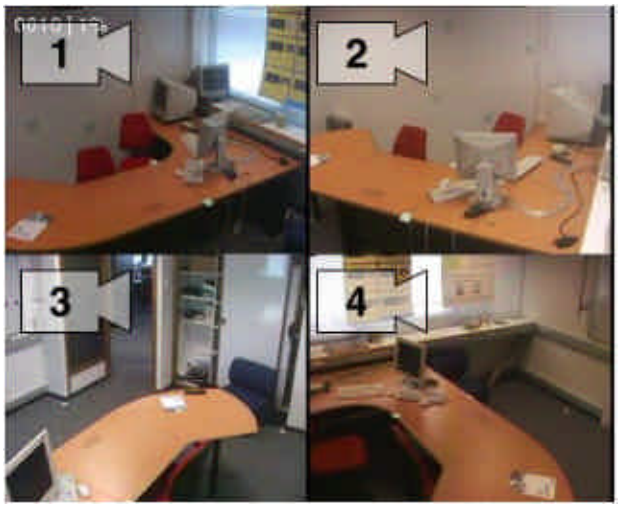


Figure 4b. *Four camera view of the environment*

Capture setup

The sequences are captured in the FAME Augmented Meeting Environment of project PRIMA, laboratory GRAVIR, at INRIA Rhône-Alpes, Grenoble, France. They are taken simultaneously from four ceiling cameras oriented towards the user.

Script

Each capture sequence follows the following script:
the user

1. enters the office and sits;
2. clicks to display a pattern on the desk (for synchronization purposes);
3. successively points 8 different positions on the whiteboard;
4. stands up and exits the office.

Lighting conditions

Scene illumination roughly consists of 60% natural diffuse light and 40% neon light.

Capture

Capture is performed using *video4linux* and *ffmpeg* from a 25Hz, non-interleaved PAL stream at CIF size (352x288 pixels). The four view are synchronized *a posteriori*; the maximum delay between two different view is one frame (40 ms).

data set

All of the data may be gound at http://www-prima.inrialpes.fr/Pointing04/data-hand.html .
Quadriscopic videos of each capture are provided, in an downsampled and compressed format, for reference purposes. These videos are labeled with a frame counter in the upper left corner, and the sequence name in the lower left corner.

The videos are named *_montaged-<a><b>.avi*, where *<a>* is the capture ID of the filmed person and *<b>* is 1 for the sequence where the ground truth is known, and 2 when the ground truth is hidden. These videos were assembled with *ImageMagick-5.5.7* and compressed in *mpeg4* format with *ffmpeg-0.4.8*.

Image sequences

The sequences are provided as a set of tarballs containing *zlib*-compressed PNG format images. The tarballs are named *sequence-<a><b><c>.tar*, where *<c>* is the view ID and *<a>* and *<b>* have the same meaning as above. They contain a directory named *sequence-<a><b><c>*, which itself contains a set of *frame-<n>.png* image files (*<n>* is a four-digit integer).

The sequences are also provided as heavily-compressed *mpeg4* videos, named *sequence-<a><b><c>.avi* (same conventions). The PNG files were generated and compressed from raw PPM files using *ImageMagick*.

Geometric information

Correspondences between points in world coordinates and points in view coordinates are provided in the geometry.txt file (tab-separated format). Column 1 gives the point names, columns 2-5 give the coordinates in each of the views (point *0,0* is in the lower left of a view), and column 6 gives the world coordinates. The office-map.pdf is a top-down map of the office with axes and coordinates (in centimeters).

Ground truth

The ground truth is provided for sequences where *<c>* is 1. In file ground-truth.txt is a list of frame indices where the user points to a known position. This ground truth could be used e.g. for system calibration. The points are the corners of the room's whiteboard and the midpoints of the whiteboard's borders.

A Two-level Pose Estimation Framework Using Majority Voting of Gabor Wavelets and Bunch Graph Analysis

Junwen Wu, Jens M. Pedersen, Duangmanee (Pew) Putthividhya, Daniel Norgaard and Mohan M. Trivedi
Computer Vision and Robotics Research Lab, University of California, San Diego,
La Jolla, CA 92037, USA
{juwu, mejdahl, putthi, norgaard, mtrivedi}@ucsd.edu

Abstract

In this paper a two-level approach for estimating face pose from a single static image is presented. Gabor wavelets are used as the basic features. The objective of the first level is to derive a good estimate of the pose within some uncertainty. The objective of the second level processing is to minimize this uncertainty by analyzing finer structural details captured by the bunch graphs. The first level analysis enables the use of rigid bunch graph. The framework is evaluated with extensive series of experiments. Using only a single level, 90% accuracy (within ±15 degree) and over 98% (within ±30 degree) was achieved on the complete dataset of 1,395 images. Second level classification was evaluated separately for all the poses; the overall accuracy is 58.02%

1 Introduction

Human-computer interaction is an active research topic in computer vision and intelligent systems. The essential aim is to determine human's identity and activity in different environment settings [1-2]. Development of practical systems for intelligent environments can utilize gestures, pointers or the direction in which a person's face is pointed to identify an area of interest [3]. The top right image in Fig.1 illustrates the face-pointing problem. Face pose is determined uniquely by both the pan angel β and the tilt angle α. The top left and bottom two images give some typical application scenarios for face pointing.

Existing pose estimation algorithms can be categorized into one of the following two classes: 3D pose estimation and 2D pose estimation. The difference comes from the input. For 3D pose estimation, in general multiple frame input is available; while for 2D pose estimation, usually the input is static frame. The face-pointing problem we solve in this paper falls into this category. For 3D pose estimation, the input could be subsequent frames from a time sequence [4-6], from which the motion of the face, including scaling, translation and rotation, can be obtained by head tracking. This can be used for a variety of computer vision systems. In our own research we have considered this in the context of an intelligent meeting room [1], intelligent vehicles [7], and wide area surveillance [8]. The input could also be stereo pair of the face images [9]. Correspondences between the stereo pair are established from salient facial features, using which the depth map can be reconstructed. The 3-D coordinates of the salient facial features are estimated hence after to determine the face pose.

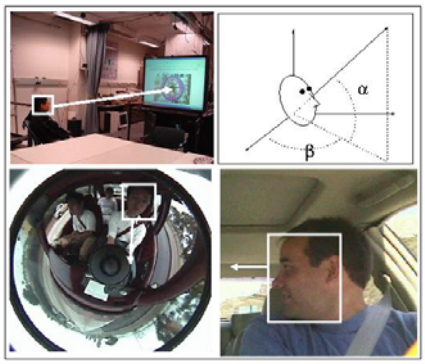


Figure 1. Illustration of face pointing problem and possible applications

The 2D pose estimation problem poses a different challenge. Researchers have put many efforts to investigate the problem [10-12]. However, most efforts are not sufficient for face pointing due to insufficient resolution of the estimation. For face pointing applications, both the pan angle and the tilt angle need to be estimated in a fine scale.

2 Face pose estimation framework

The proposed solution for the face pose estimation is a two-level pose estimation scheme in a coarse-to-fine fashion. The first level is a multi-resolution subspace analysis level, which provides an estimate of the pose based on a holistic analysis of the image. The second level is a structural landmark analysis, where structural information formed by local salient features provides the necessary detail for a finer pose assessment, refines this estimate. Fig.2 outlines this

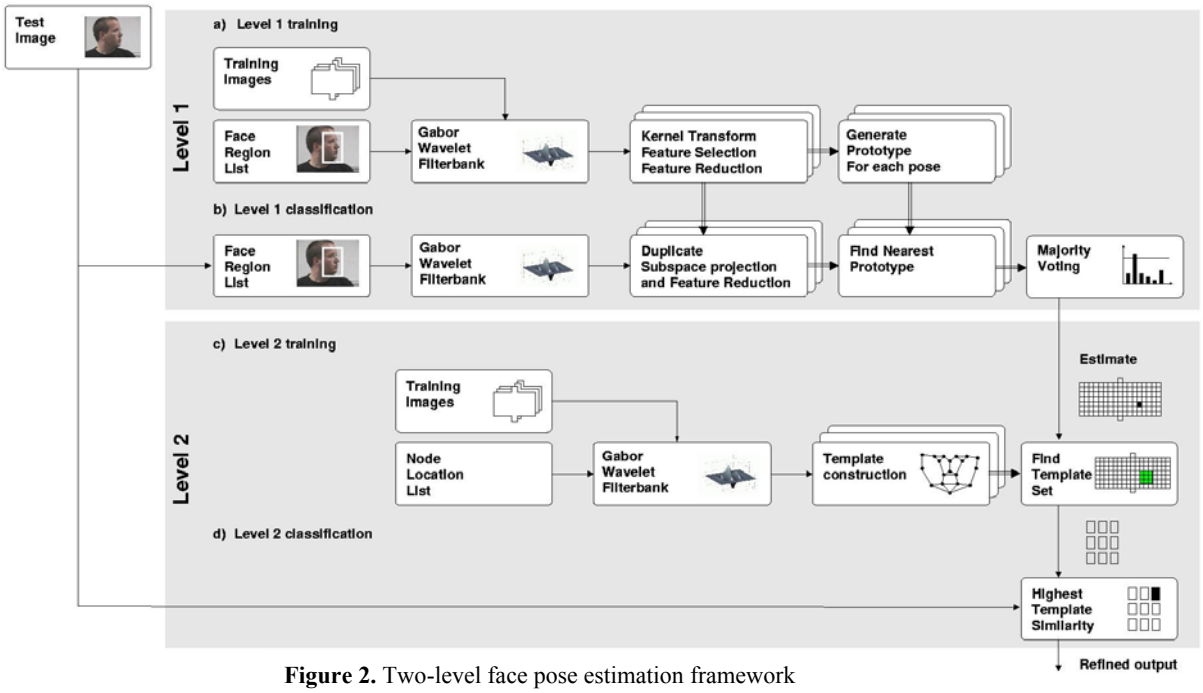


Figure 2. Two-level face pose estimation framework

flow. The two-level approach is based on the rationale that visual cues characterizing facial pose has unique multi-resolution spatial frequency and structural signature.

The multi-resolution subspace analysis level has the objective of deriving pose estimates with some uncertainties. In Fig.2 (a) and (b) the training and classification for the multi-resolution subspace analysis are detailed. The face regions of the training images are transformed to the multiscale spatial-frequency domain by Gabor wavelet transform. These regions are from manually cropping to avoid error from alignment. However, automatic cropping can be realized with face detection algorithms [13-14], followed by alignment or image registration. Nearest prototypes

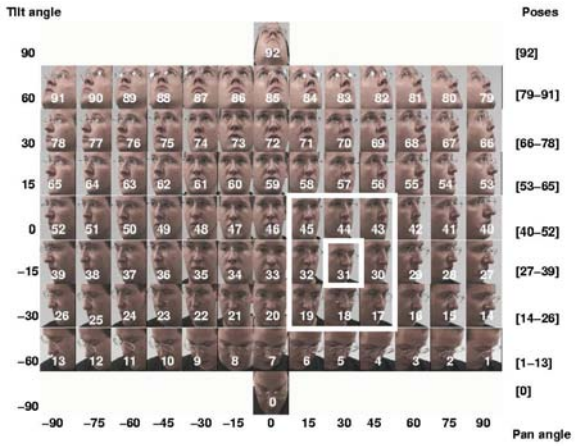


Figure 3. Example of all the poses in FGNet Pointing'04 dataset

in the projected subspaces are used for the first level classification. Kernel Discriminant Analysis (KDA) [15] and Principal Component Analysis (PCA) are used to find the best subspace descriptors. In the training procedure, for each filter response at every pose, a prototype is built by taking the corresponding mean. In the classification procedure, for each corresponding filter response, a class label can be obtained by nearest prototype matching. The class labels from multiple filter banks are fused by majority voting. The multi-resolution subspace classification provides pose estimation with some uncertainties, which is correct up to ±15 degree around the true pose.

To refine the estimate from the multi-resolution subspace analysis, a structural landmark based classification technique is used. A rigid face bunch graph, as proposed in [16-17], is constructed as a template for each pose and a matching scheme based on a normalized correlation score is employed. Bunch graph representation captures the essential geometric configurations of a set of salient facial components .The nodes of the graphs are calculated using Gabor Jets on facial components, which is a vector of magnitudes of Gabor wavelet filter responses of different frequencies and orientations. The classification from the multi-resolution subspace analysis serves to find a subset of the rigid bunch graphs template. By template matching the best pose within the specified subset is found, as sketched in Fig.2(c) and (d).

The data sets used for evaluating this approach are provided by the organizers of the Pointing '04 work-

shop. In the dataset 15 subjects are present and assuming poses in varying orientations. The orientation varies in 15 degree pan angle intervals from profile to profile, -90 to 90 degrees. The tilt angle jumps with 15 degrees for 0,15,30. Then the gap is increased to 30 degrees, jumping to a tilt angle of 60 degrees. Similarly for negative tilt angles. In addition to these 91 poses, two extreme poses with tilt angles of ±90 degrees and a pan angle of 0 degrees are also in the data set, ending up with a total of 93 poses. This is duplicated so two such pose configurations exist for each test subject, namely training and testing set. For simplicity, each pose is labeled as shown in Fig.3.

3. Face pose estimation approach

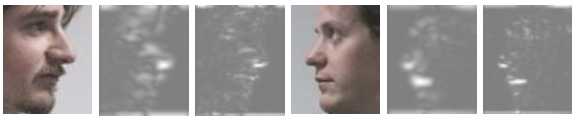
The following sections describe the details of the two-level approach. In section 3.1, the feature extraction algorithm is shown. In section 3.2, the details of the classification strategy are described.

3.1 Multi-resolution feature extraction

Frequency domain analysis techniques have a nice property in extracting the structural features as well as suppressing the undesired variations, such as changes in illumination, changes in person identity, etc. However, frequency domain representation has its own disadvantage: The localization information is lost. Naturally, people will seek a joint spatial frequency representation. Gabor wavelets are one such solution. Gabor wavelets are recognized as being good feature detectors since optimal wavelets can ideally extract the position and orientation of both global and local features [18] as well as preserving frequency information.

3.1.1 Gabor wavelet transform

Gabor wavelet transform is a convolution of the image with a family of Gabor kernels. All Gabor kernels are generated by a *mother wavelet* by dilation and rotation. The mother wavelet is a plane wave generated from a complex exponential and restricted by a Gaussian envelop. In equation (1) - (3), a DC-free mother wavelet is given [16-17]:



(a) (b)

Figure 4. Examples of the Gabor wavelet transform. For both (a) and (b): the leftmost images are the original images; the middle images are the filter responses for the 17th Gabor kernel; the rightmost images are the filter responses for the 33rd Gabor kernel.

$$\psi_{\vec{k}}(\vec{x}) := B(k,x)\left(\exp\left(i\vec{k}\cdot\vec{x}\right) - \exp\left(-\frac{\sigma^2}{2}\right)\right) \quad (1)$$

$$B(k,x) = \frac{k^2}{\sigma^2}\exp\left(-\frac{k^2}{2\sigma^2}x^2\right) \quad (2)$$

$$\left\|\psi_{\vec{k}}(\vec{x})\right\|^2 \sim k^2 \quad (3)$$

where $B(k,x)$ is the Gaussian envelop function restricting the plane wave, $\exp\left(i\vec{k}\cdot\vec{x}\right)$ is the complex-valued plane wave and $\exp\left(-\sigma^2/2\right)$ is the DC-component. The set of Gabor kernel can be given as:

$$\psi_{\vec{k}}(\vec{x}) = k^2 \cdot \psi_{\binom{1}{0}}\left(k\Re(\varphi)\cdot\vec{x}\right), \quad (4)$$

where $\vec{k} = (k,\varphi)$ is the spatial frequency in polar coordinates and $\Re(\varphi) = [\cos\varphi \;\; \sin\varphi; -\sin\varphi \;\; \cos\varphi]$.

DC-free versions of Gabor kernels are of great interests to the researchers in computer vision area due to their invariance property to the uniform background illumination change [16-17]. Only the magnitude of the wavelet transformation is because the phase response is highly sensitive to the non-perfect alignment of the data. In our implementation, for PCA, a family of Gabor kernels with 48 spatial frequencies is used (6 scales and 8 rotations); while for KDA only the first 24 spatial frequencies are used for best performance. We also use the whole 48 filters, however the performance is a little bit worse than that from the first 24, yet still much better than that for PCA (89.7% *v.s.*85.16%). Example of the transformed data is shown in Fig.4.

3.1.2 Feature selection in transformed domain

The wavelets transform representation suffers from high dimensionality. Subspace projection is used to reduce the dimension as well as extracting the most essential information. Two different subspaces are used individually, and their performance is compared. One is the PCA subspace projection, and another is the KDA. The former is a widely used method in subspace feature extraction. It aims to find the subspace that describes most variance while suppress known noise as well as possible. It is computed by first calculating the covariance matrix:

$$\Sigma = \frac{1}{N}\sum_{i=1}^{N}(x_i-\mu)(x_i-\mu)^T, \mu = \frac{1}{N}\sum_{i=1}^{N}x_i \quad (5)$$

Alternately, the principal components is computed by solving the Eigenvalue problem of:

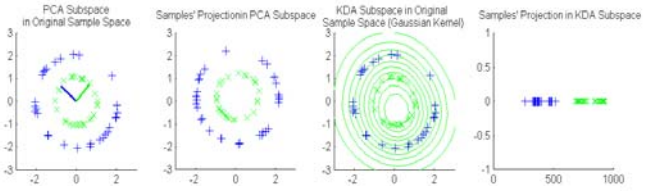


Figure 5. Projection of non-linear data classification problem, with PCA (left) and KDA (right).

$$\Sigma V = V\lambda\,, \tag{6}$$

where $\Lambda = diag(\lambda_1,\cdots,\lambda_D)$ is a diagonal matrix whose elements $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_D$ are the Eigenvalues of Σ. $V = \lfloor v_1, v_2, \cdots, v_D \rfloor$ is the matrix whose columns are the corresponding eigenvectors. Hence, the reduced PCA subspace is formed by the first $M < D$ eigenvectors. However, since PCA is a linear transformation from second order statistics, it is clearly beyond its capabilities to extract the non-linear structure or the higher order statistics of the feature space. Also, the first principals will probably not reveal the most discriminating structure of the underlying class information. This introduces the KDA, which is a non-linear variant of the classical Linear Discriminant Analysis (LDA) [14-[15]. For LDA, it finds the projection according to the Fisher's criterion, which maximize the Rayleigh coefficient:

$$J(w) = \frac{w^T \mathbf{S}_B w}{w^T \mathbf{S}_W w} \tag{7}$$

with respect to *w*. $\mathbf{S}_B$ and $\mathbf{S}_w$ are the between class scatter matrix and within class scatter matrix respectively, which is defined as follows:

$$\mathbf{S}_B = \sum_{k \in K} |I_k| (\mu_k - \mu)(\mu_k - \mu)^T \tag{8}$$

$$\mathbf{S}_W = \sum_{k \in K} \sum_{x \in X_k} (x - \mu_k)(x - \mu_k)^T \tag{9}$$

where μ is the total mean, μ_k is the *k*-th class mean and $|I_k|$ denotes the size of class *k*. KDA is created by projecting the input data non-linearly into a kernel feature space *F*, and within this applying Fisher's criterion, equation (7), thus yielding a non-linear discriminant in the input space. To form the KDA, *w* is expressed in terms of the projected input data *x*:

$$w = \sum_{i=1}^{N} \alpha_i \Phi(x_i) \tag{10}$$

where $\Phi(x_i)$ is the non-linear transform and N is the size of the dataset. Now the definition of the class mean vector μ_k in equation (8) can be rewritten as

$$w^T \mu_k = \frac{1}{N_k} \sum_{i=1}^{N} \sum_{j=1}^{N_k} \alpha_i K(x_i, x_j^k) = \alpha^T \mathbf{S}_M \tag{11}$$

where $K(x_i, x_j^k) = \left\langle \Phi(x_i), \Phi\left(x_j^k\right) \right\rangle$ is the kernel function and N_k is the size of class k. In our application, we use the Gaussian kernel. By defining

$$(\mathbf{S}_M)_{ki} = \frac{1}{N_k} \sum_{j=1}^{N_k} K(x_i, x_j^k) \tag{12}$$

$$\mathbf{S}_N = \sum_{k=1}^{N_k} \mathbf{K}_k (\mathbf{I} - \mathbf{1}_k) \mathbf{K}_k^T \tag{13}$$

where $\mathbf{K}_k$ is a $N \times N_k$ matrix centered kernel matrix of class *k* and $\mathbf{1}_k$ is a matrix with all entries $1/N_k$, the Fisher Discriminant Analysis in the feature space *F* can be rewritten as:

$$J(\alpha) = \frac{\alpha^T \mathbf{S}_M \alpha}{\alpha^T \mathbf{S}_N \alpha} \tag{14}$$

Equivalent to the algorithm in the input space, the maximizing problem is solved by finding the Eigenvalues of $\mathbf{S}_N{}^{-1}\mathbf{S}_M$, and the feature space projection U_A by selecting the first $M < D$ eigenvectors. Hence the KDA projection is obtained by:

$$y = U_A{}^{\mathrm{T}} K_x\,, \tag{15}$$

where $K_x = (K(x, x_1), \cdots, K(x, x_N))$. The projected vectors y in the subspace are the features we use. The non-linear properties can be seen in Fig.5, where both the PCA and KDA feature space representations are illustrated for comparison in a binary nonlinear classification problem. As can be seen the PCA is not able to produce a more discriminating representations due to the non-linearity of the data, whereas the KDA transforms the data into two well-separated clusters.

3.2 Classification

Two-level classification scheme is proposed. In the first level, the pose is estimated with localization ability up to ±15 degree in both pan and tilt, corresponding to a 3×3 neighborhood around the true pose position. Resultantly, in the second level the problem conforms to a 9-class classification problem

instead of a 93-class one. This makes it feasible to use rigid bunch graphs to refine the estimation.

3.2.1 Multi-resolution subspace classification by majority voting

We use the nearest prototype as the basic classifier for the first level classification. For every Gabor wavelet response, class mean in the transformed feature subspace is calculated and used as the prototype. For every Gabor kernel, we can get a basic classifier. Therefore, there are M basic classifiers altogether, for PCA subspace, M=48 while for KDA subspace, M=24. Assuming that all Gabor wavelets are equally important for the pose estimation, we use the majority voting to determine the pose.

The prototype of each class is given by the mean of the training samples in the projected subspace:

$$\mu_{y,k,f} = \frac{1}{N_k}\sum_{i=1}^{N_k} y_{i,f}, \tag{16}$$

where $f = 1,\cdots,M$ and $k = 1,\cdots,93$.

$$d(y,k,f) = \left\| y_f - \mu_{y,k,f} \right\|, \tag{17}$$

$$\ell(y,f) = \arg\min_k d(y,k,f). \tag{18}$$

The classification result is given by:

$$C(y) = \arg\max_c \{\#(\ell(y,f) = c)\} \tag{19}$$

Both the feature set from PCA and KDA are used for the first level classification, for performance comparison.

3.2.2 Structural landmark analysis by bunch graph template matching

The coarse pose estimation is refined in the second level. The use of Gabor filter responses computed from the entire face image poses certain drawbacks to the problem of accurate pose estimation. Due to a small difference between neighboring poses, PCA and KDA might not be able to select the features that best discriminate poses that are strikingly similar. In this section, we present a geometrical structure based approach which exploits accurate localization of salient features on a human face, e.g. pupils, nose tip, corners of mouth, and etc. together with their geometric configuration to aid in pose classification. The motivation behind the use of geometric relationships between salient points on a face lies in an observation that with different degrees of rotation (both in the pan and tilt directions), the relative distances between salient points correspondingly change. In this step, we applied face bunch graph algorithm [16-17] to first accurately locate a predefined set of salient features on a face. For each pose, we construct a template that captures the essential geometric configuration of the salient features. Using a simple template matching scheme, the template that results in the highest similarity score is declared a match.

3.2.2.1 Face representation & Model Graph Generation

The basic object representation applied is labeled graph. In our implementation, we adopt the same representation of face bunch graph as used in [16-17]. A face is represented as a graph with nodes formed by Gabor jets of 5 scales and 8 orientations at salient facial components. The nodes are connected and labeled with distance information. For each pose, a model graph is generated. First, the issue of which salient points on a face to be used as nodes is addressed. In the frontal parallel view case as shown in the leftmost image of Fig.6, 19 nodes are selected. In a more oblique view in the middle and right images of Fig.6, only 13 nodes are used. A face bunch graph is constructed by bundling the graph from each training image together and computing the average relative distance between nodes.

3.2.2.2 Similarity Measurement

Matching between different graphs is done by evaluating the similarity between the ordered Gabor jets [16-17]. Normalized cross correlation is used as the similarity function, where $x_i(f)$ corresponds to the magnitude response of i^{th} node at f^{th} filter and $J_i = (x_i(1),\cdots,x_i(N_f))$ is the Gabor jet for i^{th} nodes:

$$D_x(J_i,J_j) = \frac{\sum_f x_i(f)x_j(f)}{\sqrt{\sum_f x_i^2(f)\sum_f x_j^2(f)}} \tag{20}$$

A graph similarity between an image graph, G^I, and

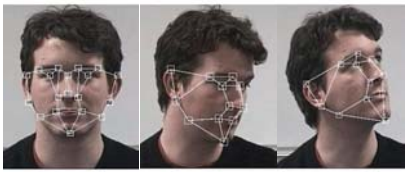


Figure 6. Examples of the face bunch graph

the face bunch graph of a particular pose, B, is computed by searching through the stacked model graph, B^m, for each node to find the best fitting jet in the bundle that maximizes the jet similarity function. The

multi-resolution subspace classification results enable us to confine the graph to be rigid and limit the matching process in a small subset of the graph tem-

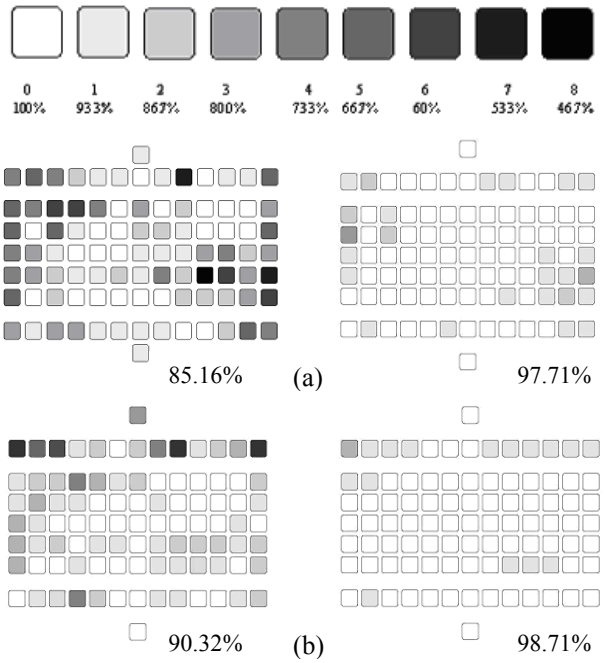


Figure 7. Results evaluated for the first level classification in PCA and KDA subspaces. The middle row (a) gives the error in the PCA subspaces of 48 wavelets. The left figure evaluates the localization ability up to the 3x3 sub-window around the true pose, which corresponds to ±15 degree; the right figure evaluates error on the localization ability up to 5x5 sub-window around the true pose, which corresponds to ±30 degree. The bottom row gives the similar performance evaluation for the KDA subspace (Gaussian) kernel.

plates. The average response over all the nodes N_n is used as the overall graph similarity measure.

$$D_B(G^I, B) = \frac{1}{N_n}\sum_{n=1}^{N_n} \max_m (D_x(J_n^I, J_n^{B_m})) \qquad (21)$$

3.2.2.3 Template Matching

In second-level pose classification, we attempt to refine the classification results from multi-resolution subspace analysis. Multi-resolution subspace analysis gives the estimation with an uncertainty of ±15 degree. It gives the possible 3x3 region (9 poses) that the pose falls in. In the matching stage, the face bunch graph for the test image is constructed and compared with the model face bunch graphs from the 9 poses. The pose with the highest similarity score gives the final pose estimation.

4 Experimental results and analysis

Experimental results from both levels are discussed individually. The integrated performance evaluation is still under investigation. It may also require another dataset with more precise ground-truth.

4.1 Classification in multi-resolution subspace classification

The purpose of the first level is to localize the poses at the accuracy up to the $N \times N$ sub-window around the true pose. The accuracy is evaluated according to this purpose: if the pose estimation falls out of the $N \times N$ sub-window around its true value, it is determined as falsely classified. In our implement N=3 is used. Bigger N gives better accuracy, however, the localization ability is weaker, which will cause more difficulty for the second level refinement. In Fig.7 the errors from PCA (48 filters) and KDA (24 filters) are shown respectively. PCA subspace projection can give us a total accuracy of 85.16%, whereas KDA enhances the performance to 90.61%. To get a better understanding of how these errors distributed, we also evaluated the error on the 5x5 sub-windows, corresponding to ±30 degree uncertainty. The results are shown as the right diagrams in Fig.7. The PCA gives a total accuracy of 97.71%, while KDA gives 98.71%. It shows that only few samples have large estimation deviation from its true value.

4.2 Refinement by structural landmark analysis

The refinement step works on classifying poses in a window of 3x3 neighboring poses as seen in Fig.8. The accuracy (out of 15 testing images) for each pose is summarized in Table 1. Pose 0 and pose 92 is not considered in the second level refinement because these twp poses can be determined without uncertainty from the multi-resolution subspace classifica-

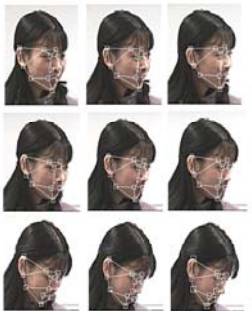


Figure 8. Examples of face bunch graphs in the 3x3 neighboring poses to be examined in the second level.

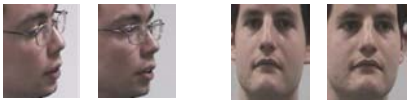


Figure 9. Example of the ambiguity in data

tion. The overall accuracy for 91 poses is 58.02%, while the accuracy for each individual pose has a large variance from 20% to 93.3%. The results show that face bunch graph template matching is a promising candidate for the structural landmark refinement. However, the final performance needs to be evaluated by integrating multi-resolution subspace analysis and structural landmark analysis together. This work is still under investigation. The number of nodes used in the template graphs ranges from 13 to 19. For near frontal poses where more salient facial features are visible, more nodes are selected, while for poses with oblique views a fewer number of nodes is used. After analyzing some mis-classification results, we have concluded that some errors arise from the use of templates with inadequate structural details. This problem could potentially be fixed by adding more nodes and edges. Several misclassification errors result from the inherent ambiguity in both the training and the testing images. As seen in the Fig.9, in the left-most image pair, these two poses are supposed to be different by 15 degrees in the pan and tilt direction, which obviously is not the case. Again in the right image pair, the 15-degree angle difference is not apparent. Fig.10 shows a diagram of error distribution over all poses. It is observed that the best accuracy appears at the near frontal positions where more nodes are used to construct the model graphs, hence more discriminating template structures. The close-to-extreme positions also yield high accuracy classification. This outcome is expected since fewer ambiguities are present for those poses.

5 Conclusion and discussions

In this paper we discussed a two-level approach for estimating face pose from a single static image. The rationale for this approach is the observation that visual cues characterizing facial pose has unique multi-resolution spatial frequency and structural signatures. For effective extraction of such signatures, we use Gabor wavelets transform as basic features. For systematic analysis of the finer structural details associated with facial features, we employ rigid bunch graphs. The first level of the approach has the objective of confining the estimation into a smaller range; therefore rigid bunch graph is sufficient in the second level refinement. Bunch graph exploits the structural details in the facial features, which makes it capable for pose location refinement Extensive series

Table 1. Second-level refinement

#	%	#	%	#	%	#	%
0	---	24	46.7	48	80.0	72	66.7
1	40.0	25	46.7	49	73.3	73	93.3
2	46.7	26	40.0	50	67.7	74	66.7
3	33.3	27	53.3	51	60.0	75	80.0
4	33.3	28	66.7	52	60.0	76	46.7
5	66.7	29	53.3	53	33.3	77	53.3
6	73.3	30	53.3	54	66.7	78	73.3
7	46.7	31	60.0	55	60.0	79	26.7
8	73.3	32	46.7	56	26.7	80	73.3
9	46.7	33	60.0	57	66.7	81	53.3
10	20.0	34	66.7	58	66.7	82	67.7
11	26.7	35	60.0	59	60.0	83	53.3
12	46.7	36	40.0	60	73.3	84	86.7
13	26.7	37	33.3	61	60.0	85	73.3
14	53.3	38	46.7	62	40.0	86	86.7
15	40.0	39	53.3	63	46.7	87	40.0
16	80.0	40	60.0	64	26.7	88	46.7
17	66.7	41	86.7	65	40.0	89	33.3
18	73.3	42	66.7	66	66.7	90	60.0
19	66.7	43	86.7	67	60.0	91	67.7
20	53.3	44	80.0	68	73.3	92	----
21	73.3	45	86.7	69	73.3		
22	53.3	46	66.7	70	60.0		
23	53.3	47	66.7	71	86.7		

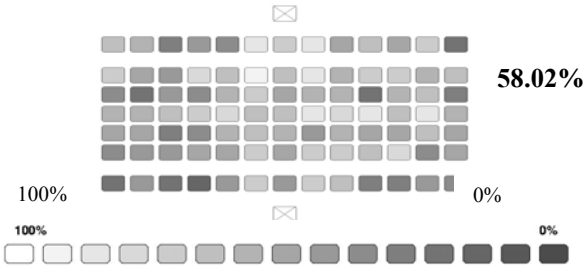


Figure 10. Error distribution for structural landmark analysis classification. The legend is different from Fig.7 to make it more visible.

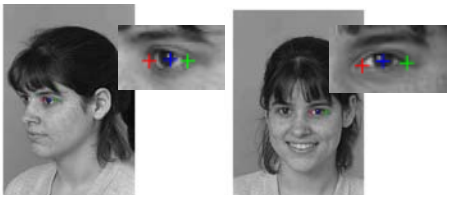


Figure 11. Preliminary results about facial landmark based face region registration. The markers in different colors show the detected predefined landmarks. Image data are from FERET database [20].

of experiments were conducted to evaluate the pose estimation approach. Using only a single level, 90% accuracy (within ±15 degree) was achieved on the complete dataset of 1,395 images. Second level classification was evaluated for the subsets of poses with a total accuracy of 58.02%, without any uncertainty. This result might benefit from using more nodes, and is therefore identified as a future research area. Having verified the basic efficacy of the proposed ap-

proach, further research for improving the computational performance and for evaluation using data sets with more precise ground truth information is desired. Currently automated face alignment algorithm based on face landmark detection is under investigation and some preliminary results give promising results, as shown in Fig.11.

Acknowledgements

Our research was supported in part by grants from the UC Discovery Program and the TSWG of the US Department of Defense. We are thankful for the guidance of and interactions with our colleagues Dr. Doug Fidaleo, Joel McCall and Kohsia Huang from the CVRR Laboratory. We thank the organizers of the Pointing'04 Workshop and the PRIMA group of INRIA for providing the dataset used in our evaluation. Portions of the research in this paper use the Color FERET database of facial images collected under the FERET program.

References

[1]. K. Huang and M. M. Trivedi, Video arrays for real-time tracking of person, head, and face in an intelligent room. *Machine Vision and Applications*, vol. 14, no. 2, pp. 103-111, June 2003.

[2]. K. Huang and M. M. Trivedi, Robust Real-Time Detection, Tracking, and Pose Estimation of Faces in Video Streams, In *Proceedings of International Conference on Pattern Recognition* 2004, (*to appear*).

[3]. Crowley, J., Coutaz, J., Berard, F., *Things That See*. *Communications of the ACM*, Mar. '00, p. 54-64

[4]. K. Nickel and R. Stiefelhagen. Pointing gesture recognition based on 3D-tracking of face, hands and head orientation. *Proceedings of the 5th international conference on Multimodal interfaces.* 2003

[5]. K Seo, I. Cohen, S. You and U. Neumann. Face pose estimation system by combining hybrid ICA-SVM learning and re-registration, *Proc. of Asian Conference on Computer Vision*, 27-30 2004.

[6]. M. La Cascia, S. Sclaroff, and V. Athitsos. Fast, reliable head tracking under varying illumination: An approach based on registration of texture-mapped 3D models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(6):322--336, 2000

[7]. K. Huang, M. M. Trivedi, Distributed Video Arrays for Tracking, Human Identification, and Activity Analysis, *Proc. of the 4th IEEE Int. Conference on Multimedia and Expo,* pp. 9-12, 2003

[8]. K. Huang, M. M. Trivedi, T. Gandhi, "Driver's View and Vehicle Surround Estimation using Omnidirectional Video Stream" *IEEE Intelligent Vehicles Symposium,* p.444-449, Jun 9-11, '03

[9]. M. Xu and T. Akatsuka. Detecting head pose from stereo image sequences for active face recognition. In *Proceedings of Int. Conf. on Automatic Face and Gesture Recog.,* p.82-87, Apr. 1998

[10]. S. Z. Li, H. Zhang, X. Peng, X. Hou and Q. Cheng Multi-View Face Pose Estimation Based on Supervised ISA Learning. In *Proc. of 5th IEEE Int. Conference on Automatic Face and Gesture Recognition*, May 20 - 21, 2002

[11]. S. Gong, S. McKenna, and J.J. Collins. An investigation into face pose distributions. In *Proceedings of IEEE Int. Conference on Automatic Face and Gesture Recognition*, p. 265-270, Oct. 96

[12]. L. Chen, L. Zhang, Y. Hu, M. Li, H. Zhang, Head Pose Estimation Using Fisher Manifold Learning, in *Proceeding of IEEE International Workshop on Analysis and Modeling of Faces and Gestures*, in conjunction with ICCV-2003, 2003

[13]. P.Viola and M.Jones. Robust real-time object detection. In *ICCV 2001. Workshop on Statistical and Computation Theories of Vision.* Volume 2, 2001.

[14]. S. Mika, G. Rätsch, J. Weston, B. Schölkopf and K. Müller, Fisher Discriminant Analysis with Kernels, *Proc. of IEEE Neural Networks for Signal Processing Workshop*, 8 pages, 1999

[15]. Y Li, S Gong and H Liddell. Recognising Trajectories of Facial Identities Using Kernel Discriminant Analysis, In *Proceedings of The British Machine Vision Conference*, 2001

[16]. M. Potzsch, N. Kruger and C. von der Malsburg. Determination of face position and pose with a learned representation based on labeled graphs. Institut for Neuroinformatik. *Internal Report,* RuhrUniversitat, Bochum, 1996.

[17]. L. Wiskott, J. Fellous, N. Krüger and C von der Malsburg. Face Recognition by Elastic Bunch Graph Matching. In *Proc. of the 7th Int. Conference on Computer Analysis of Images and Patterns,* CAIP'97

[18].B. J.MacLennan. Gabor representations ofspatiotemporal visual images, *Technical Report CS-91-144*, Computer Science Department, University of Tennessee, Knoxville. Accessible via URL: http://www.cs.utk.edu/~mclennan., 1991.

[19]. R.-L. Hsu, M.~Abdel-Mottaleb, and A.K. Jain. Face detection in color images. In *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* . 24(5):696--706, May. 2002

[20]. P. J. Phillips and H. Moon and S. A. Rizvi and P. J. Rauss, "The FERET Evaluation Methodology for Face Recognition Algorithms," IEEE Trans. Pattern Analysis and Machine Intelligence, Volume 22, October 2000, pp. 1090-1104.

Estimating Head Pose with Neural Networks - Results on the Pointing04 ICPR Workshop Evaluation Data

Rainer Stiefelhagen
Interactive Systems Laboratories
Universität Karlsruhe (TH)
Germany
E-mail: stiefel@ira.uka.de

Abstract

In this paper we report the results of a neural network based approach to head pose esimtation on the evaluation data set provided for the Pointing04 ICPR workshop. In the presented approach, we use neural networks to estimate a person's horizontal and vertical head orientation from facial images, which automatically were extracted from the provided data set.

With our approach, we achieved an average estimation error of 9.5 degrees for pan and 9.7 degrees for tilt estimation with a multi-user system that was trained on images from all 15 people in the database..

1 Introduction

In recent years many researchers have addressed the problem of vision-based estimation of a person's head orientation (or head "pose"). Related work can be categorized in two approaches: model based approaches and appearance based approaches: In model-based approaches, usually a number of facial features, such as eyes, nostrils, lip-corners, have to be located. Knowing the relative positions of these facial features, the head pose can be computed [2, 9, 3]. Detecting the facial features, however, is a challenging problem and tracking is likely to fail. Appearance based approaches either use some kind of function approximation technique such as neural networks [1, 6, 5], or a face database [4] to encode example images. With appearance based approaches no facial landmark detection is needed, instead the whole image of the face is used for classification.

In the Interactive Systems Lab, we have investigated both approaches. We employed purely neural network [6, 10] and model-based approaches to estimate a user's head pose [9].

In our work we found that robust head pose estimation results could be acchieved using an appearance based approach, where head pose is estimated from facial images using neural networks. This approach has proven to work well on high-resolution facial images as well as low resolution facial images captured with omnidirectional cameras[11, 7].

In this work we report the results of the neural network based approach to head pose esimtation on the evaluation data provided for the Pointing04 ICPR workshop.

2 Estimating Head Pose Using Neural Nets

A major advantage of using neural networks to estimate head pose as compared to using a model based approach is its robustness: With model based approaches to head pose estimation [2, 9, 3], head pose is computed by finding correspondences between facial landmarks points (such as eyes, nostrils, lip corners) in the image and their respective locations in a head model. Therefore these approaches rely on tracking a minimum number of facial landmark points in the image correctly, which is a difficult task and likely to fail. On the other hand, the neural network-based approach doesn't require tracking detailed facial features because the whole facial region is used for estimating the user's head pose. This also allow for head pose estimation on facial images of low resolution.

In our approach we are using neural networks to estimate pan and tilt of a person's head, given preprocessed facial images as input to the neural net. This approach is similar to the approach described by Schiele and Waibel [6].

However, the system described in [6] estimated only head rotation in pan direction. In this research we use neural network to estimate head rotation in both pan and tilt directions. Rae and Ritter [5] describe a user dependent neural network based system to estimate pan and tilt of a person. In their approach, color segmentation, ellipse fitting and Gabor-filtering on a segmented face are used for preprocessing. They report an average accuracy of 9 degrees for pan and 7 degrees for tilt for one user with a user dependent system.

In our previous work we have reported head pose estimation results on good resolution facial images, captured with a pan-tilt zoom camera [8, 10] as well as on low resolution images captured with an omnidirectional camera [11, 7]. Figure 1 shows some sample images used in our previous work. On both good resolution images and low resolution images, head pose estimation results with average estimation errors of less than 4 degrees for pan and tilt for multi-user systems (12 users) were achieved. For new users, av-

a) b) c) d)

Figure 1. Sample images used in our previous work on head pose estimation. Images c) and d) were captured with an omnidirectional camera [11].

Figure 2. Sample images from the Pointing'04 head pose data base.

erage errors of less than 10 degrees for pan and tilt were achieved.

In the remainder of this section, we present our approach in detail and report the head pose estimation results on the data provided for the Pointing04 ICPR workshop.

2.1 The Pointing04 Workshop Head Pose Data Base

The database used for this evaluation consists of 15 sets of images. Each set contains of 2 series of 93 images of the same person at different poses. The first serie is used for learning, the second is for testing. There are 15 people in the database, wearing glasses or not and having various skin color. The pose, or head orientation is determined by 2 angles (h,v), which varies from -90 degrees to +90 degrees. A sample of a serie is depicted in Figure 2.

The images for this database have all been collected within the FAME project by the PRIMA Team in INRIA Rhone-Alpes. To obtain different poses, markers were put in the whole room, which correspond to certain poses (h,v) and at which the subjects had to look during data acquisition. Further details about the data set and the acquisition of the data can be found on the workshop website.

2.2 Preprocessing of Images

To locate and extract the faces from the collected images, we use a statistical skin color model [12]. The largest skin colored region in the input image is selected as the face.

We have investigated two different image preprocessing methods as input to the neural nets for pose estimation [8]: 1) Using normalized grayscale images of the user's face as input and 2) applying edge detection to the images before feeding them into the nets.

In the first preprocessing approach, histogram normalization is applied to the grayscale face images as a means towards normalizing against different lighting conditions. No additional feature extraction is performed. The normalized grayscale images are down-sampled to a fixed size of 20x30 pixels and then are used as input to the nets. In the second approach, a horizontal and a vertical edge operator plus thresholding is applied to the facial grayscale images. The resulting edge images are down-sampled to 20x30 pixels and are both used as input to the neural nets.

Since in our previous work we obtained the best results when combining the histogram normalized and the edge images as input to the neural nets , we are only presenting results using this combination of preprocessed images as input to the neural net here. Figure 3 shows the preprocessed images of a user's faces.

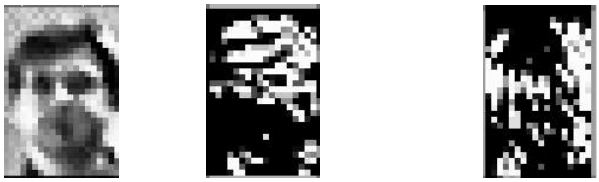


Figure 3. Preprocessed images: normalized grayscale, horizontal edge and vertical edge image (from left to right)

2.3 Neural Net Architecture, Training and Results

We have trained separate nets to estimate head pan and tilt. For each net, a multi-layer perceptron architecture with

one output unit (for pan or tilt), one hidden layer with 20 to 80 hidden units and an input retina of 20x90 units for the three input images of size 20x30 pixels. Output activations for pan and tilt were normalized to vary between zero and one. Training of the neural net was done using standard back-propagation.

2.3.1 Results with Multi-User System

To train a multi-user neural network, we divided the data set of the 15 users into a training set consisting of 2232 images (80% of the data), a cross-evaluation set of size 279 images (10%) and a test set with a size of 279 images (10%). After training, we achieved a mean error of 10.6 degrees for pan and 10.4 degrees for tilt on the multi-user test set.

	pan	tilt
basic set of training images	10.6	10.4
+ additional mirrored images used	9.5	9.7

Table 1. Average error in degrees for pan and tilt estimation on the Pointing04 ICPR workshop data.

In order to obtain additional training data, we have artificially mirrored all of the images in the training set (as well as the labels for head pan, of course). As a result, the available amount of data could be doubled without having the effort of additional data collection.

After training with the additional data, we achieved an average error of 9.5 degrees for pan and 9.7 degrees for tilt on the multi-user test set.

Table 2 summarizes the pan and tilt estimation results on the Pointing04 ICPR workshop data.

It has to be noted that the face orientation data used here consists of faces collected at orientations of 15 degree steps for horizontal rotation, and 30 degree steps for vertical rotation. The task of head estimation head orientation could therefore be treated as a classification problem, where the correct orientation class for pan and tilt orientation has to be detected from an input image.

Instead of treating the face orientation task as a classification problem, however, we have decided to estimate the absolute head orientation directly, as in our previous work.

To obtain a classification measurement of our approach, we have mapped the obtained head pose estimations to the head orientation classes provided in the data. By doing this we achieved a classification accuracy of 52% for horizontal orientation (13 classes) and 66.3% for vertical orientation (7 classes). The corresponding confusion matrices are depicted below.

3 Conclusion

In this paper we have reported head pose estimation results on the Pointing04 workshop evaluation set for facial orientation. We have used a neural network based approach to estimate head pose from facial input images. On a randomly chosen test set containing 10% of the images from the provided evaluation data, we achieved head pose estimation with an average error of 9.5 degrees for horizontal head rotation (pan) and 9.7 degrees for vertical head rotation (tilt). This corresponds to correct classification rates of 52% for classifying the correct horizontal head orientation class (1 out of 13 possible classes) and 66.3% for classifying the correct vertical head rotation class (1 out of 7 possible classes).

	-90	-60	-30	0	30	60	90	sum
-90	1	2	0	0	0	0	0	3
-60	0	24	9	0	0	0	0	33
-30	0	5	26	6	0	0	0	37
0	0	1	26	80	30	0	0	137
30	0	0	0	5	24	4	0	33
60	0	0	0	1	4	29	0	34
90	0	0	0	0	0	1	1	2
	1	32	61	92	58	34	1	279

Table 3. Confusion matrix for classifying vertical head orientation. Correct classification was achieved 66.3% of the time.

Acknowledgments

This work has partially been funded by the European Commission under contract nr. 506909 within the project CHIL (http://chil.server.de).

References

[1] D. Beymer, A. Shashua, and T. Poggio. Example-based image analysis and synthesis. In *Proceedings of Siggraph'94*, 1994.

[2] A. H. Gee and R. Cipolla. Non-intrusive gaze tracking for human-computer interaction. In *Proc. Mechatronics and Machine Vision in Practise*, pages 112–117, 1994.

[3] T. Jebara and A. Pentland. Parametrized structure from motion for 3d adaptive feedback tracking of faces. In *Proceedings of Computer Vision and Pattern Recognition*, 1997.

[4] A. Pentland, B. Moghaddam, and T. Starner. View-based and modular eigenspaces for face recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1994.

[5] R. Rae and H. J. Ritter. Recognition of human head orientation based on artificial neural networks. *IEEE Transactions on neural networks*, 9(2):257–265, March 1998.

[6] B. Schiele and A. Waibel. Gaze tracking based on face-color. In *International Workshop on Automatic Face- and Gesture-Recognition*, pages 344–348, 1995.

[7] R. Stiefelhagen. Tracking focus of attention in meetings. In *International Conference on Multimodal Interfaces*, pages 273–280, Pittsburgh, PA, October 2002. IEEE.

[8] R. Stiefelhagen, M. Finke, J. Yang, and A. Waibel. From gaze to focus of attention. In M. Turk, editor, *Proceedings of Workshop on Perceptual User Interfaces: PUI 98*, pages 25–30, San Francisco, CA, November, 4-6th 1998.

[9] R. Stiefelhagen, J. Yang, and A. Waibel. A model-based gaze tracking system. In *Proceedings of IEEE International Joint Symposia on Intelligence and Systems*, pages 304 – 310, 1996.

	-90	-75	-60	-45	-30	-15	0	15	30	45	60	75	90	sum
-90	0	1	2	0	0	0	0	0	0	0	0	0	0	3
-75	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-60	0	1	15	9	8	0	0	0	0	0	0	0	0	33
-45	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-30	0	0	0	9	21	4	3	0	0	0	0	0	0	37
-15	0	0	0	2	6	24	12	1	0	0	0	0	0	45
0	0	0	0	0	1	11	23	15	0	0	0	0	0	50
15	0	0	0	0	0	1	7	23	11	0	0	0	0	42
30	0	0	0	0	0	1	0	6	18	8	0	0	0	33
45	0	0	0	0	0	0	0	0	0	0	0	0	0	0
60	0	0	0	0	0	0	1	0	2	7	21	3	0	34
75	0	0	0	0	0	0	0	0	0	0	0	0	0	0
90	0	0	0	0	0	0	0	0	0	0	1	1	0	2
	0	2	17	20	36	41	46	45	31	15	22	4	0	279

Table 2. Confusion matrix for classifying horizontal head orientation. Absolute counts are given. Correct classification was achieved 52.0% of the time.

[10] R. Stiefelhagen, J. Yang, and A. Waibel. Modeling focus of attention for meeting indexing. In *Proceedings of ACM Multimedia '99*, pages 3–10. ACM, 1999.

[11] R. Stiefelhagen, J. Yang, and A. Waibel. Simultaneous tracking of head poses in a panoramic view. In *International Conference on Pattern Recognition*, volume 3, pages 726–729, September 2000.

[12] J. Yang and A. Waibel. A real-time face tracker. In *Proceedings of WACV*, pages 142–147, 1996.

Estimating Face orientation from Robust Detection of Salient Facial Structures

Nicolas Gourier Daniela Hall James L. Crowley
Prima Project, GRAVIR Laboratory, INRIA Rhône-Alpes, Montbonnot, FRANCE

Abstract

This paper addresses the problem of estimating face orientation from automatic detection of salient facial structures using learned robust features. Face imagettes are detected using color and described using a weighted sum of locally normalized Gaussian receptive fields. Robust face features are learned by clustering the Gaussian derivative responses within a training set face imagettes. The most reliable clusters are identified and used as features for detecting salient facial structures. We have found that a single cluster is sufficient to provide a detector for salient facial structures that is robust to face orientation, illumination and identity. We describe how clusters are learned and which facial structures are detected. We show use of this detection to estimate facial orientation.

1. Introduction

We are interested in automatically determining which facial structures can be most reliably detected under variations in illumination, position, orientation and human identity. Our objective is to obtain a set of facial structures that can serve as landmarks for tracking and recognition of facial expressions. We employ a fast, pixel level, detection algorithm to isolate and normalize the face regions. Normalized face images are described by calculating a vector of scale-normalized Gaussian derivatives at each pixel. Salient facial structures are detected using linear combinations of these descriptors. Such functions are learned using K-means clustering of the Gaussian derivative responses obtained from a set of training images. The resulting clusters specify linear combinations of Gaussian derivatives that act as detection functions for facial features that remain salient under variations in pose, illumination and identity.

2. Approaches to Facial Structure Detection

Facial structure detection may be performed using global or local features. A popular method for global analysis of face images is to project a normalized image into a linear subspace determined using a technique such as principal components analysis (PCA) [12]. However, projection highly sensitive to small changes in face position and image scale, as well as partial occlusions and as a result has proved unusable in real systems. In general, global techniques such as projection to a principle components space tend to be sensitive to partial occlusions as well as changes in identity.

An alternative is to measure the relative position of salient anatomical facial structures such as eyes and lips [2]. The challenge is that such facial structures are difficult to detect in a general manner. Most authors rely on complex adhoc operations that tend to be highly sensitive to environmental conditions.

We define salient features as features that draw attention. Features isolated in a dense feature space are salient features [18]. Determining such local feature points can be performed by partitionning the face image into several regions, by using textons as in [8] or finding generic features [4, 10, 11]. Facial structures detection can be done using eigenfeatures [15], blobs [16] or saddle points and maxima of the luminance distribution [17]. But such descriptors are sensitive to illumination and provide an overabundance of points, which can lead to accumulation of errors. Interest points are not robust to pose, and are not well adapted to deformable objects such as the human face.

Our objective is to design descriptors that are robust to illumination, scale and orientation. Such a description can be obtained using Gaussian Derivatives, as well as Gabor Wavelets to describe the appearance of each local neighborhood.

Gabor wavelets provide a very general description function as presented in [3], [14], [7] and [9]. Unfortunately, normalized Gabor wavelets tend to be very expensive to compute.

Similar information can be obtained from a vector of Gaussian derivatives, with the advantage the very fast techniques exist for computing scale normalized Gausian derivatives [13]. We employ such a description to compose a detection function for salient facial features that is invariant to scale, orientation and illumination intensity.

Our approach is composed of several steps. First we employ a robust face tracker to detect and normalize the image

of the face. This step, described in section 3, provides a substantial reduction in computation time. Scale normalized Gaussian Derivatives are then computed using a fast pyramid algorithm [13]. Weighted sums of Gaussian derivatives are then used to detect pixels that correspond to salient face regions. The weighting functions are learned by a process that selects combinations of Gaussian derivatives that correspond to the regions that can be detected in the faces of a maximum number of individuals see from a maximum number of viewing directions. This learning process is described in section 4. Face orientation is estimated from the relative positions of the salient regions, as described in section 5. Experimental results using the Pointing '04 face data base are provided in Section 6.

3. Face Image Normalization

We employ a robust video rate face tracker to provide an initial detection and normalization of a face region to a face imagette. Our tracker uses pixel level detection of skin colored regions using a Bayesian estimation of the probability that a pixel corresponds to skin based on its chrominance [6]. This process is described in this section.

3.1. Pixel Level Detection and Tracking using Skin Chrominance

To detect the face, we first detect skin regions in the image using a probabilistic detection of skin chrominance. We compute chrominance by normalizing the red and green components of the RGB color vector by the intensity (R+G+B). Normalizing intensity removes the variations due to angle between the local surface normal and the illumination source. Photons reflected from skin will exhibit a precise value of (r,g) that is determined by the skin pigment and the illumination spectrum. The conditional probability densities for the (r,g) vector for skin regions and for all the image can easily be estimated by histograms. Bayes rule shows that the ratio of these histograms provides a lookup table that maps (r,g) to the conditional probability of skin $p(Pixel \in Skin|r,g)$.

$$p(Pix \in Skin|r,g) = \frac{p(r,g|Pix \in Skin)p(Pix \in Skin)}{p(r,g)} \tag{1}$$

Face position and surface extent are estimated using moments and tracked using a zeroth order Kalman Filter. The tracking process provides a region of interest (ROI) that permits processing to be focused on the face region. Tracking reduces computational cost and improves resistance to distraction by background clutter.

In each image, the skin probability image is calculated within the predicted ROI by table lookup as described above. Pixels within the ROI are then multiplied by a Gaussian predicted by tracking. This step, inspired by robust statistical techniques, improves robustness to background clutter [6].

Both the tracking process and face normalization are based on moments. The first moment (center of gravity) provides a robust estimate of face position, while the second moment provides a measure of the width, height and slant of the face. The first and second moments of the face are used to normalize the face position and orientation, as well as the size of the imagette that represents the face.

We estimate first and second moments with the following formulas (2):

$$\begin{aligned}
\mu_x &= \tfrac{1}{S}\textstyle\sum p_{skin}(x,y)\cdot x\cdot G(x,y,\vec{\mu},C),\\
\mu_y &= \tfrac{1}{S}\textstyle\sum p_{skin}(x,y)\cdot y\cdot G(x,y,\vec{\mu},C),\\
\sigma_x^2 &= \tfrac{1}{S}\textstyle\sum p_{skin}(x,y)(x-\mu_x)^2 G(x,y,\vec{\mu},C),\\
\sigma_y^2 &= \tfrac{1}{S}\textstyle\sum p_{skin}(x,y)(y-\mu_y)^2 G(x,y,\vec{\mu},C),\\
\sigma_{xy} &= \tfrac{1}{S}\textstyle\sum p_{skin}(x,y)(y-\mu_y)(x-\mu_x) G(x,y,\vec{\mu},C),
\end{aligned} \tag{2}$$

where $S = \sum p_{skin}(x,y)\cdot G(x,y,\vec{\mu},C)$

3.2. Performance of the Face Tracker

To initialize our face tracker, we employ either the user's selection on the frame, or a generic ratio histogram. The choice of the number of histogram cells used to form the lookup table for skin detection is an important parameter. Histograms with too few cells will not properly discriminate skin from similar colored surfaces such as wood. Inversely, using too many cells renders the process overly sensitive to minor variations in illumination spectrum as well as skin blemishes. We have empirically observed that (r,g) histograms on the order of ranges 32x32 cells provides a good compromise for face detection. A more thorough analysis is provided by [1].

The face tracker has been carefully optimized to run at real time, and can process 384x288 pixel images at video-rate on a 800 MHz Pentium processor. Eye detection rate on representative video sequences can be seen in table 1 and Figure 1. In this case, an error occurs when the computed ellipse does not contain an eye visible in the image.

An important property for a face tracker is jitter. Jitter is measured as the square of the difference in position and size of the detected pixels of the face when the subject is not moving. We have calculated the variance of the moments of the position and size of the detected face region

Table 1. *Eye detection rate*

Sequence	Number of images	Eye Detection rate
A	500	99,9 %
B	700	99,8 %
C	580	94,2 %
D	300	93,1 %

A : Head slow translation
B : Head fast translation
C : Head zoom and inclination in the plane
D : Head pitch and yaw

Table 2. *Stability of the position and the size of the detected face*

Pose	Front	Half-profile	Profile
X Center	0,31 %	1,13 %	3,23 %
Y Center	0,64 %	1,05 %	1,58 %
Width	0,55 %	1,08 %	1,38 %
Height	0,64 %	1,14 %	1,38 %

on sequences of 20 seconds taken when the subject's head has a certain pose and is not moving. Results are shown in Table 2. We observe that many of the errors occur when the subject is in profile. In this case, detection of the neck can modify the detected region.

Figure 1. Example of face tracking. First and second moments provides an ellipse which delimits the face in the image

3.3. Normalized Face Imagette

The process described above provides a gray scale (intensity) imagette of the face that is normalized in position and size. Intensity, computed as sum of R+G+B, provides stable salient features based on facial structures. Normalizing the moments of the face imagette allows us to restrict processing to a fixed set of positions and scales, thus reducing computation time, as well as providing a fixed number of operations for each face.

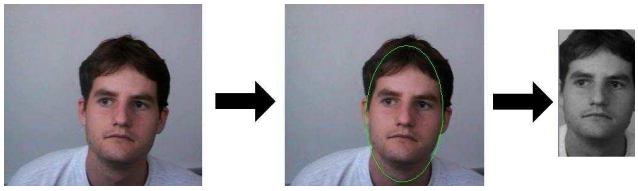


Figure 2. Face Image Normalization

4. Generic Face Features Selection

In this section, we search for facial features robust to changes in illumination, pose and identity. We show how to describe an image with receptive fields, then how to automatically learn facial features with clustering and finally determine salient regions of a face.

4.1. Normalized Receptive Fields

Gaussian derivatives provide a feature vector for local appearance that can be made invariant. We use a five dimensional feature vector computed at each pixel by computing the convolution with the first derivative of a Gaussian in x and y direction (G_x, G_y) and the second derivatives (G_{xx}, G_{xy} and G_{yy}). We use grey-level image of the face to be robust to chrominance variations of lights (sun, neon lights,...). We do not use the zeroth order Gaussian derivative in order to remain robust to changes in illumination intensity. Derivatives of higher order have been found to contribute little information for detection [5].

The feature vector (G_x,G_y,G_{xx},G_{xy},G_{yy}) describes the local appearance of a neighboorhood and is determined using Gaussian derivatives that are normalized to the characteristic scale at each pixel. An example of feature vector of a pixel can be seen in Figure 3. The characteristic scale at each pixel is determined with the local maximum of the Laplacian as function of scale (the scale parameter of the Gaussian), as proposed in [21]. The normalization of face image into an imagette allows us to reduce the range in which the characteristic scale is searched. Two neighboorhoods similar in appearance are close in the feature space. We use a fast, pyramid based, process for determining scale normalized gaussian derivatives [13].

4.2. Learning robust feature detectors by clustering

The vector of Gaussian derivatives form a feature space. In order to provide a distance metric, Gaussian derivative vectors may be normalized by their variance taken from a sample set. The vectors of Gaussian derivatives from face imagettes taken from a variety of viewing angle form clouds

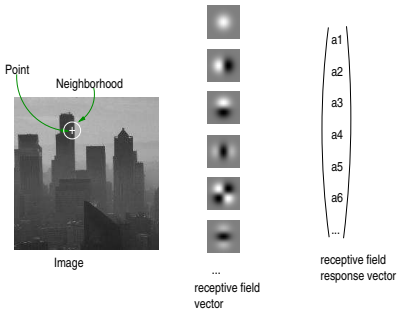


Figure 3. Appearance based feature vector

in this feature space. Each such cloud corresponds to a linear combination of Gaussian derivatives that do not change as viewing angle changes. Such clouds may be detected using K-means clustering.

A clustering algorithm, such as K-means can be used to determine a local description of appearance for specific facial structures. The center of gravity of a cluster can be used to determine the coefficients for a linear classifier. The mass of the cloud provides the basis for the determining the suitability of the cluster. Ideally we want clusters that have a low mass in each image (i.e. that correspond to a few specific facial structures), but a high overall mass in a set of training images taken from different viewpoints. Gaussian derivative vectors that satisfy both criteria are ideal for robust facial structure detection. Our experiments have shown that such clusters are sufficiently robust that even a single cluster can provide a robust detector for salient facial structures.

4.3. Robust Facial Structures

Applying clustering to the feature vectors for multiple images from several faces provide appearance clusters for background, hair and different skin regions as well as salient facial structures. For each pixel, we determine the most likely cluster using a sum of squared difference from the cluster center. The squared difference of each Gaussian derivative is normalized by the overall variance of that derivative. The sum of the squares of the normalized distance provide a similarity metric.

The process of robust facial structures detection is shown in Figure 4. Each pixel is assigned to the most likely cluster as defined by the smallest normalized distance. If the normalized distance is greater than a threshold, the pixel is assigned to a "background" class. Adjacent pixels in the same class are detected by connectivity analysis and grouped to form image regions. These image regions correspond to salient facial structures such as the eyes, nose, mouth and chin.

Detection using this method can give rise to a number of small spurious detected regions. These can be eliminated by using a bounding box of the region as defined by the connected components algorithm. Regions with a small bounding box are eliminated. The remaining regions correspond to salient facial structures. Connected components analysis also provides some geometrical information about the detected regions. The first and second moments of the connected components provides information about position and extent. This information can be reprojected to the original image.

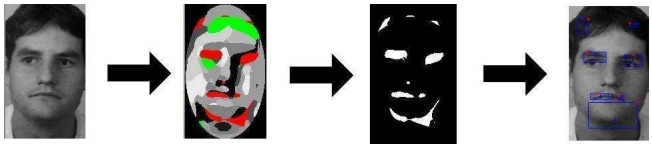


Figure 4. Robust facial structures detection process

Face image normalization,
Mapping : Regions in red and green are considered as salient robust facial structures and reprojected into a binary map,
Connected components analysis

5. Pose Estimation

Head orientation, or pose, is determined by 2 angles, the vertical angle α_v and the horizontal angle α_h. A dense sampling of appearance space, such as provided by the Pointing '04 database, makes it possible estimate these angles by image classification. A more precise estimate requires geometric calculation based on the relative image positions of salient image structures. These two methods may be used in a complementary manner, with the coarse estimate obtained by classification used to initialize a refined calculation based on image position of salient facial structures.

In this section we discuss how to compute this more refined calculation based on the relative image positions of salient facial structures.

5.1. Detecting Eyes

The position of salient facial structures using the method described above can vary with respect to image pose, as illustrated in figure 5. Even for a particular viewing angle, a particular robustly detected salient facial structure may occur at different relative positions for different subjects. For example, figure 6 shows the eye positions detected for several people. Our conclusion is that the simple position of

robust facial structures is not sufficient to allow direct structure identification.

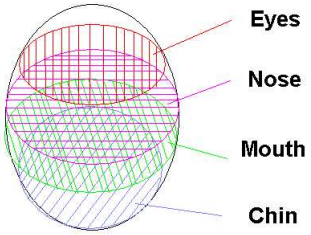


Figure 5. Facial structures position variation for one person when the pose is changing

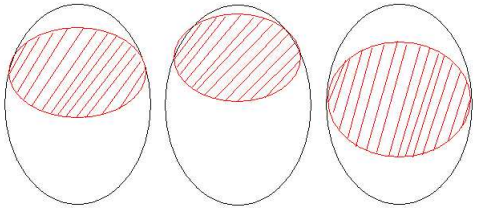


Figure 6. Eyes postion variation for 3 subjects when the pose is changing

We use a bayesian classifier to identify detected regions corresponding to particular salient facial structures. We estimate the probability that a bounding box contains a particular facial structure. Eyes have been found to be the most salient of facial structures (see Section 6.3 for details). Furthermore, knowing their position in the face provides strong geometric constraints for searching for other facial structures. Thus our first step is to identify the bounding boxes that correspond to the eyes.

There are three possible configurations of detected eye regions for bounding boxes that contain eyes :
1) One bounding box for each eye
2) One bounding box for both eyes
3) One bounding box for one eye. This situation appears when the face is turned so that only one eye is visible from the camera.
Giving a configuration, we compute the probability that each bounding box corresponds to an eye. Configurations are tested in the order of the three configurations listed above. The bounding box containing eyes are selected with a winner-takes-all process using Bayes rule. Eyes are identified and their position is given by the center of gravity of the bounding box in configurations 1 and 3, and extremities of the bounding box in configuration 2.

5.2. Computing Head Pose

In this section we discuss how to compute α_h and α_v using robust facial structures. Because of facial symmetry, horizontal pose can be estimated with positions of both eyes with regard to the face. The trigonometric computation of the horizontal angle is shown figure 7. We obtain the following equations (3):

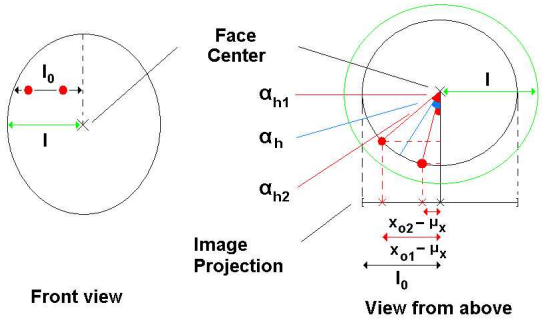


Figure 7. Horizontal pose computation

$$x_o^1 - \mu_x = l_0 sin(\alpha_h^1)$$

$$x_o^2 - \mu_x = l_0 sin(\alpha_h^2)$$

$$\alpha_h = \frac{(\alpha_h^1 + \alpha_h^2)}{2} \quad (3)$$

The relative position of eyes is not sufficient to estimate the vertical pose α_v. Because of the variation of eye position in the face when $\alpha_v = \alpha_h = 0$ due to the subjects, specifying eyes position for $\alpha_v = 0$ is a difficult task. This problem can be bypassed by calibrating an eyes position at $\alpha_v = 0$ for each subject, but the system becomes dependent on identity. We must use positions of other robust facial structures to estimate the vertical pose. But as with eyes, positions of salient structures in the face varies with identity. Furthermore, even for human eyes, a 15 degrees difference in vertical orientation is not apparent. A solution would be to consider distances between other facial structures.

6. Experimental Results and Discussion

6.1. Training data

The choice of a good database is essential for proper learning. To detect salient facial structures that are robust under changing conditions, we used two front images of 15 subjects to learn feature vectors. Subject were 20 to 40

Table 3. *Recall / Precision in % when changing the head orientation in the training process*
Results obtained with a detection threshold of 0.25

Images	Frontal	Near-Frontal	All
Person 1	36,7 / 30,1	40,9 / 4,8	31,1 / 24,6
Person 2	34 / 35,4	35,6 / 4,1	35,2 / 6,7

years old. Five subjects have facial hair and seven people wear glasses. Non-frontal images can introduce noise in the data, because some facial structures have different appearances in different poses. As an example, the experiments from two subjects are shown in Table 3. Front pose provides more generic appearance for salient facial structures, which remain robust on multiple poses after learning, whereas profile images provides appearance for salient structures in profile, but not for front.

To remain robust to changes in identity, we have used images from 15 different people from the Pointing '04 database. These subjects may be grouped into two classes:

- Class A, in which the face is typical with regard to people in the database. In the Pointing '04 data, 73% of the subjects have white skin, European facial structure and no beard.
- Class B, in which the face is atypical in some respect. Examples of atypical faces from the database include those who wear glasses, have a beard or have a slightly different skin pigment. In the Pointing '04 data, 27% of the subjects have darker skin, oriental facial structures or a beard.

We have observed the following results from the learning process:

- The clustering C(A) is performed only with faces from class A.
  - Regions obtained for facial structures of a subject a $\in$ A are significant and robust.
  - Regions obtained for facial structures of a subject b $\in$ B are less significant and more noisy.
- The clustering C(A+a) is done with people belonging to class A and a new subject (a) also belonging to class A. Regions obtained for facial structures of the subject (a) are less robust than those obtained with the previous clustering C(A), indicating that robustness decreases as we add subjects.
- The clustering C(A+b) is done with people belonging to class A and a single subject (b) belonging to class B. Regions obtained for facial structures for the subject (b) are less noisy and more salient than those obtained with the previous clustering C(A).

Table 4. *Results obtained with 30 front images and a detection threshold of 0.4*

Number K	2	3	5	7
Recall	11,7 %	22,7 %	70,7 %	30,7 %
Precision	2,3 %	13,1 %	18,2 %	21,5 %

Number K	10	15	20
Recall	40,2 %	12,2 %	6,1 %
Precision	47,7 %	57,3 %	11,7 %

These observations can be explained in the following way. The clustering C(A) performed with "common" faces provides better results on subjects of class A, than subjects of class B. Therefore the clustering C(A) is not well adapted for subjects of class B. We must then use other people in our learning process to remain robust to changes in identity.

Adding a new subject a from $\in$ A in the clustering does not bring much additional information, even on the subject (a). Furthermore, it can lead to a degradation of robustness and more noise, because the method becomes specialized for people from the class A, degrading independence to identity. The method may be said to "overfit" the training data and lose generality.

Adding a new subject b $\in$ B provides better detection of facial structures on (b), whose appearance differs from those of class A. Clustering C(A+b) adapts to the image of the face of (b) without becoming specialized. Furthermore, salient facial structures are more often detected with C(A+b) than with the clustering C(A).

6.2. Influence of the number of clusters

The clustering step gathers feature vectors into K clusters. This step is an important part in the learning process. Therefore, the choice of the number of clusters K is crucial. If K is too small, appearance clusters won't be discriminative enough to detect salient facial structures. If K is too big, regions will be too small and too unstable in the image. During our experiments, we tested several K and obtained good results with K = 10. Resulting images with different number of clusters can be seen in Figure 8.

To measure the recall and the precision for each different K, we have employed a 10x15 grid on the normalized imagette of the face (see Table 4). Cases in the grid are manually labelled as follows : 1 if the case contain a facial salient structure, 0 otherwise. During the tests, a case of the grid gets the value 1 if the ratio of the number of salient cluster pixels in the case over the total number of pixels in the case exceeds a fixed threshold. This threshold is called

Table 5. *Recall/Precision with regard to detection threshold*

Detection Threshold	0,1	0,25	0,4
Recall	46,3 %	34 %	23,2 %
Precision	22,2 %	25,4 %	26,9 %
Detection Threshold	0,5	0,66	0,75
Recall	17,9 %	10,3 %	7,5 %
Precision	27 %	27 %	27,3 %

the detection threshold (see Table 5).

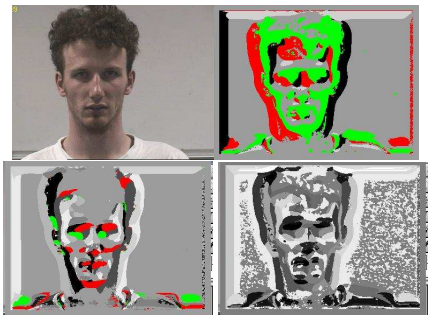


Figure 8. Influence of the number of clusters
Regions in red and green are considered as salient robust facial structures
Top left image is the original image
Top right image is obtained with 5 clusters, which are not discriminative enough
Bottom left image is obtained with 10 clusters
Bottom right image is obtained with 15 clusters. Regions are too small to be relevant

6.3. Facial structure detection performance

Tests have been made with representative people under changing lighting and pose conditions. The pose is determined by 2 angles (h,v), which vary from -90 degrees to +90 degrees. Each set contains 93 images of the same person at different poses. The Pointing '04 database includes faces with glasses as well as a variety of skin pigments. We have calculated the detection rate for each structure for four representative faces (see Table 6).

With an average detection rate of 97 %, eyes are the most often detected facial structure. Eye appearance does not vary as much as the other facial structures because of their spherical shape and thus eyes can be detected under several points of view. Glasses have little effect on eye detection.

Table 6. *Facial structure positive detection rate*

Structure	Person 1	Person 2	Person 3	Person 4
Eyes	99 %	97 %	98 %	95 %
Nose	70 %	82 %	61 %	82 %
Mouth	85 %	90 %	95 %	85 %
Chin	84 %	88 %	91 %	84 %
Specificity	-	Glasses	Beard	Matt skin

The salience of a mouth improves when it is surrounded by a beard and thus mouth detection is slightly better than eye detecion for bearded subjects.

For 63% of the observed errors, the head pitch is inferior to -30 degrees, indicating that the subject is looking down. This situation represents only 29% of all poses. Indeed, in this situation, eyes are no more visible in the image, but only eyebrows. Therefore, we have trained our algorithm on images on which subjects' head pitch is inferior to -30 degrees. In this case, the resulting clusters are less discriminating and provides lower detection rate on face images. As a consequence, some facial structures, such as chin and eyes, are less salient. Eye detection is 59% less efficient with the algorithm trained with images of people looking down. The nose has the worst detection average rate with 74%. It does not have as many symmetry properties as eyes and its appearance can suffer many variations. That is why the nose is less often detected than other facial structures.

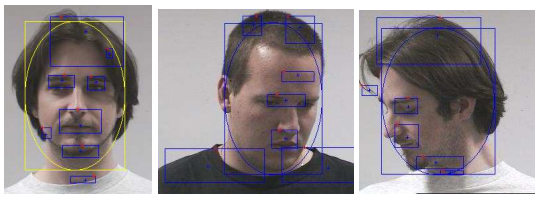


Figure 9. Examples of facial structures detection [19]

6.4. Influence of the size of the face imagette

To show the importance of the face image normalization step, we have measured eye detection rates with different sizes of the face imagettes. Results of these experiments can be seen in table 7. Tests have been made on a sequence of 500 images in which the subject moves but has both eyes remaining visible on the screen. The head size changes from 50x50 to 20x20 pixels in the sequences.

The last size, 50x50 pixels, corresponds to face image analysis without normalization, as the face in the sequence

Table 7. *Eyes detection rate with regard to the size of the imagette in pixels*

Size	120x200	120x120	60x100	50x50
Detection	98,2 %	97,8 %	94,2 %	1,8 %

has a maximal size of 50x50 pixels. We can see how the normalizing the first and second moments of the imagette enhances the detection rate. This provides the ability to deal with 20x20 pixels images of the head, such as panoramic or wide-angle public cameras images. When this operation is not performed, regions will be more imprecise and may not be found. Increasing the size of the normalized face image increases the accuracy of structure detection in the original image of the face. For our experiments, the face imagette has a size of 60x100 pixels.

6.5. Pose Estimation

Due to difficulties in estimating the vertical pose, we have only estimated the horizontal pose. Absolute difference between the real and the estimated horizontal angle α_h has been computed for all 1395 images of fifteen subjects in the Pointing'04 database. Mean absolute error in degrees of the horizontal angle for each pose is represented figure 10.

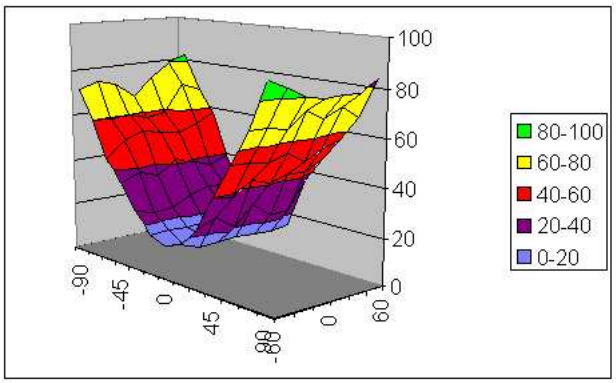


Figure 10. Mean absolute error in degrees of horizontal angle at each pose

Mean error of the horizontal angle does not vary much as vertical pose changes. When $|\alpha_h| <= 45^o$, mean error drops to 15 to 5 degrees. But at $|\alpha_h| > 45^o$, mean error can reach 90 degrees. There are several explanations for this observation.

First, in the case only one eye is visible at the screen, which roughly corresponds to $|\alpha_h| >\approx 45^o$, estimating the horizontal pose becomes difficult, because the other eye cannot be seen. As we need two position for eyes to estimate α_h, the computation of the horizontal pose can not be made accurately. Furthermore, the fact that people do not have the same distance between eyes makes the prediction of the position of the other eye inaccurate and computing the horizontal angle is even more difficult.

Another problem is the neck detection. The user's neck can be detected or not as part of the face because of its chrominance. In profile, detecting the neck disrupts face orientation estimation and can modify the slant angle of the face in the image plan. As the horizontal pose estimation relies on the face estimation, the neck also yields a false estimation for the horizontal pose.

An additional problem is caused by hair. Hair are the part of the face that vary the most with regard to identity, degrading invariance to identity. When the user is in profile, hair is more visible in the image. Finally, A mesh of hair is sometimes detected as an eye, and this detection provides a false estimation for the horizontal pose.

Conclusions

We have proposed a new approach to detect salient facial structures in a manner that is robust to changes in viewing angle, illumination and identity. The imagette containing the face is normalized in scale and orientation using moments provided by a face tracker. Each pixel in the face image is associated with an appearance cluster. One particular cluster stands for salient robust face structures which are: eyes, nose, mouth, chin. We have tried to extract and exploit a maximum of information provided by a single image of a face and to limit the loss of generality.

Detected regions can be delimited with rectangles in the image. Identifying facial structures using positions relative to the face image is difficult because multiple variations of structures are possible.

These variations are due to changing orientation, facial expression of emotion and especially identity. A Bayesian classifier is used to identify the regions. Eyes have been found to be the most salient of the facial structures. They can be used to obtain a coarse estimation of the horizontal pose, but are not sufficient to compute vertical pose. Because of variations in the structures in the face with regard to the identity and the pose, vertical pose is difficult to accurately estimate.

Mean error for the horizontal pose does not vary with vertical angle. Error reaches 5 to 15 degrees when $|\alpha_h| <= 45^o$, but increases when $|\alpha_h| > 45^o$. This is due to the fact the horizontal angle is hard to estimate with only one eye visible on the image and that the neck detection disrupts the face estimation. Hair can also be misclassified as eyes. All these observations tend to show that the robustness to

identity is the most difficult criteria to respect.

References

[1] M. Storring "Computer Vision and Human Skin Color" Doctoral Thesis, Alborg University, June, 2004.

[2] M. Zobel, A. Gebhard, D. Paulus, J. Denzler, H. Niemann "Robust Facial Feature Localization by Coupled Features" *4th International Conference on Automatic Face and Gesture Recognition*, March 28-30/2000 Grenoble, France pp. 2-7.

[3] D. Hall, J.L. Crowley "Computation of Generic Features for Object Classifi cation" *ScaleSpace*, 2003.

[4] S. J. McKenna, S. Gong, R. P. Wurtz, J. Tanner, D. Banin "Tracking Facial Feature Points with Gabor Wavelets and Shape Models" *1st Int. Conf. on Audio- and Videobased Biometric Person Authentication*, Lecture Notes in Computer Science 1997.

[5] D. Hall "Viewpoint Independant Object Recognition from Local Appearence" *PhD Thesis*, 2001.

[6] K. Schwerdt, J.L. Crowley "Robust face Tracking using Color" *Automatic face and Gesture Recognition*, pp 90-95, 2000.

[7] L. Wiskott, J-M. Fellous, N. Kruger, C. von der Malsburg "Face Recognition by Elastic Bunch Graph Matching" *Pattern Analysis and Machine Intelligence*, Vol 19 pp 775-779, 1997.

[8] J. Malik, S. Belongie, T. Leung, J. Shi "Contour and Texture Analysis for Image Segmentation" *IJCV*, Vol 43 pp 7-27, 2001.

[9] V. Kruger, G. Sommer "Gabor Wavelets Networks for Object Representation and Face Recognition" *Deutsche Arbeitsgemeinschaft fur Mustererkennung*, Vol 22, Sept. 13-15, 2000.

[10] C. Schmid "Constructing Models for Content-Based Image Retrieval" *Computer Vision and Pattern Recognition*, 2001.

[11] D. Lisin, E. Risemann, A. Hanson "Extracting Salient Image Features for Reliable matching using Outlier Detection Techniques" *Computer Vision Systems Third International Conference*, pp 481-491, 2003.

[12] M. Turk, A. Pentland "Eigenfaces for Recognition" *Cognitive Neuroscience*, Vol 3(1), pp 71-96, 1991.

[13] J.L. Crowley, O. Riff "Fast Computation of Scale Normalised Receptive Fields" *International Conference Scale Space*, June 2003, Island of Skye.

[14] Y. Wu, K. Toyama "Wide-Range, Person- and Illumination-Insensitive Head Orientation Estimation" *4th International Conference on Automatic Face and Gesture Recognition*, March 2000 Grenoble, France pp. 183-188.

[15] A.C, Varchmin, R. Rae, H. Ritter "Image based Recognition of gaze Direction using Adaptative methods" *International Gesture Workshop*, Springer, 1997, pp. 245-257.

[16] A.J. Howell, H. Buxton "Active Vision Techniques for Visually Mediated Interaction" *Image and Vision Computing 20(12)*, 2002, pp.861-871.

[17] T. Otsuka, J. Ohya "Real-time Estimation of Head Motion Using weak Perspective Epipolar Geometry" *Proceedings of WACV'98*, October 1998, Princeton.

[18] K.N. Walker, T.F. Cootes, C.J Taylor "Automatically Building Appearance Models from Image Sequences using Salient Features" *IVC(20), No. 5-6*, 15 April 2002, pp. 435-440.

[19] A.M. Martinez and R. Benavente "The AR Face Database" *CVC Technical Report n.24*, June 1998.

[20] G.J. Klinker, S.A. Shafer, T. Kanade "A Physical Approach to Color Image Understanding" *IJCV 1990*

[21] T. Lindeberg "Feature Detection with Automatic Scale Selection" *IJCV 1998 (30) Number 2*, pp. 79-116.

Pointing Gesture Visual Recognition for Large Display

Sébastien Carbini, Jean Emmanuel Viallet and Olivier Bernier
France Télécom R&D/DTL/TIC
Technopole Anticipa, 2 Avenue Pierre Marzin
22307 Lannion Cedex - France
{sebastien.carbini, jeanemmanuel.viallet, olivier.bernier}@rd.francetelecom.com

Abstract

Among gestures naturally performed by users during communication, pointing gestures can be easily recognized and included in more natural new Human Computer Interfaces. We approximate the eye-finger pointing direction of a user by detecting and tracking, in real time, the 3d positions of the centre of the face and of both hands; the positions are obtained by a stereoscopic device located on the top of the display. From the head position and biometric constraints, we define both a rest area and an action area. In this former area, the hands are searched for and the pointing intention is detected. The first hand spontaneously moved forward by the user is defined as the pointing hand whereas the second detected hand, when it first moves forwards, is considered as the selection hand. Experiments on spatial precision, carried out with a group of users, show that the minimum size of an object to be easily pointed is some 1.5 percent of the diagonal of the large display.

1. Introduction

Computer vision applied to gesture recognition allows users to freely interact unencumbered, without carrying specific devices or markers. Amongst gestures occurring during non-verbal communication, pointing gestures can be easily recognized and included in more natural new human computer interfaces. Several studies have been performed in the field of vision based pointing gesture recognition. In this paper, we consider a pointing gesture, not as spatio-temporal trajectory to be recognized, but together with [2, 5, 6] as the instantaneous pointed location on a display. In [2], the pointed direction is given by the forearm of the user, estimated with a 3d model of the bust and of the arm. From an image processing point of view, the forearm exhibits few discriminating features and is thus difficult to detect when one points in the direction of the camera. On the other hand, the face has a stable and characteristic shape that can be detected and tracked. Without visual feedback, a pointing method with an axis that does not include an alignment with an eye (extended finger axis, forearm or extended arm axis) is less precise that a pointing gesture which involves aiming at a target using both one's eye and typically the tip of a finger.
We propose to use an eye-alignment aiming convention and to approximate the "eye-tip of the finger" pointing direction by the face-hand direction (Figure 1). The first hand spontaneously moved towards the display by the user is defined as the pointing hand whereas the second detected hand, when it first move forwards, is considered as the selection hand (similar to a mouse click) or as a control of a third axis useful for virtual 3d world interactions.
With the EM algorithm, the best ellipsoidal models of the face and hands, which correspond to skin color pixels seen by two cameras, can be found. But the 3d tracking suffers from skin-color distractors (background or clothes) [1]. With two cameras, tracking of both hands, detected within a predefined "engagement plane" allows bi-manual interaction with standard size PC monitor [7]. We propose a face and hand tracker robust to lighting changes, complex background changes, partial face occultation and skin color distractors suitable for large screen display without a fixed engagement plane but with an engagement plane relative to the current position of the user, thus free to move.

2. Methodology

In the pointing method described here, it is not a necessity to know beforehand the dominant hand of a user. It is both suited for right-handed or left-handed users. Furthermore, no calibration step or manual initialisation is needed. The face of the user is automatically detected as it enters the field of view of the camera, provided that the image face width is greater than 15 pixels and the out of plane rotation of the face is below 50 degrees [3]. The user begins to interact with the display, with the hand he or she favours to use. As soon as a body part is detected (face or hand), the body

part is continuously tracked until tracking failure is automatically detected: then body part detection is re-triggered (Figure 2). Pointing is taken into account only in the action area, where the hand is sufficiently in front of the head. Otherwise, the hand lies in the rest area (hand on a hip for instance) and the user is able not to interact continuously (Figure 3-a).

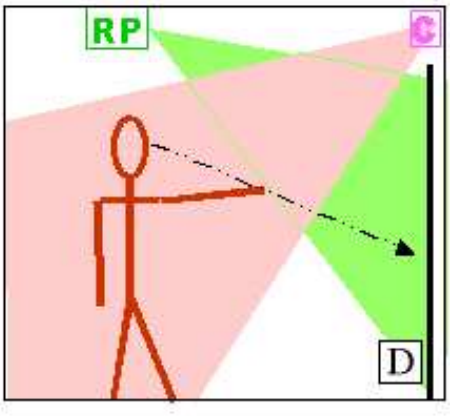


Figure 1. Experimental setup. With camera (C) resolution greater than 160x120, head and hands are still tracked as the user stands behind the retroprojector (RP) and interacts with the display (D).

3. Face detection initialization

An accurate face detector is used to initialize the tracking: it is a modular neural network with high detection rate and a very low false alarm rate. It has been described in details in [3]. For HMI applications, failure to detect face in an image is not crucial since the face can be detected in the following images. On the other hand, continuously tracking a false alarm is a major problem. The face detector is accurate but slow. To speed up the process, the following procedure is used. The full rectified image is equally divided into sixteen regions. At each image, candidate regions for face detection are detected. These are regions with a sufficient number of moving pixels, skin colored pixels (Figure 3-c) and depth pixels within the face detector detection range (Figure 3-d). Only one of these candidate region is then selected at each image. The face detector is only applied on this region, giving real time performance. A selection algorithm ensures that a region is not selected again before all other candidate regions are first tested. Once a face is found, the 3d tracking is initialized on the rectified image rectangle corresponding to the detected face.

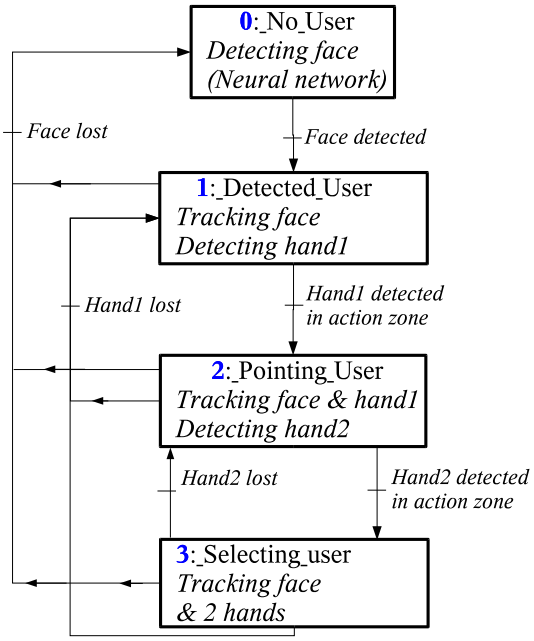


Figure 2. The different states of the pointing gesture recognition system.

4. Tracking statistical model

The core of the tracking process is the statistical model, which aims at explaining the result of the pre-processing stages and is adapted at each image using the EM algorithm. The model is composed of two independent parts, one to explain the skin colored pixels and a second to explain the depth pixels.

The color part is a mixture of two probabilities, one for the face pixels, and the other for the remaining skin colored pixels. The depth part is also a mixture of two probabilities, one for the face pixels, and the other for the remaining depth pixels. The color and depth face probabilities are respectively the product of a gaussian for the position of the pixel with a color and with a depth histogram. Let us define for each skin colored pixel a vector (color observation vector) X and for each depth pixel a vector D:

$$X = \begin{pmatrix} x \\ y \\ u \\ v \end{pmatrix}, \quad D = \begin{pmatrix} x \\ y \\ d \end{pmatrix}$$

where x and y are the coordinates of the pixel, u, v the chrominance part of the color of this pixel in the YUV color space, and d is the depth of the pixel. the probability for the face components are:

$$\mathcal{P}^{face}_{color}(X) = \frac{1}{2\pi\sigma^2} e^{-\frac{1}{2\sigma^2}\left\{(x-x_0)^2+(y-y_0)^2\right\}} H^{face}_{color}(u,v)$$

$$\mathcal{P}^{face}_{depth}(D) = \frac{1}{2\pi\sigma^2} e^{-\frac{1}{2\sigma^2}\left\{(x-x_0)^2+(y-y_0)^2\right\}} H^{face}_{depth}(d)$$

where x_0 and y_0 are the coordinates of the center of the face. For the other components:

$$\mathcal{P}(X)^{no}_{color} = H^{no}_{color}(u,v), \mathcal{P}(D)^{no}_{depth} = H^{no}_{depth}(d)$$

$H^{face}_{color,depth}$ and $H^{no}_{color,depth}$ are two color and two depth histograms.

All histograms are normalized as probabilities (sum equal to one). The parameters of the model, which are updated at each image, are the position of the face and the four histograms. At initialization, the skin colored pixels inside the detected rectangle (corresponding to the face) are used to initialize the face color histogram. The non face color histogram is initialized as the complement of the face color histogram. The face depth and non face depth histograms are initialized in a similar way.

4.1. Model parameters priors

The parameters for the previous image are used to force priors on the current parameters in order to avoid tracking divergence. A simple gaussian centered on the previous position is used as face position prior without using a dynamic model of face movement. The model can track a face without restriction on its movement, at the cost of a probability of tracking failure for a fast moving face 3d close to another face or hand. A dynamic model can be introduced by simply replacing the previous position with an estimation of the current position, using prior positions and a dynamic model. For the histograms, the exponential of the Kullback-Leibler distance between the prior histograms and the current histograms is used. Let h be the previous image histograms:

$$\mathcal{P}^{prior}\left(H^{face,no}_{color}\right) = e^{\alpha_c \sum_{u,v} h^{face,no}_{color}(u,v) log(H^{face,no}_{color}(u,v))}$$

$$\mathcal{P}^{prior}\left(H^{face,no}_{depth}\right) = e^{\alpha_d \sum_{d} h^{face,no}_{depth}(d) log(H^{face,no}_{depth}(d))}$$

where α_c and α_d are parameters chosen to take into account the expected variability of the corresponding histogram. For the color and depth histograms, fairly stable, we chose $\alpha_c = \alpha_d$.

4.2. Model parameters update

The model is initialized at each image using the parameters of the previous image. The model is adapted to the current image or, more precisely, to the current observations vectors, using the EM algorithm. The EM algorithm is used with a fixed number of iterations per image, to control and limit the computing time. Only the observation vectors corresponding to moving pixels, or those in the vicinity of the previous position of the face, are taken into account. This facilitates the tracking process, and speeds up the EM algorithm.

In the model, the hidden variables are the association of each observation vector to each component of the mixtures. The mean values of the hidden variables knowing the current estimated parameters are computed in the E expectation step. These mean values are, for each observation vector, the probability that it belongs to each component. Let z^{cf}_i the probability that a color vector X_i belongs to the face, and z^{cn}_i the probability that it belongs to the other color component. Let z^{df}_j the probability that a depth vector D_j belongs to the face, and z^{dn}_j the probability that it belongs to the other depth component. The following mean histograms are calculated:

$$\hat{H}^{face,no}_{color}(u,v) = \frac{1}{I}\sum_i z^{cf,cn}_i \delta(u_i - u)\delta(v_i - v)$$

$$\hat{H}^{face,no}_{depth}(d) = \frac{1}{J}\sum_j z^{df,dn}_j \delta(d_j - d)$$

where δ is the Kronecker function, I is the number of color observation vectors and J is the number of depth observation vectors.

The new estimated color and depth histograms (face and non face) are expressed as a linear combination of the mean histograms $\hat{H}$ and the previous h histograms:

$$H^{face,no}_{color}(u,v) = \frac{1}{b_c}\left\{\hat{H}^{face,no}_{color}(u,v) + \alpha_c h^{face,no}_{color}(u,v)\right\}$$

$$H^{face,no}_{depth}(d) = \frac{1}{b_d}\left\{\hat{H}^{face,no}_{depth}(d) + \alpha_d h^{face,no}_{depth}(d)\right\}$$

where b_c and b_d are normalization factors, so that the histograms H can be interpreted as probabilities.

$$\bar{x}_{color} = \frac{1}{I}\sum_i z^{cf}_i x^c_i$$

is the position estimated from the color observations and

$$\bar{x}_{depth} = \frac{1}{J}\sum_j z^{df}_j x^d_j$$

from the depth observations. The new $\bar{x}$ position of the model is given by $\bar{x} = \frac{1}{2}(\bar{x}_{color} + \bar{x}_{depth})$. The $\bar{y}$ position is obtained in a similar manner. The $\bar{z}$ position is extracted from the depth observations.

5 Hand detection and tracking

Contrary to faces, hands exhibit extremely variable shapes as seen from a camera and is thus difficult to detect, specifically at low image resolution. Once the face is detected and using disparity information, the 3d position of the face is obtained. Moving skin colour zones are taken as hand candidates. Biometric constraints limit the search space to a centroid centred on the face. Furthermore, it is reasonable to admit that, when interacting with the display, the user moves its hand towards the display and sufficiently away from its face (more than 30 cm). Thus the hand search space is restricted to a volume delimited by a sphere and a plane, a volume called the 'action area' (Figure 3-a).

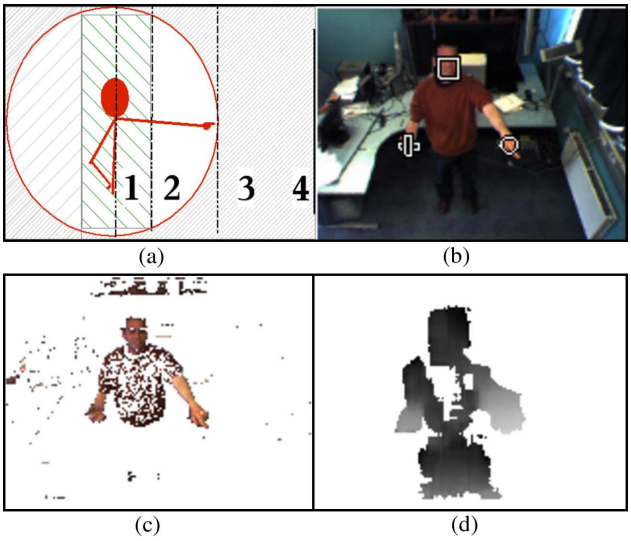


Figure 3. (a): 1. Rest area, 2. Action area, 3. Hand unreachable area, 4. Display - (b): Face position (square), pointing hand position (circle), selection hand position (cross) - (c) Skin color - (d) Disparity in both rest and action area.

A hand is detected as a skin color moving zone, in the action area, the closest to the display. The first detected hand is considered to be the pointing hand.
Then the second hand is detected in a similar manner after having previously discarded both the area of the first detected hand and the area of the corresponding arm from the search space. Indeed, a naked arm or an arm covered by a skin colour cloth could be mistakenly selected as the second skin colour moving zone closest to the display. In order to detect the arm, a skin colour zone is first initialised on the hand. Then it is merged, at each iteration, with skin colour neighbouring pixels with continuous depth values and located in front of the face. The arm detection iterates until the zone no longer grows. The second hand is used as a control command (grabbing a chess piece in figure 7) as it enters the action area or to control a third axis by changing its distance to the display (the zoom for navigation purposes in figure 8).
Tracking of the two hands and their pointing-command labels is necessary otherwise as soon as the selection hand gets closer to the display than the other hand, their actions would swap. Hand tracking, initialised upon hand detection, is performed in a similar manner (see previous section) as face tracking. With up sleeves or skin colour clothes, tracking can possibly position itself anywhere along the arm. In order to find the hand, the tracking solution is oriented, using disparity gradient, towards the arm extremity the closest to the display. This reframing is inadequate for small gradients but then the forearm, parallel to the display, is in the rest area and pointing is not taken into account.
Since body parts tracking rely on similar features, in case of occlusion of the face or a hand by another hand, only one of the body part is correctly tracked whereas the other is erroneously anchored on the first and cannot any longer be controlled by the user. The tracked zones remain fused even after the end of the physical occlusion.
In pointing situation, the most frequent case of occlusion occurs when a hand passes in front of the head in the camera direction. To solve this problem, we consider that it is the hand that moves and not the face: face tracking is temporarily interrupted until the end of the occlusion and for the hand tracking the search space is further constrained in order to discard pixels with disparity values close to the face.
The other cases of occlusions are more difficult to deal with and we first identify fusion of body parts. In case of a face-hand fusion, experiments show that usually it is the estimated hand that positions itself on the face and stays there, whereas if the estimated face is anchored on the physical hand then face tracking quickly fails, tracking failure is automatically detected and forces the re-initialisation of the face by detection. Therefore, in case of detected face-hand fusion, the face is kept and the hand destroyed and detected again. Hand-hand fusion leads to the elimination of both hands and hand detection is launched.
Face and hand tracking failure, either in the rest or the action area, are automatically detected by estimating the number of skin-colour valid disparity pixels. If lower than a threshold, the body part tracking is considered to have failed and detection is automatically re-triggered.

6 Experimental set-up and performances

The mean performances of the pointing system is estimated by several experiments on spatial and temporal precision, experiments carried out by a group of 14 users. A user is located at 1.5 m of a retro-projected image of 2 x 1.7

m size. A Bumblebee stereo camera [4], used with a resolution 160x120 is located above the display at an angle of 45 degrees with the display (Figure1) in order to maximize hand displacement on the image and thus spatial precision when the user points towards the four corners of the display. In this set-up, a 1 cm change of the face position (with a hand still) roughly corresponds to a 1 cm change of the aimed location on the display. The system performs at 25 Hz, 75 percent load, on a Pentium IV (3 GHz).
In order to characterize temporal stability, a user is asked to aim at a cross at the centre of the display for about 10 seconds. The mean temporal stability is 2.29 cm and a typical result is given in figure 4-a. To evaluate the spatial precision, each user has to follow as closely as possible the path of rectangles, centred on the display and of decreasing size. The mean distance of the aimed location to the closest rectangle boundary point are estimated. A typical track is given in figure 4-b.

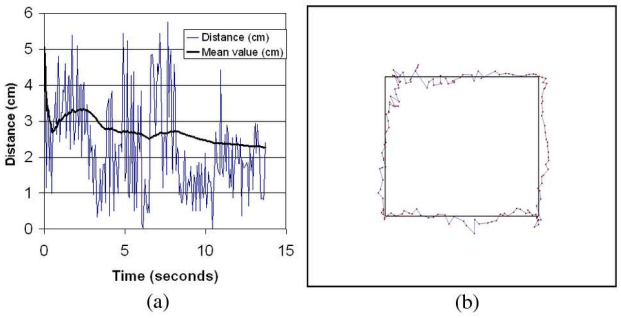


Figure 4. Pointing gestures experiments: (a) Temporal stability (mean distance = 2.29 cm). (b) Spatial precision path for a rectangle size of 102x85 cm (mean distance = 2.75 cm).

WxH (cm)	61x51	102x85	143x120	185x154
D (cm)	2.38	2.70	3.24	3.45

Table 1. Mean distance D to the rectangle of width W and height H.

One may notice that the precision during a moving pointing gesture is less than the precision obtained for a still gesture. The moving precision is equivalent for both horizontal and vertical displacements (which comforts the 45 degrees orientation of the camera with the display). Moreover, the precision continuously decreases as a user aims from the centre to the boundary of the display (Table 1).
The next experiment seeks to determine the minimal size of

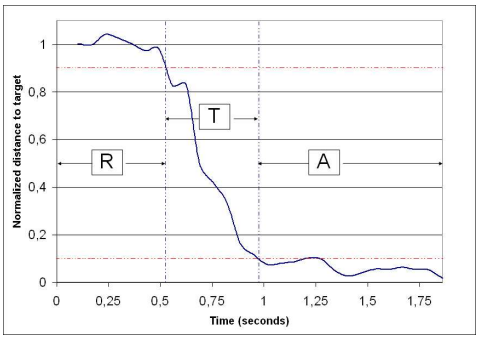


Figure 5. Typical inter-target trajectory: Reaction time R (distance>0,9), Trip time T (0,1<distance<0,9), Adjustment time A(distance<0,1).

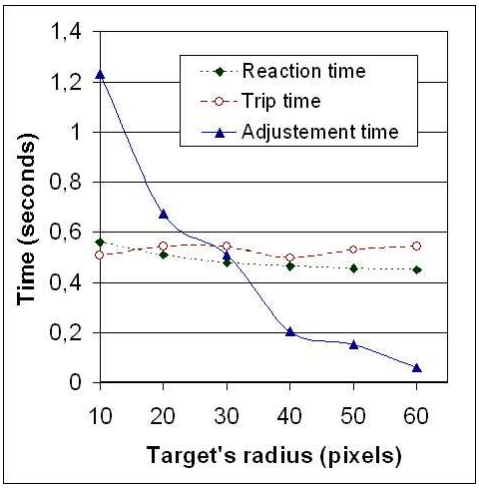


Figure 6. Evolution of reaction time, trip time and adjustment time versus target radius.

an object a user can reliably point at. A user is ask to successively aim at targets, located at the four corners of a square of 1 meter side, the size of the targets decreasing every four targets. A typical trajectory between two targets exhibits three parts (Figure 5). The first part (R) corresponds to the reaction time a user needs to ensure that the previous target has been reached, to locate the next target and start to move. The second part (T) is the trip time a user needs to quickly move across the inter-target space. During the last part (A), the user adjusts more precisely the aimed location in order to hit the target. As expected, both reaction and trip time are independent of the target size (Figure 6) whereas adjustment time increases as target size diminishes. We consider that when the adjustment time is greater than the reaction time, the precision is not sufficient and is a penalty for hu-

Figure 7. Moving a knight with hands in a chess game.

man computer interaction. We derive that the minimal size of an object which can easily be pointed at by a user, on a 2 x 1.7 m display, is around 5 cm, which allows a user to play chess (Figure 7) in the current conditions of our face-hand pointing system. Locations close to the boundary of the display are more difficult to reach than those at the center. However, this precision is sufficient for hand controlled navigation task (Figure 8).

7 Conclusion

An automatic detection and tracking of head and hands, allows to approximate an "eye-tip of the finger" direction by the head-hand direction for pointing purpose and interaction with large screens. The obtained temporal stability could allow a user to select an object pointed at during a few seconds as performed by [2], but may lead to accidental selection. A selection alternative could consist in identifying a change of shape of the pointing hand but is not feasible with the current 160x120 image resolution. Second hand tracking for secondary command provides an interesting solution because it involves spatially distinct modalities for pointing and for command.

Figure 8. Top : Pointing to explore a panoramic image with predominant hand (left or right). Bottom : Zooming in the image using the second hand.

References

[1] O. Bernier and D. Collobert. Head and hands 3d tracking in real time by the em algorithm. In *RATFG-RTS ICCV Workshop*, Vancouver, Canada, July 2001.

[2] D. Demirdjian and T. Darrell. 3-d articulated pose tracking for untethered diectic reference. In *Proceedings of International Conference on Multimodal Interfaces*, Pittsburgh, Pennsylvania, October 2002.

[3] R. Feraud, O. Bernier, J. Viallet, and M. Collobert. A fast and accurate face detector based on neural networks. *Pattern Analysis and Machine Intelligence*, 23(1), January 2001.

[4] http://www.ptgrey.com/products/bumblebee/index.html.

[5] N. Jojic, B. Brumitt, B. Meyers, and S. Harris. Detecting and estimating pointing gestures in dense disparity maps. In *IEEE International Conference onFace and Gesture recognition*, Grenoble, France, 2000.

[6] K. Nickel and R. Stiefelhagen. Pointing gesture recognition based on 3d-tracking of face, hands and head orientation. In *International Conference on Multimodal Interfaces*, pages 140–146, Vancouver, Canada, 2003.

[7] A. Wilson and N. Oliver. Gwindows:robust stereo vision for gesture-based control of windows. In *Proceedings of International Conference on Multimodal Interfaces*, 2003.

algorithm to work. The image is normalized in scale and orientation by a face tracker. Each pixel in the face image is associated to an appearance cluster. One particular cluster stands for salient robust face features which are: eyes, nose, mouth, chin. We have tried to extract and exploit the maximum of informations contained on a single image of a face and to limit the loss of generality.

These regions can be delimited with rectangles in the image. Identifying facial features using positions relative to the face image is difficult because of multiple variations of features possible. These variations are due to changing orientation, emotion and especially identity. A Bayesian classifier is used to identify the regions. Eyes have been found to be the most salient of the facial features. They can be used to obtain a coarse estimation of the horizontal pose, but are not sufficient to compute vertical pose. Because of high variations of the features in the face with regard to the identity and the pose, vertical pose is difficult to compute accurately.

Mean error for the horizontal pose does not vary with vertical angle. Error reaches 5 to 15 degrees when $|\alpha_h| <= 45^{\mathrm{O}}$, but increases when $|\alpha_h| > 45^{\mathrm{O}}$. This is due to the fact the horizontal angle is hard to estimate with only one eye visible on the image and that the neck detection disrupts the face estimation. Pieces of hair can also be misclassified as eyes too. All these observations tend to show that the robustness to identity is the most difficult criteria to respect.

References

[1] M. Zobel, A. Gebhard, D. Paulus, J. Denzler, H. Niemann "Robust Facial Feature Localization by Coupled Features" *4th International Conference on Automatic Face and Gesture Recognition*, March 28-30/2000 Grenoble, France pp. 2-7.

[2] D. Hall, J.L. Crowley "Computation of Generic Features for Object Classifi cation" *ScaleSpace*, 2003.

[3] S. J. McKenna, S. Gong, R. P. Wurtz, J. Tanner, D. Banin "Tracking Facial Feature Points with Gabor Wavelets and Shape Models" *1st Int. Conf. on Audio- and Videobased Biometric Person Authentication*, Lecture Notes in Computer Science 1997.

[4] D. Hall "Viewpoint Independant Object Recognition from Local Appearence" *PhD Thesis*, 2001.

[5] K. Schwerdt, J.L. Crowley "Robust face Tracking using Color" *Automatic face and Gesture Recognition*, pp 90-95, 2000.

[6] L. Wiskott, J-M. Fellous, N. Kruger, C. von der Malsburg "Face Recognition by Elastic Bunch Graph Matching" *Pattern Analysis and Machine Intelligence*, Vol 19 pp 775-779, 1997.

[7] J. Malik, S. Belongie, T. Leung, J. Shi "Contour and Texture Analysis for Image Segmentation" *IJCV*, Vol 43 pp 7-27, 2001.

[8] V. Kruger, G. Sommer "Gabor Wavelets Networks for Object Representation and Face Recognition" *Deutsche Arbeitsgemeinschaft fur Mustererkennung*, Vol 22, Sept. 13-15, 2000.

[9] C. Schmid "Constructing Models for Content-Based Image Retrieval" *Computer Vision and Pattern Recognition*, 2001.

[10] D. Lisin, E. Risemann, A. Hanson "Extracting Salient Image Features for Reliable matching using Outlier Detection Techniques" *Computer Vision Systems Third International Conference*, pp 481-491, 2003.

[11] M. Turk, A. Pentland "Eigenfaces for Recognition" *Cognitive Neuroscience*, Vol 3(1), pp 71-96, 1991.

[12] J.L. Crowley, O. Riff "Fast Computation of Scale Normalised Receptive Fields" *International Conference Scale Space*, June 2003, Island of Skye.

[13] Y. Wu, K. Toyama "Wide-Range, Person- and Illumination-Insensitive Head Orientation Estimation" *4th International Conference on Automatic Face and Gesture Recognition*, March 2000 Grenoble, France pp. 183-188.

[14] A.C, Varchmin, R. Rae, H. Ritter "Image based Recognition of gaze Direction using Adaptative methods" *International Gesture Workshop*, Springer, 1997, pp. 245-257.

[15] A.J. Howell, H. Buxton "Active Vision Techniques for Visually Mediated Interaction" *Image and Vision Computing 20(12)*, 2002, pp.861-871.

[16] T. Otsuka, J. Ohya "Real-time Estimation of Head Motion Using weak Perspective Epipolar Geometry" *Proceedings of WACV'98*, October 1998, Princeton.

[17] K.N. Walker, T.F. Cootes, C.J Taylor "Automatically Building Appearance Models from Image Sequences using Salient Features" *IVC(20), No. 5-6*, 15 April 2002, pp. 435-440.

[18] A.M. Martinez and R. Benavente "The AR Face Database" *CVC Technical Report n.24*, June 1998.

[19] G.J. Klinker, S.A. Shafer, T. Kanade "A Physical Approach to Color Image Understanding" *IJCV 1990*

[20] T. Lindeberg "Feature Detection with Automatic Scale Selection" *IJCV 1998 (30) Number 2*, pp. 79-116.

A Natural Interface to a Virtual Environment through Computer Vision-estimated Pointing Gestures

Thomas B. Moeslund, Moritz Störring, and Erik Granum

Laboratory of Computer Vision and Media Technology
Aalborg University, Niels Jernes Vej 14
DK-9220 Aalborg East, Denmark
Email:{tbm,mst,eg}@cvmt.auc.dk

Abstract. This paper describes the development of a natural interface to a virtual environment. The interface is through a natural pointing gesture and replaces pointing devices which are normally used to interact with virtual environments. The pointing gesture is estimated in 3D using kinematic knowledge of the arm during pointing and monocular computer vision. The latter is used to extract the 2D position of the user's hand and map it into 3D. Off-line tests of the system show promising results with an average errors of 76*mm* when pointing at a screen 2*m* away. The implementation of a real time system is currently in progress and is expected to run with 25*Hz*.

1 Introduction

In recent years the concept of a virtual environment has emerged. A virtual environment is a computer generated world wherein everything imaginable can appear. It has therefore become known as a virtual world or rather a virtual reality (VR). The 'visual entrance' to VR is a screen which acts as a window into the VR. Ideally one may feel immersed in the virtual world. For this to be believable a user is either to wear a head-mounted display or be located in front of a large screen, or even better, be completely surrounded by large screens.

The application areas of VR are numerous: training (e.g. doctors training simulated operations [13], flight simulators), collaborative work [9], entertainments (e.g. games, chat rooms, virtual museums [17]), product development and presentations (e.g. in architecture, construction of cars, urban planning [12]), data mining [3], research, and art. In most of these applications the user needs to interact with the environment, e.g. to pinpoint an object, indicate a direction, or select a menu point. A number of pointing devices and advanced 3D mouses (space mouses) have been developed to support these interactions. As many other technical devices we are surrounded with, these interfaces are based on the computer's terms which many times are not natural or intuitive to use. This is a general problem of Human Computer Interaction (HCI) and is an active research area. The trend is to develop interaction methods closer to those used in human-human interaction, i.e. the use of speech and body language (gestures) [15].

At the authors' department a virtual environment in the form of a six sided VR-CUBE[1], see figure 1, has been installed. A Stylus [19] is used as pointing device when interacting with the different applications in the VR-CUBE (figure 1 b). The 3D position and orientation of the Stylus is registered by a magnetic tracking system and used to generate a bright 3D line in the virtual world indicating the user's pointing direction, similar to a laser-pen.

In this paper we propose to replace pointing devices, such as the Stylus, with a computer vision system capable of recognising natural pointing gestures of the hand without the use of markers or other special assumptions. This will make the interaction less cumbersome and more intuitive. We choose to explore how well this may be achieved using just one camera. In this paper we will focus on interaction with only one of the sides in the VR-CUBE. This is sufficient for initial feasibility and usability studies and expendable to all sides by using more cameras.

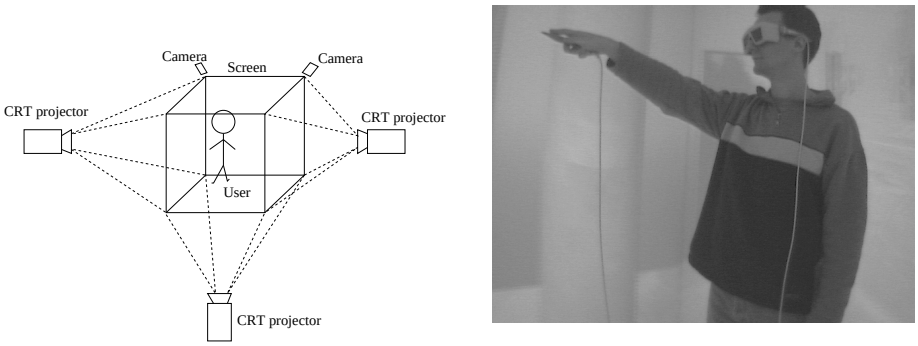


Fig. 1. VR-CUBE: a) Schematic view of the VR-CUBE. The size is 2.5 x 2.5 x 2.5*m*. Note that only three of the six projectors and two of the four cameras are shown. b) User inside the VR-CUBE interacting by pointing with a Stylus held in the right hand.

2 Pointing Gesture

The pointing gesture belongs to the class of gestures known as *deictic gestures* which MacNeill [16] describes as "gestures pointing to something or somebody either concrete or abstract". The use of the gesture depends on the context and the person using it [14]. However, it has mainly two usages: to indicate a direction or to pinpoint a certain object. A direction is mainly indicated by the orientation of the lower arm.

The direction when pinpointing an object depends on the user's distance to the object. If an object is close to the user the direction of the index finger is used. This idea is

[1] A VR-CUBE is a comparable installation to a CAVE^{TM}(CAVE Automatic Virtual Environment) [5] of the Electronic Visualization Laboratory, University of Illinois at Chicago.

used in [6] where an active contour is used to estimate the direction of the index finger. A stereo setup is used to identify the object the user is pointing to.

In the extreme case the user actually touches the object with the index finger. This is mainly used when the objects the user can point to are located on a 2D surface (e.g. a computer screen) very close to the user. In [20] the user points to text and images projected onto a desk. The tip of the index finger is found using an infra-red camera.

In [4] the desk pointed to is larger than the length of the user's arm and a pointer is therefore used instead of the index finger. The tip of the pointer is found using background subtraction.

When the object pointing to is more than approximately one meter away the pointing direction is indicated by the line spanned by the hand (index finger) and the visual focus (defined as the centre-point between the eyes). Experiments have shown that the direction is consistently (for individual users) placed just lateral to the hand-eye line [21]. Whether this is done to avoid occluding the object or as a result of the proprioception is unknown. Still, the hand-eye line is a rather good approximation. In [11] the top point on the head and the index finger are estimated as the most extreme points belonging to the silhouette of the user. Since no 3D information is available the object pointing toward is found by searching a triangular area in the image defined by the two extreme points.

In [10] a dense depth map of the scene wherein a user is pointing is used. After a depth-background subtraction the data are classified into points belonging to the arm and points belonging to the rest of the body. The index finger and top of the head are found as the two extreme points in the two classes.

In [7] two cameras are used to estimate the 3D position of the index finger which is found as the extreme point of the silhouette produced utilising IR-cameras. During an initialisation phase the user is asked to point at different marks (whose positions are known) on a screen. The visual focus point is estimated as the convergence point of lines spanned by the index-finger and the different marks. This means that the location of the visual focus is adapted to individual users and their pointing habit. However, it also means that the user is not allowed to change the body position (except for the arm, naturally) during pointing.

2.1 Context

In our scenario the distance between the user and the screen is approximately 1-2 meter. Objects can be displayed to appear both close to and far from the user, e.g. 0.1 or 10 meters away, thus both cases mentioned above might occur. However, pointing is mainly used when objects appear to be at least 2 meters away, hence the pointing direction is indicated by the line spanned by the hand and the visual focus.

The user in the VR-CUBE is wearing stereo-glasses, see figure 1 b). A magnetic tracker is mounted on these glasses. It measures the 3D position and orientation of the user's head which is used to update the images on the screen from the user's point of view. One could therefore simply use the position and orientation of the tracker as the pointing direction. However, conscious head movements for pointing has shown to be rather unnatural and will possibly transform the carpal-tunnel syndrome problem into the neck region [1]. Furthermore, due to the Midas Touch Problem [1] it is not as

practical as it sounds. However, the 3D position of the tracker can be used to estimate the visual focus and therefore only the 3D position of the hand needs to be estimated in order to calculate the pointing direction. This could then be used to replace pointing devices with a natural and more intuitive action - the pointing gesture.

Estimating the exact 3D position of the hand from just one camera is a difficult task. However, the required precision can be reduced by making the user a 'part' of the system feedback loop. The user can see his pointing direction indicated by a 3D line starting at his hand and pointing in the direction the system 'thinks' he is pointing. Thus, the user can adjust the pointing direction on the fly.

2.2 Content of the Paper

The remaining part of this paper is structured as follows. In section three the method used to estimate the pointing gesture is presented. Section four presents the experiments carried out to test the proposed method. Finally the method and results are discussed in section five.

3 Method

Since we focus on the interaction with only one side we assume that the user's torso is fronto-parallel with respect to the screen. That allows for an estimation of the position of the shoulder based on the position of the head (glasses). The vector between the glasses and the shoulder is called displacement vector in the following. This is discussed further in section 4.2. The pointing direction is estimated as the line spanned by the hand and the visual focus. In order to estimate the position of the hand from a single camera we exploit the fact that the distance between the shoulder and the hand (denoted R), when pointing, is rather independent of the pointing direction. This implies that the hand, when pointing, will be located on the surface of a sphere with radius R and centre in the user's shoulder (X_S, Y_S, Z_S):

$$(X - X_S)^2 + (Y - Y_S)^2 + (Z - Z_S)^2 \;=\; R^2 \tag{1}$$

These coordinates originate from the cave-coordinate system which has its origin in the centre of the floor (in the cave) and axes parallel to the sides of the cave. Throughout the rest of this paper the cave coordinate system is used.

The camera used in our system is calibrated[2] to the cave coordinate system. The calibration enables us to map an image point (pixel) to a 3D line in the cave coordinate system. By estimating the position of the hand in the image we obtain an equation of a straight line in 3D:

$$\boldsymbol{P}(t) = \mathbf{P}_0 + t \cdot \mathbf{D} \;\Rightarrow\; \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \end{bmatrix} + t \cdot \begin{bmatrix} D_1 \\ D_2 \\ D_3 \end{bmatrix} \tag{2}$$

[2] We use Tsai's calibration method [22] with full optimisation

where $\mathbf{P}_0$ is the optical centre of the camera and $\mathbf{D}$ is the direction unit vector of the line.

The 3D position of the hand is found as the point where the line intersects the sphere. This is obtained by inserting the three rows of equation 2 into equation 1 resulting in a second order equation in t. Complex solutions indicate no intersection and are therefore ignored. If only one real solution exist we have a unique solution, otherwise we have to eliminate one of the two solutions.

A solution which is not within the field-of-view with respect to the orientation of the tracker is eliminated. If further elimination is required we use prediction, i.e. to choose the most likely position according to previous positions. This is done through a simple first order predictor. The pointing direction is hereafter found as the line spanned by the non-eliminated intersection point and the visual focus point. The line is expressed as a line in space similar to the one in equation 2. For a pointing direction to be valid the position of the tracker and the hand need to be constant for a certain amount of time.

3.1 Estimating the 2D Position of the Hand in the Image

The VR-CUBE at the authors' department is equipped with four miniature s-video cameras which are placed in its four upper corners. They may be used for usability studies and for computer vision based user interfaces. The only illumination sources during image capture are the CRT-projectors[3], which are back-projecting images with 120Hz on the six sides of the VR-CUBE, see figure 1. This gives a diffuse ambient illumination inside the VR-CUBE which changes its colour depending on the displayed images. The brightness inside the VR-CUBE is determined by the displayed images as well. The average brightness in a 'normal' application is 25 Lux, which is rather little for colour machine vision. The auto gain of the cameras is therefore set to maximum sensitivity, the shutter is switched off, and the maximum opening is used, which results in noisy images with little colour variations.

Hirose *et al.* [9] recently proposed a system to segment the user in a VR-CUBE from the background in order to generate a video avatar. They used infrared cameras to cope with the poor light conditions and simulate a reference background image which is then subtracted from the infrared image containing the user. They get satisfying results.

The simulation of the background also gives information about the illumination the user is exposed to. This could be used, e.g. to estimate an intensity threshold for segmenting the user. However, due to the orientation of the cameras in the VR-CUBE this would be calculation intensive because the cameras' field of view covers parts of three sides, that means a background image has to be synthesised. Furthermore, the image processing is taking place on another computer, thus a lot of data would have to be transfered.

In this project we are using one of the s-video cameras and *a priori* knowledge about the scenario in the camera's field of view:

– Only one user at a time is present in the VR-CUBE

[3] Cathode Ray Tube projector. Each projector consists of three CRTs. One for red, green, and blue, respectively. The VR-CUBE is equipped with ELECTRICHOME MARQUEE® projectors

– The 3D position and orientation of the user's head is known by a magnetic tracker
– The background is brighter than the user, because an image is back-projected on each side and the sides have, especially at the shorter wavelengths, a higher reflectance than human skin
– Skin has a good reflectance for long wavelengths

Figure 2 shows the algorithm to segment the user's hand and estimate its 2D position in the image. Firstly the image areas where the user's hand could appear when pointing are estimated using the 3D position and orientation of the user's head (from the magnetic tracker), a model of the human motor system and the kinematic constraints related to it, and the camera parameters (calculating the field of view). Furthermore, a first order predictor [2] is used to estimate the position of the hand from the position in the previous image frame. In the following we will, however, describe our algorithm on the entire image for illustrative purposes.

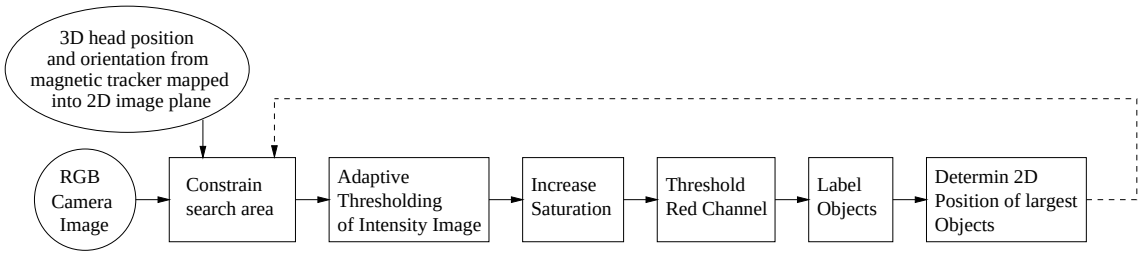
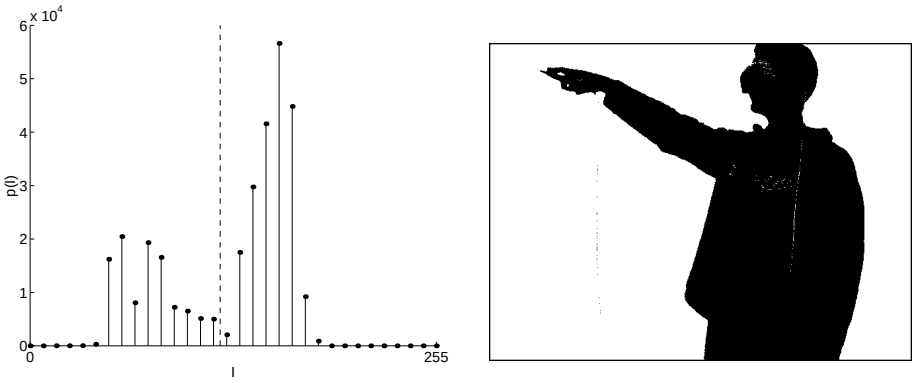


Fig. 2. Segmentation algorithm for the 2D position estimation of the hand in the camera image.

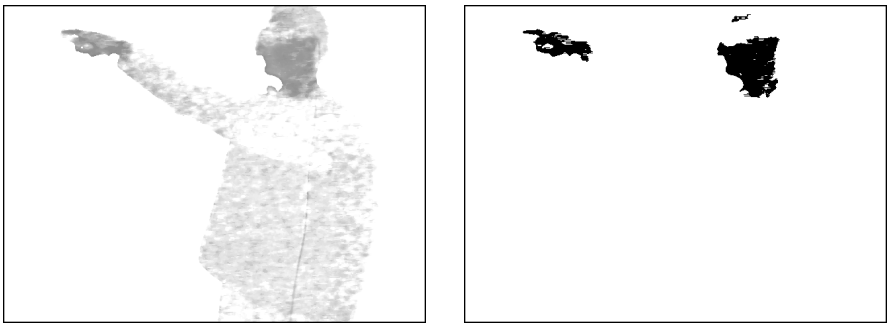
The histogram of the intensity image has a bimodal distribution, the brighter pixels originate from the background whereas the darker originate from the user, figure 3 a). This is used to segment the user from the background. The optimal threshold between the two distributions can be found by minimising the weighted sum of group variances [18]. The estimated threshold is indicated by the dashed line. Figure 3 b) is the resulting binary image after applying this threshold.

The colour variations in the camera image are poor. All colours are close the the gray vector. Therefore the saturation of the image colours is increased by an empirical factor. The red channel of the segmented pixels has maxima in the skin areas (figure 4 a) as long as the user is not wearing clothes with a high reflectance in the long (red) wavelengths. The histogram of the red channel is bimodal, hence it is also thresholded by minimising the weighted sum of group variances. After thresholding a labelling [8] is applied. Figure 4 b) shows the segmentation result of the three largest object. As the position of the head is known the associable skin areas are excluded. The remaining object is the user's hand. Its position in the image is calculated by the first central moments (centre of mass) [8].



a b

Fig. 3. Segmentation of the user. a) Histogram of the intensity image. The dashed line is the threshold found by minimisation of the weighted sum of group variance. b) Thresholded image.



a b

Fig. 4. a) Red channel of the pre-segmented camera image. b) Thresholded red channel after labelling the three largest objects. The gray values of the images are inverted for representation purpose.

4 Experimental Evaluation

This section presents the experimental evaluation of the different parts of the system. First the accuracy of pointing as described in section 3 is tested. Secondly the segmentation of the hand (section 3.1) is tested. The implementation of a real time system is currently in progress, thus test with visual feedback for the user are not yet available.

4.1 Segmentation of the Hand in the Camera Image

Several image sequences of users (Caucasian race) pointing inside the VR-CUBE were taken under different applications, hence different backgrounds and illumination conditions. The 2D position estimation of the hand has been tested off-line on these sequences (figure 4). Only qualitative results are available until now. The 2D position estimation works robustly if a mixture of colours is displayed, which is the case in the majority of the applications. The skin segmentation fails if the displayed images are too dark or if one colour is predominant, e.g. if the red CRT-projector is not used at all for display the measurements of the red channel of the camera become too noisy.

The implementation of a real time system is currently in progress. The calculation intensive part is the 2D estimation of the hand position which is working in a first non-optimised version on entire images (without reducing to regions of interest) with $10Hz$ on 320x240 pixels images on a $450MHz$ Pentium III^{TM}. We expect to get $25Hz$ after introducing the reduced search area and optimising the code.

4.2 Pointing Experiments without Visual Feedback

This subsection describes pointing experiments and their results, which were done to evaluate the accuracy of the pointing direction estimation described in section 3. A user was asked to point to 16 different points displayed on the screen as shown in figure 5. No visual feedback was given during these experiments, hence the user should be unbiased and show a natural pointing gesture. Experiments with five different user were done. An image of each pointing gesture was taken together with the data of the magnetic head tracker. The displacement vector between the head tracker and the shoulder was measured for each user.

During the evaluation of the data it turned out that the uncertainty of the position estimate of the head from the magnetic tracker was up to $15cm$ in each direction. It is at the moment not possible to calibrate the device in order to achieve a higher accuracy. This error is too large to be used as head position information in the method described in the previous section. In order to get a more accurate 3D position of the users' heads the visual focus point was segmented in the image data and together with the X position of the tracker, the 3D position of the visual focus point was estimated. This position was then used to estimate the position of the shoulder by the displacement vector as described in section 3. Figure 6 a) shows the results of a representative pointing experiment. The circles ($\circ$) are the real positions displayed on the screen and the asterisks ($*$) connected by the dashed line are the respective estimated positions where the user is pointing to. The error in figure 6 a) is up to $0.7m$. There are no estimates for the column

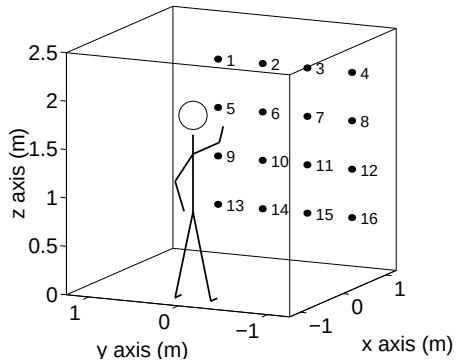


Fig. 5. Experimental setup for pointing experiments without visual feedback in the VR-CUBE. The user has a distance of approximately 2*m* from the screen where 16 points in a 0.5*m* raster are displayed.

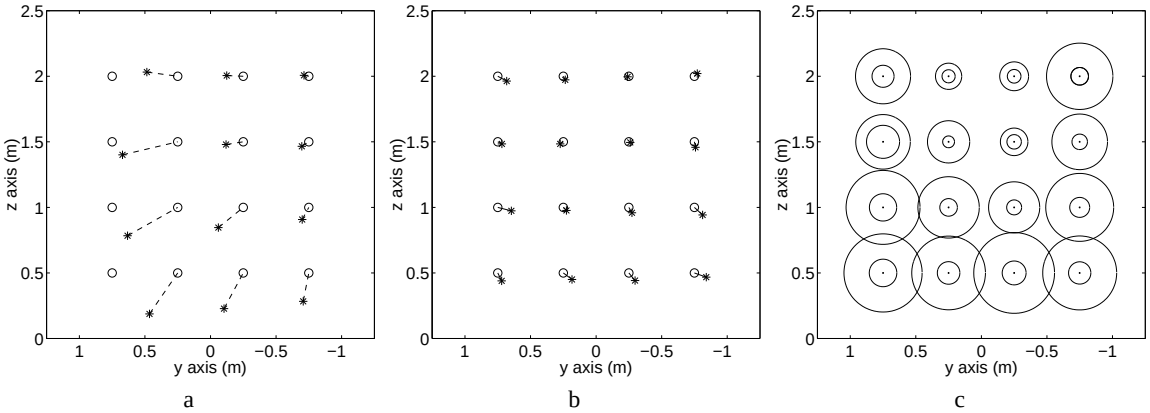


Fig. 6. Results from pointing experiments. The circles in the two first figures are the real positions on the screen. The asterisks are the estimated pointing directions from the system. a) The results of a representative user, using a constant displacement vector. b) The results of a representative user, using a LUT for the displacement vector. c) The inner circle shows the average error of all experiments. The outer circle shows the maximum error of all experiments.

to the left because there is no intersection between the sphere in equation 1 and the line spanned by the camera and the hand of the user.

The error is increasing the more the user points to the left. This is mainly due to the incorrect assumption (made in section 3) that the displacement vector is constant. The direction and magnitude of the displacement vector between the tracker and shoulder is varying. This is illustrated in figure 7.

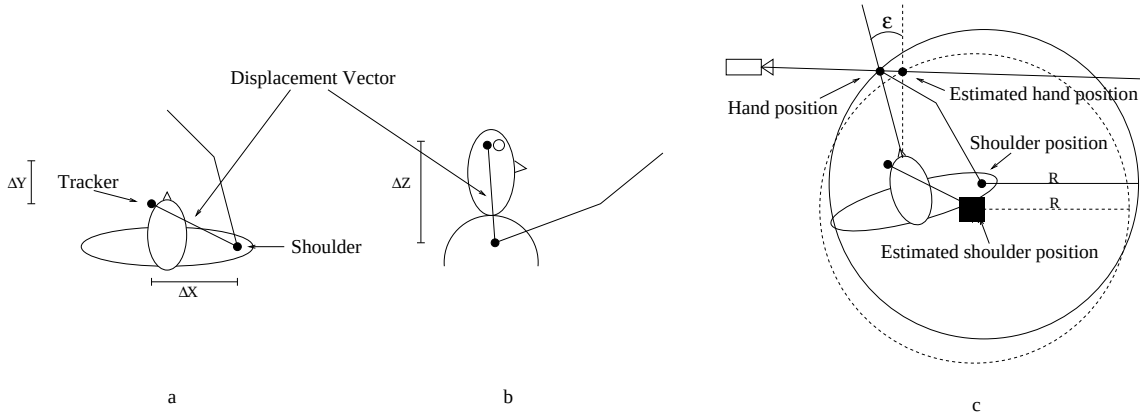


Fig. 7. a+b) The user and the displacement vector between the tracker and shoulder seen from above (a) and from the the right side (b). c) An illustration of the error introduced by assuming the torso to be fronto-parallel.

Figure 7.a and 7.b illustrate the direction and magnitude of the displacement vector between the tracker and shoulder when the user's head is looking straight ahead. As the head is rotated to the left the shoulder is also rotated as illustrated in figure 7.c. This results in a wrong centre of the sphere and therefore a wrong estimation of the 3D hand position. The error is illustrated as the angle ε. Beside the rotation the shoulder is also squeezed which makes the relation between the tracker (head) rotation and the displacement vector non-linear.

Figure 8 shows the components of the displacement vector for the 16 test-points in figure 5 (for a representative user) estimated from the shoulder position in the image data and the tracker data. For each user a lookup table (LUT) of displacement vectors as a function of the head rotation was build. Figure 6 b) shows the result of a representative pointing experiment (same as used in figure 6 a) using a LUT of displacement vectors to estimate the 3D position of the shoulder. Notice that after the position of the shoulder has been correction estimates for the left column is available.

Table 1 shows the average errors and the maximum errors of the five pointing experiments in *mm* for the respective points on the screen. These errors are also illustrated in figure 6 c) where the inner circle indicates the average errors and the outer circle the maximum errors. The average error of all points in all experiments is 76*mm*.

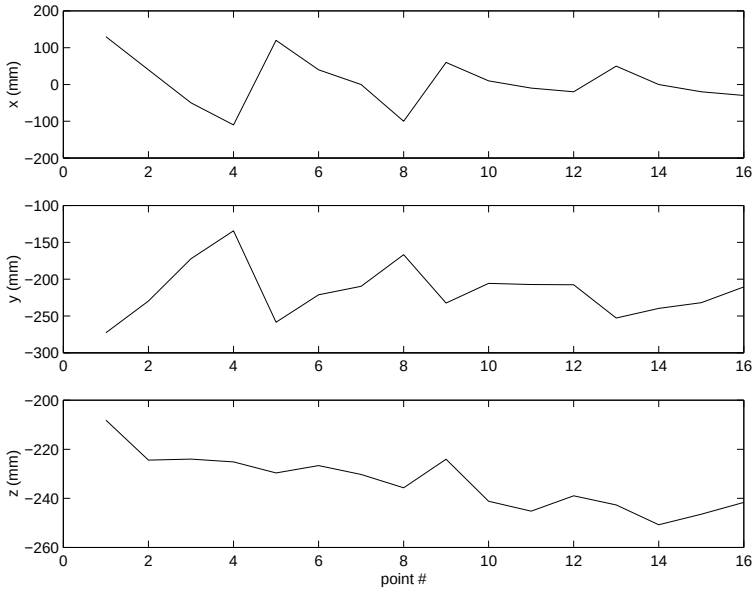


Fig. 8. Components (x,y,z) of the displacement vector as a function of the test-points in figure 5.

Table 1. Average errors and (maximum errors) in *mm* for the respective points on the screen.

	y axis			
z axis	750	250	-250	-750
2000	84 (210)	50 (100)	52 (110)	67 (253)
1500	126 (208)	45 (161)	55 (105)	59 (212)
1000	104 (282)	67 (234)	57 (195)	76 (259)
500	105 (298)	86 (281)	91 (308)	85 (282)

5 Discussion

In this paper we have demonstrated that technical interface devices can be replaced by a natural gesture, namely finger pointing. The pointing gesture is estimated as the line spanned by the 3D position of the hand and the visual focus, defined as the centre point between the eyes. The visual focus point is at the moment estimated from the image data and a X measure. In the future this should be given from the position and orientation of the electro magnetic tracker mounted on the stereo glasses worn by the user. The 3D position of the hand is estimated as the intersection between a 3D line spanned by the hand and camera, and a sphere with centre in the shoulder of the user and radius equal to the length of the user's arm when pointing, R. Pointing experiments with five different user were done. Each user was asked to point to 16 points at a screen in $2m$ distance. Due to , especially, movements of the shoulder during pointing errors up to $700mm$ between the estimated and the real position on the screen was observed. To reduce the errors a LUT was used to correct the position of the shoulder. This reduced the average error to $76mm$ and the maximum error to $308mm$. This we find to be a rather accurate result given the user is standing two meters away. However, whether this error is too large depends on the application.

In the final system the estimated pointing direction will be indicated by a bright 3D line seen through the stereo glasses starting at the finger of the user and ending at the object pointed to. Thus, the error is less critical since the user is part of the system loop and can correct on the fly. In other words, if the effect of the error do not hinder the user in accurate pointing (using the feedback of the 3D line), then they may be acceptable. However, if they do or if the system is to be used in applications where no feedback is present, e.g. in a non-virtual world, then we need to know the effect of the different sources of errors and how to compensate for them.

The error originates from five different sources: the tracker, the image processing, the definition of the pointing direction, the assumption of the torso being fronto-parallel with respect to the screen, and the assumption that R is constant.

Currently we are deriving explicit expressions for the error sources presented above and setting up test scenarios to measure the effect of these errors. Further experiments will be done in the VR-CUBE to characterise the accuracy and usability as soon as the real time implementation is finished. The experiments will show whether the method allows us to replace the traditional pointing devices as is suggested by our off-line tests.

Another issue which we intend to investigate is the Midas Touch Problem - how to inform the system that a pointing gesture is present. In a simple test scenario with only one gesture - pointing, it is relatively easy to determine when it is performed. As mentioned above (see also [10]) the gesture is recognised when the position of the hand is constant for a number of frames. However, in more realistic scenarios where multiple gestures can appear, the problem is more difficult. One type of solution is presented in [7] where the thumb is used as a mouse bottom. Another, and more natural, is to acommandate the gesture with a spoken input [4], e.g. ”select that (point) object”. Which path we will follow is yet to be decided.

References

1. L. Bakman, M. Blidegn, and M. Wittrup. Improving Human-Computer Interaction by adding Speech, Gaze Tracking, and Agents to a WIMP-based Environment. Master's thesis, Aalborg University, 1998.
2. Y. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association*. Academic Press, INC., 1988.
3. M. Böhlen, E. Granum, S. Lauritzen, and P. Mylov. 3d visual data mining. http://www.cs.auc.dk/3DVDM/.
4. T. Brøndsted, L. Larsen, M. Manthey, P. McKevitt, T. Moeslund, and K. Olesen. The Intellimedia WorkBench - an environment for building multimodal systems. In *second international Conference on Cooperative Multimodal Communication*, 1998.
5. D. Browning, C. Cruz-Neira, D. Sandin, and T. A. DeFanti. Virtual reality: The design and implementation of the cave. In *SIGGRAPH'93 Computer Graphics Conference*, pages 135–142. ACM SIGGRAPH, Aug. 1993.
6. R. Cipolla, P. Hadfield, and N. Hollinghurst. Uncalibrated Stereo Vision with Pointing for a Man-Machine Interface. In *IAPR Workshop on Machine Vision Applications*, Yokohama, Japan, December 1994.
7. M. Fukumoto, Y. Suenaga, and K. Mase. "Finger-Pointer": Pointing Interface By Image Processing. *Computer & Graphics*, 18(5), 1994.
8. R. C. Gonzalez and P. Wintz. *Digital Image Processing*. ADDISON-WESLEY PUBLISHING COMPANY, 1987.
9. M. Hirose, T. Ogi, and T. Yamada. Integrating live video for immersive environments. *IEEE MultiMedia*, 6(3):14–22, July 1999.
10. N. Jojic, B. Brumitt, B. Meyers, S. Harris, and T. Huang. Detection and Estimation of Pointing Gestures in Dense Disparity Maps. In *The fourth International Conference on Automatic Face- and Gesture-Recognition*, Grenoble, France, March 2000.
11. R. Kahn and M. Swain. Understanding People Pointing: The Perseus System. In *International Symposium on Computer Vision*, Coral Gables, Florida, November 1995.
12. E. Kjems. Creating 3d-models for the purpose of planning. In *6th international conference on computers in urban planning & urban management*, Venice, Italy, Sept. 1999.
13. O. Larsen, J. Haase, L. Østergaard, K. Hansen, T. Søndergaard, and H. Nielsen. The Virtual Brain Project - Development of a Neurosurgical Simulator. Accepted for The 9th Annual Medicine Meets Virtual Reality, January 2001.
14. E. Littmann, A. Drees, and H. Ritter. Neural Recognition of Human Pointing Gestures in Real Images. *Neural Processing Letters*, 3:61–71, 1996.
15. B. MacIntyre and S. Feiner. Future multimedia user interfaces. *Multimedia Systems*, 4:250–268, 1996.
16. D. MacNeill. *Hand and mind: what gestures reveal about thought*. University of Chicago Press, 1992.
17. K. Mase and R. Kadobayashi. Gesture Interface for a Virtual Walk-through. In *Workshop on Perceptual User Interface*, 1997.
18. N. Otsu. A thresholding selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9:62–66, 1979.
19. Polhemus. Stylus magnetic tracker. http://www.polhemus.com/stylusds.htm.
20. Y. Sato, Y. Kobayashi, and H. Koike. Fast Tracking of Hands and Fingertips in Infrared Images for Augmented Desk Interface. In *The fourth International Conference on Automatic Face- and Gesture-Recognition*, Grenoble, France, March 2000.
21. J. Taylor and D. McCloskey. Pointing. *Behavioural Brain Research*, 29:1–5, 1988.

22. R. Y. Tsai. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal of Robotics and Automation*, 3(4):323–344, August 1987.

Skin colour cue for computer vision

Moritz Störring
Computer Vision and Media Technology Laboratory
Aalborg University, Niels Jernes Vej 14
DK-9220 Aalborg East, Denmark
mst@cvmt.aau.dk

Artificial and ambient intelligence is becoming ubiquitous in our daily-life and requires more intuitive and user-friendly human computer interaction than traditional interfaces like keyboard and mice. The use of face and gesture recognition will therefore be common in future interfaces, and the most unobtrusive technology to achieve such interfaces are computer vision based methods. A crucial processing step for the success of such systems is robust detection and tracking of faces and hands, which is frequently done by combining complementary cues, e.g., motion, shape, and colour [1, 4]. Comprehensive surveys on the current state-of-the-art of face detection methods were recently published in [2] and [5].

With increasing quality of colour video cameras and growing computation power skin colour is more and more used in face and hand detection because it is invariant to orientation and size, gives an extra dimension compared to grey scale methods, and is fast to process.

The main problems with the robustness of skin colour detection are however: (1) dependence on the illumination colour, (2) it varies between individuals, and (3) many everyday-life objects are skin colour like, i.e., skin colour is not unique [3].

This talk starts with a general review of the current state-of-the-art of computer vision based skin colour detection and modelling approaches, suggesting a taxonomy of the methods reported in the literature.

In the following of the talk the focus will be on methods for building an adaptive skin colour cue using physics-based approaches to model skin colour. The image formation process is investigated theoretically and experimentally with respect to human skin colours under changing and mixed illumination conditions.

It is shown that skin colour "perception" as viewed by a state-of-the-art colour video camera can be modelled sufficiently accurate with a physics-based approach given the illumination spectra, the reflectance of skin, and the camera characteristics. Furthermore, everyday-life illumination spectra can be modelled appropriately as Blackbody radiators in this context. This skin colour modelling may provide the basis for applications such as adaptive skin segmentation.

For adaptive segmentation it may also be useful to estimate the illumination colour. Two methods are suggested and tested to estimate the illumination colour from observation of skin colour. The first uses the diffuse reflections from skin and the second uses the surface or highlight reflections. These methods are complementary and their accuracies are sufficient to improve adaptive skin segmentation.

In order to track skin areas through changing illumination conditions and to distinguish them from other skin coloured objects a method is proposed to model the skin colour distribution as a unimodal Gaussian. The parameters of the Gaussian can be modelled selectively for arbitrary illumination using a physics-based approach.

The talk concludes with an outlook how an adaptive skin colour cue may be build that, in combination with other cues, will enable robust face and hand detection in unconstrained environments.

References

[1] J. L. Crowley. Integration and control of reactive visual processes. *J. of Robotics and Autonomous Systems*, 16(1):17–27, Nov. 1995.

[2] E. Hjelmas and B. K. Low. Face detection: A survey. *Computer Vision and Image Understanding*, 83(3):236–274, Sept. 2001.

[3] M. Störring. *Computer Vision and Human Skin Colour*. PhD thesis, Faculty of Engineering and Science, Aalborg University, Niels Jernes Vej 14, DK-9220 Aalborg, 2004.

[4] J. Triesch and C. von der Malsburg. Self-organized integration of adaptive visual cues for face tracking. In *4th IEEE Int. Conf. on Automatic Face- and Gesture-Recognition*, pages 102–107, Grenoble, France, Mar. 2000.

[5] M.-H. Yang, D. Kriegman, and N. Ahuja. Detecting faces in images: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):34–58, Jan. 2002.

HMM and IOHMM for the Recognition of Mono- and Bi-Manual 3D Hand Gestures

Agnès Just, Sébastien Marcel
Institut Dalle Molle d'Intelligence Artificielle Perceptive (IDIAP)
CH-1920 Martigny
just,marcel@idiap.ch (http://www.idiap.ch)
Olivier Bernier, Jean-Emmanuel Viallet
France Telecom Research & Development
FR-22300 Lannion
olivier.bernier@francetelecom.com
(http://www.rd.francetelecom.com)

Abstract

In this paper, we address the problem of the recognition of isolated complex mono- and bi-manual hand gestures. In the proposed system, hand gestures are represented by the 3D trajectories of blobs obtained by tracking colored body parts. In this paper, we study the results obtained on a complex database of mono- and bi-manual gestures. These results are obtained by using Input/Output Hidden Markov Model (IOHMM), implemented within the framework of an open source machine learning library, and are compared to Hidden Markov Model (HMM).

1 Introduction

Nowadays, Human-Computer Interaction (HCI) is usually done using keyboards, mice or graphic boards. The use of hand gestures for HCI can help people to communicate with computers in a more intuitive way. The potential power of gestures has already been demonstrated in applications that use the hand gesture input to control a computer while giving a presentation for instance. Other possible applications of gesture recognition techniques include computer-controlled games, teleconferencing, robotics or the manipulation of objects by CAD designers. In gestural HCI, the use of video cameras is more natural than any dedicated acquisition device (such as data-gloves for instance) but is also much more challenging. In video-based hand gesture recognition (HGR) it is necessary to distinguish two aspects of hand gestures: the *static* aspect and the *dynamic* aspect. The *static* aspect is characterized by a pose or configuration of the hand in an image. The *dynamic* aspect is defined either by the trajectory of the hand, or by the sequence of hand postures in a sequence of images.

There are two sub-problems to address when dealing with dynamic hand gesture recognition: segmentation and classification. Segmentation aims at identifying the beginning and/or the end of a gesture given a continuous stream of data. Usually, this stream of data is made by a random sequence of known gestures and non-gestures. And given an isolated gesture sequence, classification outputs the class the gesture belongs to.

In this paper, we will focus on the classification of isolated hand gestures. First, we present an overview of related work on HGR. In section 3, we describe our approach to capture mono- and bi-manual 3D hand gestures, and we describe the database. Then, we introduce Input/Output Hidden Markov Model (IOHMM) and we present experimental results. Finally, we discuss the results and conclude.

2 Related Work

Dynamic HGR is a **sequence processing** problem that can be accomplished by using various techniques. Darell and Pentland in [6] used a vision-based approach to model both object and behavior. The object views were represented using sets of view models. This approach allowed them to learn their model by observation. The disadvantage of this method is that complex articulated objects have a very large range of appearances. Therefore, they used a representation based on interpolation of appearance from a relatively small number of views. The gesture classification is performed by stereotypical space-time patterns (i.e. the gestures) matched with stored gesture patterns using dynamic time warping (DTW). This system was tested on only

two gestures. And images were focused on the hand. The experiment was also user-dependent since each of the seven users were involved in both the training and testing phases.

Finite state machine (FSM) was the first technique applied to sequence processing. It was applied to gestures by Davis and Shah [7]. Each gesture was decomposed into four distinct phases. Since the four phases occurred in fixed order, a FSM was used to guide the flow and to recognize seven gestures. Experiments were using a close-up on the hand. Hong et al. [9] proposed another approach based on FSM, that used 2D positions of the centers of the user's head and hands as features. Their system permitted to recognize in real-time four mono-manual gestures. But the most important technique, widely used for dynamic HGR, is Hidden Markov Models. This approach is inspired by the success of the application of HMMs both in speech recognition and in hand-written character recognition fields [1]. Starner and Pentland in [2] used an eight element feature vector consisting of each hand's x and y position, angle of axis of least inertia, and eccentricity of bounding ellipse. They used networks of HMMs to recognize a sequence of gestures taken from the American Sign Language. Training was performed by labeling each sign with the corresponding video stream. They used language modeling to segment the different signs. The Viterbi decoding algorithm was both used with and without a strong grammar based on the known form of the sentences. With a lexicon of forty words, they obtained 91,9% of accuracy in the test phase. Unfortunately, these results are almost impossible to reproduce.

More recently, Marcel et al. [4] have proposed Input/Output Hidden Markov Models (IOHMMs). An IOHMM is based on a non-homogeneous Markov chain where emission and transition probabilities depend on the input. On the opposite, HMMs are based on homogeneous Markov chains since the dynamic of the system is determined only by the transition probabilities which are time independent. Their system was able to recognize four types of gestures with 98,2% of accuracy. Furthermore, this database is publicly available from the Internet [1]. But in their article, they used IOHMM for a small gesture vocabulary (only four gestures).

In most of the studies on hand gestures, a small vocabulary has been used. In the next section, we describe a more important database for the recognition of mono- and bi-manual 3D hand gestures.

3 Mono- and Bi-manual 3D Hand Gestures

Most of human activities involve the use of two hands. Furthermore, gestures occur in a 3D space and not in a 2D image plane. In the proposed system, mono- and bi-manual hand gestures are represented by the 3D trajectories of blobs. Blobs are obtained by tracking colored body parts in real-time using the EM algorithm. This approach is similar to the statistical region approach for person tracking [5], for gesture recognition [10].

3.1 Tracking Blobs in 3D

A detailed description of the 3D blob tracking algorithm can be found in [3]. This algorithm tracks head and hands in near real-time (12Hz) using two cameras (Figure 1).

Figure 1. Left: left and right captured images. Center: left and right images with projected ellipsoids. Top right: ellipsoids projection on the frontal plane. Down right: ellipsoids projection on the side plane (the cameras are on the left side).

The algorithm is based on simple preprocessing followed by the use of a **statistical model** linking the observations (resulting from the preprocessing stage) to the parameters: the position of the hands and the head. Preprocessing consists of background subtraction followed by specific colors detection, using a simple color lookup table. The **statistical model** is composed of four ellipsoids, one for each hand, one for the head and one for the torso. Each one is projected on both camera planes as an ellipse. A Gaussian probability density function with the same center and size is associated with each ellipse. The parameters of the model (positions and orientations of the ellipsoids) are adapted to the pixels detected by the preprocessing stage. This adaptation simultaneously takes into account the detected pixels from the two cameras, and is based on the maximum likelihood principle. The EM algorithm is used to obtain the

[1] http://www.idiap.ch/~marcel/Databases/main.html

maximum of the likelihood.

3.2 Gesture Database

The database used in this paper has been obtained using the tracking method described above. The database consists of 16 gestures (Table 1) carried out by 20 different persons. Most of gestures are mono-manual and some are bi-manual (*fly*, *swim* and *clap*).

Name	Description	R	M/B
Stop/yes	Raised hand on the head level and facing palm		M
No/wipe	Idem with movements from right to left	R	M
Raise hand	Raised hand higher than the head		M
Hello/wave	Idem with movements from right to left	R	M
Left	Hand on the hip level, movements to the left	R	M
Right	Hand on the hip level, movements to the right	R	M
Up	Hand on the hip level, movements to the up	R	M
Down	Hand on the hip level, movements to the down	R	M
Front	Hand on the hip level, movements to the front	R	M
Back	Hand on the hip level, movements to the back	R	M
Swim	Swimming mimic gesture	R	B
Fly	Flying mimic gesture	R	B
Clap	On the torso level, clap the hands	R	B
Point left	On the torso level, point to the left		M
Point front	On the torso level, point to the front		M
Point right	On the torso level, point to the right		M

Table 1. Description of the 16 gestures. A hand gesture could involve one hand (Mono-manual) or both hands (Bi-manual). The gesture could be also a Repetitive movement such as *clap*.

The use of gloves with distinct colors permits to avoid occlusion problems that occur with bi-manual gestures. The person performing the gesture wears gloves of different colors and a sweat-shirt of a specific color different from the skin color and different from the glove colors in order to help the segmentation of hands, head and torso.

For each person and each gesture, there are 5 sessions and 10 shots per session. All the gestures start and end in the same rest position (the hands lying along the thighs). The temporal segmentation was manually accomplished after a recording session. For each gesture, a trajectory for each blob has been generated. Finally, the database is composed of 1000 trajectories per gesture. Gesture trajectories correspond to 3D coordinates of center of the head, of the two hands and of the torso. They are produced with the natural hand (left hand for left-handed and right hand for right-handed persons). For the left-handed persons, trajectories have been mirrored.

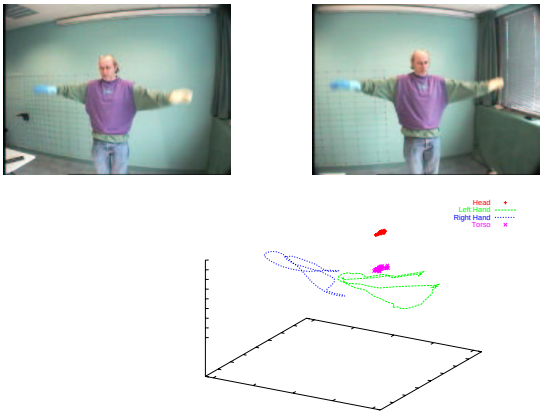


Figure 2. Left: Example of images of the "Vinci" sequence from the point of view of the left camera (on the left) and from the point of view of the right camera (on the right). Right: 3D coordinates of the center of each blob (head, torso, left hand and right hand) for a "swim" gesture.

Figure 3 shows an example of the swim gesture sequence from the point of view of the right camera. Furthermore, for each person and each session, a "Vinci" sequence has been recorded (Figure 2). This sequence gives the maximum arm spread. This figure presents also in a three dimensional space[2] the coordinates of the center of each blob (head, torso and hands) for a "swim" gesture sequence.

4 Hidden Markov Model versus Input Output Hidden Markov Model

4.1 Hidden Markov Model

A Hidden Markov Model (HMM) [1] is a statistical machine learning algorithm which models sequences of data. It consists of a set of N states, called hidden states because they are not observable. It also contains transition probabilities between these states and emission probabilities from the states to model the observations.

The data sequence is thus factorized over time by a series of hidden states and emission from these states. Let q_t be the state, y_t be the output (observation) at time t. The emission probability $P(y_t|q_t = i), \forall i = 1 \dots N$ depends only on the current state q_t. The transition probability between states $P(q_t = i|q_{t-1} = j), \forall i, j = 1 \dots N$ depends only on the previous state. Then, the training of a HMM can be carried out using the *Expectation-Maximization* (EM) algorithm [8].

[2] the z axis is the vertical axis of the person.

Figure 3. From top-left to bottom-right, a frame-by-frame decomposition of a "swim" gesture from the point of view of the right camera.

As we try to recognize 16 gesture classes, we have one HMM per class and we use a naive Bayes classifier to perform the classification.

4.2 Input Output Hidden Markov Model

An Input Output Hidden Markov Model (IOHMM) is an extension of the HMM described previously. First introduced by Bengio and Frasconi [11], IOHMMs are able to discriminate temporal sequences using a supervised training algorithm. IOHMMs map an input sequence to an output sequence. In our case, output sequence correspond to the class of the gesture. Let q_t be the state, x_t the input and y_t the output at time t. Thus q_t depends on q_{t-1} and x_t. y_t depends on q_t but also depends x_t (cf. Figure 4).

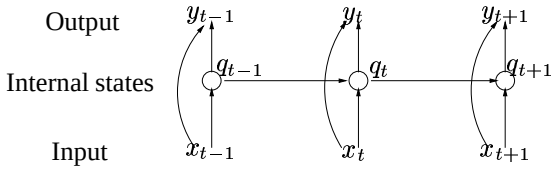


Figure 4. An IOHMM showing dependencies between the input x, output y and hidden states q of the model.

IOHMMs are composed of emission probabilities $P(y_t|q_t, x_t)$ and transition probabilities $P(q_t|q_{t-1}, x_t)$. They are time dependent since the emission and transition probabilities depend on x_t. Hence IOHMMs are based on non-homogeneous Markov chains contrary to HMMs. Consequently, the dynamic of the system is not fixed *a priori* such as in HMMs, but evolves in time and is function of the input sequence. The architecture of IOHMM also consists of a set of states q. With each state are associated two conditional distributions: one for transition probabilities and one for emission probabilities. The data sequences to model can be of two types: discrete or continuous. In the discrete case, codebooks or multinomial distributions can be used to model the conditional distributions. In the continuous case, models such as Multi Layer Perceptron [12] can be used to represent the conditional distributions. Another solution to deal with continuous observations is to perform a quantization in order to discretize the data. In an IOHMM, there are several ways to classify an input sequence $\mathbf{x}_1^T$. The most restrictive way is to compute $P(\mathbf{y}_1^T|\mathbf{x}_1^T)$, where $\mathbf{y}_1^T$ represents the output vector sequence. We can also compute the average $\frac{1}{N}\sum_{t=1}^{T} P(\mathbf{y}_t|\mathbf{x}_1^t)$ where N is the number of classes. The less restrictive method is to compute $P(\mathbf{y}_T|\mathbf{x}_1^T)$. In all these cases, the classification is achieved by finding the class c which maximizes the probability. Here we have chosen the class c such as $P(\mathbf{y}_1^T|\mathbf{x}_1^T)$ is maximum.

5 Experimental Results

In this section, we present baseline results obtained using HMMs and IOHMM on the proposed mono- and bi-manual database. In the case of IOHMM, we have conducted experiments using discrete conditional distributions. Thus we have performed a quantization step on the data. This quantization is explained later in this section. The open source machine learning library used for all experiments is Torch *http://www.torch.ch*.

	Training set	Validation set	Test set
minimum number of frames	12	6	10
average number of frames	25	24	28
maximum number of frames	64	71	89

Table 2. Minimum, average and maximum number of frames for the different subsets

5.1 Preprocessing the Database

Normalization: As a first step, a normalization has been performed on all gesture trajectories. We suppose that each gesture occurs in a cube centered on the torso and of vertex size the maximum spread given by the "Vinci" sequence. This cube is then normalized to reduce the vertex to one. Finally, the range of x, y and z coordinates varies between -0.5 and 0.5. The 3D coordinates of the head and torso are almost stationary. Thus, we keep the normalized 3D trajectories of both hands only. This leads to an input feature vector of size 6.

Feature Extraction: We also computed the difference between the coordinates of each hand for two consecutive frames. These features have been multiplied by 100 in order to have values with the same order of magnitude than x, y and z. The final feature vector is $[x_{left}, y_{left}, z_{left}, x_{right}, y_{right}, z_{right}, \Delta x_{left}, \Delta y_{left}, \Delta z_{left}, \Delta x_{right}, \Delta y_{right}, \Delta z_{right}]$ of size 12.

Quantization: The second step is the quantization of the data to efficiently use discrete IOHMM. The output sequence still encodes the gesture class: $y_t = \{0, \ldots, 15\}, \forall t$. In order to model more closely the class distribution of the data, we apply a K-means algorithm [13] class per class on the input features. For each gesture class, a K-means model with 75 clusters has been trained using the training set. The 16 resulting K-means models have been merged into a single one (1200 clusters). Finally, each frame of each sequence is quantized into one discrete value $\in [1 \ldots 1200]$ (which is the index of the nearest cluster).

5.2 Parameter Tuning

In our experiments, we have used a left/right topology for both the IOHMM and the 16 HMMs. In order to find the optimal hyper-parameters (number of states of the discrete IOHMM, number of states and number of Gaussians per state for the HMMs), the following experimental protocol has been used. For experimental purposes, the database has been split into three subsets: the training set T, the validation set V and the test set Te. T and V contain 5 subjects each. Te contains 10 subjects. For each subject, all recordings from all shots have been used. Table 2 provides the minimum/average and maximum number of frames per sequence for each subset of the database. Different possibilities for the number of states and for the number of Gaussians have been tried on T. The selection of the best parameters has been done on V. Finally, a model has been trained on both T and V and tested on Te. We obtained the best results with 5 states for the IOHMM, and 15 states and 1 Gaussian per state for each HMM.

5.3 Results

Figure 5 provides comparative results between discrete IOHMM and baseline continuous HMMs. HMM and IOHMM achieve respectively 64% and 74% average classification rate. From the results, we observe that bi-manual gestures are very well classified. Few mistakes happen between "swim" and "clap" gestures.

If we now have a look to the mono-manual gestures, we notice that first, there is a misclassification between the "stop", "no/wipe", "raise" and "hello/wave" gestures. If we refer to table 1, the only differences between these four gestures are the hand level and the oscillatory movement of the hand from the left to the right. Thus, HMMs have problems to model the oscillatory movement of the "no/wipe" and "hello" gestures. On the contrary, IOHMM has no real problem to model these oscillations (average recognition rate: 85%). With the HMMs, the "stop" and "no" gestures, such as the "raise" and "hello" gestures are misclassified one to the other. But still the non-oscillatory movements are misclassified in both cases, with the IOHMM and with the HMMs.

Let us consider the positioning gesture category ("left", "right", "up", "down", "front" and "back" gestures). The block around the diagonal of the matrix shows first that HMMs and IOHMM differentiate quite accurately this category of gestures from the others. It shows also that it has difficulties to provide the correct class within this category, even if IOHMM give better results than HMMs. Only the "left" and "right" gestures are well classified. For the others, it seems that the discriminant aspect of these gestures which is the dynamic of the hand (Table 1) is not sufficient for a good classification.

Finally, if we consider pointing gestures, HMMs and IOHMM differentiate also quite accurately this category of gestures from the others. But IOHMM give better results than HMMs as the recognition rate is around 70% for the 3 gestures, and only 60% for the HMMs. Concerning the "point left" gesture, HMMs misclassified it with the "point front" and "point right" gestures and IOHMM misclassified it only with the "point front"gesture. It shows that the location of the hand at the end of the pointing gesture is not precise enough to give a discriminant information to the IOHMM, and to the HMMs.

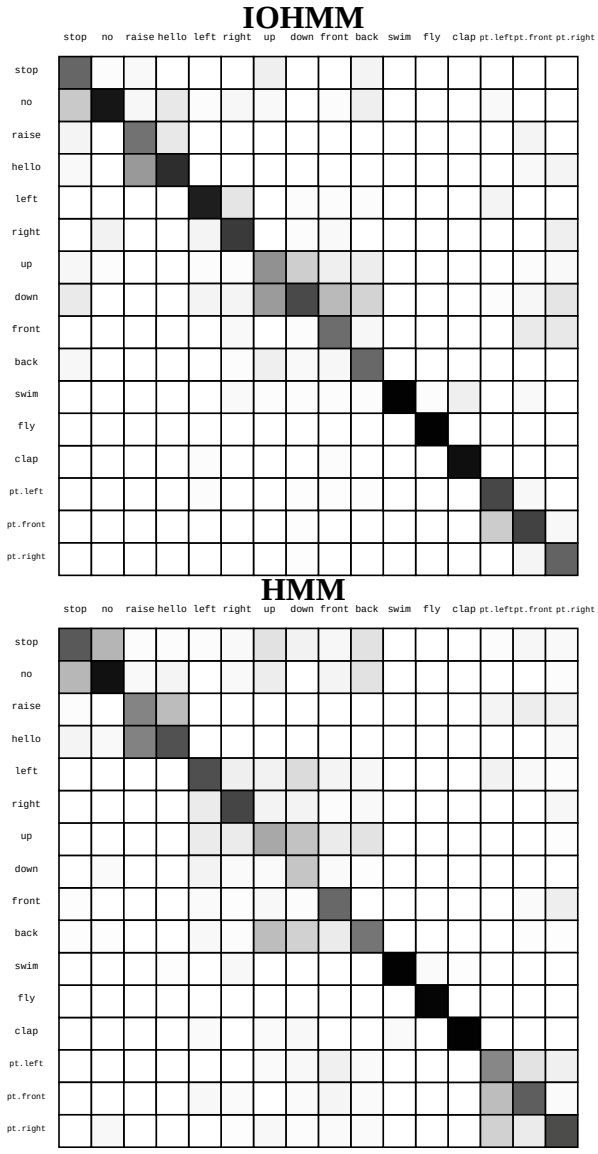


Figure 5. Confusion matrix for IOHMM and HMM on the test set (rows: desired, columns: obtained). Black squares correspond to the well-classified gestures.

6 Conclusions

In this paper, we addressed the problem of the recognition of isolated complex mono- and bi-manual hand gestures. Hand gestures were represented by the 3D trajectories of blobs obtained by tracking colored body parts.

We provide recognition results obtained on a complex database of 16 mono- and bi-manual gestures by two sequence processing algorithms, namely Input Output Hidden Markov Model (IOHMM) and Hidden Markov Model (HMM), implemented within the framework of an open source machine learning library. The obtained results are encouraging. Bi-manual gestures are very well classified, and mono-manual gestures are fairly classified. We conclude that IOHMM performs the best on this database. We will perform complementary experiments using continuous IOHMM to verify if this conclusion is still valid.

Acknowledgments

This research has been carried out in the framework of the GHOST project, funded by France Telecom R&D (project number 021B276). This work was also funded by the Swiss National Science Foundation through the National Center of Competence in Research (NCCR) on ”Interactive Multimodal Information Management (IM2)”. The authors wish to thank J. Guerin and B. Rolland for recording and annotating the gesture database.

References

[1] L. R. Rabiner, “A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition”, *Proceedings of the IEEE*, Vol. 77, Nb. 2, 1989.

[2] T. E. Starner and A. Pentland, “Visual Recognition of American Sign Language Using Hidden Markov Models”, *Int. Workshop on Automatic Face- and Gesture-Recognition*, 1995.

[3] O. Bernier and D. Collobert, “Head and Hands 3D Tracking in Real Time by the EM algorithm” *Proceeding of the IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems (RATFG-RTS’01)*, 2001.

[4] S. Marcel and O. Bernier and J.E. Viallet and D. Collobert, ”Hand Gesture Recognition using Input-Output Hidden Markov Models”, *Proc. of the FG’2000 Conference on Automatic Face and Gesture Recognition”*, 2000.

[5] C.R. Wren and A. Azarbayejani and T. Darrell and A. Pentland, ”Pfinder: Real-Time Tracking of the Human Body”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, pp. 780-785, 1997.

[6] T. Darrell and A. Pentland, “Space-time gestures”, *Proc. of the Conference on Computer Vision and Pattern Recognition*, pp. 335-340, 1993.

[7] J. Davis and M. Shah, ”Recognizing Hand Gestures”, *Proc. of European Conference on Computer Vision* Vol. 1, pp. 331-340, 1994.

[8] A.P.Dempster and N.M. Laird and D.B. Rubin, “Maximum-likelihood from incomplete data via the EM algorithm”, *Journal of Royal Statistical Society*, Vol. 39, pp. 1-38, 1977.

[9] P. Hong and M. Turk and T.S. Huang, “Gesture Modeling and Recognition Using Finite State Machines”, *Proc. of the fourth International Conference on Automatic Face and Gesture Recognition*, 2000.

[10] V.I. Pavlovic and R. Sharma and T.S. Huang, “Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review” *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 19, pp. 677-695, 1997

[11] Yoshua Bengio and Paolo Frasconi, “An Input Output HMM Architecture” *Advances in Neural Information Processing Systems* Vol. 7, pp 427–434, 1995

[12] D.E. Rumelhart, G.E. Hinton and R.J. Williams “Learning internal representations by back-propagating errors” *Nature* Vol. 323, pp 533–536, 1986

[13] J.A. Hartigan and M.A. Wong “A K-Means Clustering Algorithm” *Journal of Applied Statistics*, Vol. 28, pp 100–108, 1979