Robust Video-based Recognition of Dynamic Head Gestures in Various Domains - Comparing a Rule-based and a Stochastic Approach

Gregor McGlaun, Frank Althoff, Manfred Lang, and Gerhard Rigoll

Institute for Human-Machine Communication
Technical University of Munich (TUM)
Arcisstr. 21, 80290 Munich, Germany
{mcglaun, althoff, lang, rigoll}@ei.tum.de

Abstract. This work describes two video-based approaches for detecting and classifying dynamic head-gestures. We compare a simple, fast, and efficient rule-based algorithm with a powerful, robust, and flexible stochastic implementation. In both realizations, the head is localized via a combination of color- and shape-based segmentation. For a continuous feature extraction, the rule-based approach uses a template-matching of the nose bridge. In addition, the stochastic algorithm applies features derived from the optical flow, and classifies them by a set of discrete Hidden Markov Models. The rule-based implementation evaluates the key-feature in a finite state machine. We extensively tested the systems in two different application domains (VR desktop scenario vs. automotive environment). Six different gestures can be classified with an overall recognition rate of 93.7% (rule-based) and 97.3% (stochastic) in the VR (92.6% and 95.5% in the automotive environment, respectively). Both approaches work independently from the image background. Concerning the stochastic concept, further gesture types can easily be implemented.

1 Introduction

The development of user interfaces has become a significant factor in the software design process. Growing functional complexity and mostly restriction to purely tactile interaction devices required extensive learning periods and adaptation by the user to a high degree. To overcome these limitations, various interface types and interaction paradigms have been introduced. Multimodal interfaces currently resemble the latest step in this development. They enable the user to freely choose among multiple input devices, provide essential means to resolve recognition errors of individual components, and thus lead to systems that can be worked with easily, effectively, and above all intuitively[1]. Moreover, besides speech input, the use of gestures provides an interesting alternative for people with certain disabilities. This contribution illustrates the design and the evaluation of a system component for a video-based recognition of dynamic head gestures. The module is to be used as an integral part of an application invariant multimodal architecture.

From: GW 2003, Ed.: A. Camurri, LNCS 2915 Springer Verlag

1.1 Application Domains

Our overall research work focuses on the design of a generic platform for developing multimodal interfaces. Currently, the architectural concepts are being used in two different application domains. The first project deals with the development of a multimodal system for interacting with VRML browsers in a virtual-reality (VR) desktop environment[2]. Hence the user can arbitrarily combine conventional tactile devices with special VR hardware. As a key feature, (s)he can interact via paradigms on a semantic higher level, i.e. natural speech as well as dynamic hand or head gestures. The second project concentrates on the design of intuitive and error-robust components of a multimodal interface for controlling various infotainment and communication applications in an automotive field[3]. In both domains, the use of head gestures as an alternative or additional input possibility has proved to be very helpful with regard to increased system acceptance and the resolution of errors in the multimodal setup[4].

1.2 Related work

Many research groups have contributed significant work in the field of video-based head gesture recognition. In a system developed by Morimoto[5], movements in the facial plane are tracked by evaluating the temporal sequence of image rotations. These parameters are processed by a dynamic vector quantization scheme to form the abstract input symbols of a discrete HMM which can differentiate between four head gestures (*yes, no, maybe* and *hello*). Based on the IBM PupilCam technology, Davis[6] proposed a real-time approach for detecting user acknowledgments. Motion parameters are evaluated in a finite state machine which incorporates individual timing parameters. Using optical flow parameters as primary features, Tang[7] applies a neural network to classify ten different head gestures. The approach is quite robust with regard to different background conditions. Tang obtained an average recognition rate of 89.2% on an SGI workstation processing 30 frames per second.

1.3 System overview

Our system module for recognizing dynamic head gestures consists of four independent components: loading a single image (*image grabbing*), localizing the head candidates (*segmentation*), calculating movements of key points in the facial plane and in adjacent regions (*continuous feature extraction*), and finally, determining the type of the head gesture (*classification*).

The input image can be a frame of an MPEG stream, an isolated BMP image of a stored sequence, or directly streamed in by a dedicated hardware device (in our case a Video4Linux compatible grabbing card). Afterwards, the position and the size of potential head candidates in the given image are calculated by a color segmentation followed by a series of morphological filters. Additionally, we apply a template matching algorithm to find the nose bridge. These two steps are completed in the preprocessing which is identical for the two approaches.

On the basis of the segmentation result, the search space for the subsequent frames is restricted. Both the position differences of the nose bridge and diverse optical flow parameters are continuously stored in a feature vector. To determine which gesture occurred at which point time, elements of this vector are evaluated by a rule-based and a stochastic approach.

2 Preliminary Analysis

Before designing specific algorithms, we analyzed and categorized different types of natural dynamic head movements and determined the set of recognizable gestures. Hence we could benefit from the extensive video-material we collected in numerous usability experiments in both domains.

2.1 Gesture Vocabulary

In general, the movement of the head can completely be described by a six element vector denoting the three degrees of freedom with regard to translational and rotational movements, respectively. As an important result of a dedicated offline analysis of the video material, we found out that the majority of gestures (96.4%) has exclusively been composed of purely rotational movements. Thus in the approaches presented here, we exclusively consider head gestures that consist of one or a combination of head rotations.

With regard to the reference coordinate system shown in figure 1, six different elementary head gestures could be observed: moving the head *left* and *right* (rotation around the *yaw*-axis), *up* and *down* (rotations around the *pitch*-axis), and bending the head left and right (rotation around the *curl*-axis). Additionally, by combining basic movements, two compound gestures could be identified: head *nodding* and head *shaking*. A detailed analysis of the gesture material revealed that regarding the totality of rotations around the *yaw*-and the *curl*-axis, only 3.6% of the movements were twist gestures. Against the background of our application scenarios, this type of gestures can be neglected without a noticeable loss of system usability.

Concerning the VR desktop scenario, head gestures have mainly been used as alternative input possibility in case the hands of the user were busy with operating certain tactile devices. Moreover, head gestures have been applied

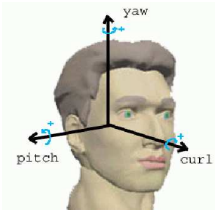


Fig. 1. Rotational axes for the different head movements

unconsciously to emphasize commands given by speech [3][2]. Analyzing the functional meaning of the observed head gestures, the purpose was to specify directions of movements in the virtual world.

Where in the VR application domain, the user can absolutely concentrate on the scene and the respective task, in the automotive environment, the primary task of the user is to drive the car (i.e. to perform a driving task). Operating the multimodal interface is the secondary task. The superposition of both tasks massively biases the workload of the user. In the automotive test scenarios, head gestures have mainly been applied to support yes/no-decisions in a dialog system, e.g. to accept or deny an incoming phone call. Moreover, in selected cases, head gestures were used to skip between individual audio tracks. In noisy fields, like the car, recognition of speech is often error-prone. Moreover, tactile interaction is usually coupled with a set of control glances at the display or special button devices[8]. Hence in special cases, head gestures offer a highly effective input alternative, as both hands can still be used to drive the car.

Since the recognition module is to be implemented in various contexts, we define two sets of possible head gestures. The first set (GS_1) contains all six gestures mentioned above. The vocabulary of the second set (GS_2) is designed to exclusively support user acknowledgment decisions. Thus, we reduced it to the gestures head *nodding* and head *shaking*.

2.2 Interaction time

To obtain quantitative results with regard to the interaction time of the individual gesture types and application scenarios, the video material has partly been segmented manually. For each gesture, 15 samples of 12 users (VR environment) and 15 samples of nine users (automotive environment) have been evaluated. Table 1 summarizes the average length of the gestures in frames and seconds, respectively. Interestingly, the compound gestures (*shake* and *nod*) had a significant smaller execution time in the automotive environment, which per judgment of the subjects, is due to an increased workload in the car (see section 2.1).

Table 1. Head-gesture interaction times in both domains

	VR desktop		Automotive	
gesture	frames	time[sec]	frames	time[sec]
LEFT	21.75	0.87	23.25	0.93
RIGHT	19.75	0.79	23.50	0.94
UP	15.50	0.62	22.75	0.91
DOWN	17.48	0.70	23.30	0.93
NOD	40.03	1.60	27.25	1.09
SHAKE	39.09	1.56	28.00	1.12
mean	25.58	1.02	24.67	0.99

3 Preprocessing

Before describing the different classification approaches, we briefly compare common techniques and sketch the preliminary steps of the image processing. The result of a successful segmentation process is a rectangle that characterizes the position and the size of potential head candidates in the input image.

3.1 Comparing head segmentation approaches

Based on the excellent overview given in[9], we experimented with various techniques. Since a fundamental requirement of our approach is real-time processing capability, various methods cannot be used due to enormous running time (e.g. Hough transformation and Eigenfaces). Localizing the eyes by checking for user blinks has proved to be insufficient when the head is moved intensely. The system could be initialized by an explicit twinkle without any kind of accompanying head motion, but this assumption would massively decrease the naturalness and usability of the system. To purely use background recognition, the image background would have to be separated from the moving foreground (the head). In this case, the background would have to be static and the foreground always dynamic, which does not hold for our application domains. Therefore, we propose a color-based segmentation approach, because it is rotation- and scale-invariant, and the calculation is very fast. Moreover, this method does not require any kind of initialization, and has proved to be highly robust with regard to arbitrary motion in the background.

3.2 Color-based segmentation

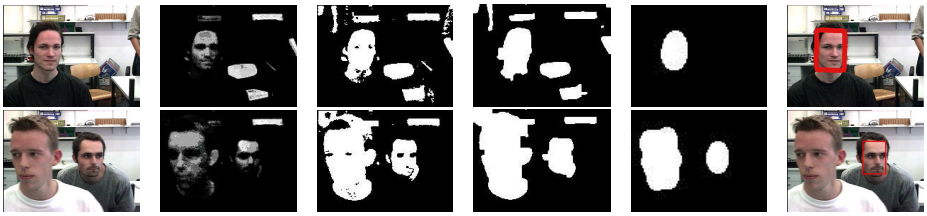
The individual steps of the segmentation process are visualized by the two sequences shown in figure 2. Given in the standard size of 382x288 pixels, the input image is in standard RGB color format, with each channel composed of 8 bit (figure 2(a)). To differentiate between skin color and background, the image is converted to the YCbCr color space. Since different skin types mostly differ in the luminance component and not with regard to the hue value, the Y-channel can be neglected in the following. Concerning the $CbCr$-plane, skin colors only cover a small fraction. For each of the color vectors, the probability of belonging to human skin can be estimated.

To simplify the color distribution, we use an approximation by the following Gaussian function. For specifying the individual parameters, the mean value $\boldsymbol{m}$ was calculated (where $\mathbb{E}$ denotes the expectation value):

$$\boldsymbol{m} = \mathbb{E}\{\boldsymbol{x_i}\} \text{ with } \boldsymbol{x_i} = \begin{pmatrix} Cr \\ Cb \end{pmatrix}$$

and the covariance matrix

$$C = \mathbb{E}\{(\boldsymbol{x_i} - \boldsymbol{m})(\boldsymbol{x_i} - \boldsymbol{m})^T\}$$



(a) (b) (c) (d) (e) (f)

Fig. 2. Individual steps of the segmentation process: the input image given in standard RGB format with a size of 382x288 pixels (a), skin-color information coded in a gray-value image (b), binarized image due to a certain threshold differentiating between potential areas of skin-color and background (c), result of a sequence of morphological filters to improve the segmentation result (d), final closing with a longish ellipse to identify potential head candidates (e), and marking head regions in the original input image (f).

on the basis of 42 random user skin samples $\boldsymbol{x_i}$. To filter out non skin-color areas, the histogram of the $CbCr$ part of the input image is multiplied with the Gaussian distribution calculated by:

$$\mathrm{p}(Cr, Cb) = \exp[-0.5(\boldsymbol{x_i} - \boldsymbol{m})^T C^{-1}(\boldsymbol{x_i} - \boldsymbol{m})].$$

The resulting histogram is used to project the color image onto a gray-value image (figure 2(b)), in which each skin color value is represented by a value according to the probability specified by $\mathrm{p}(Cr, Cb)$. Afterwards, this gray-value image is binarized differentiating between potential skin colors and background (figure 2(c)). Moreover, we apply a sequence of morphological filters on the binary image. First a *closing* with a small ellipse eliminates small particles that have occurred due to noise. Then an *opening* with a medium-sized rectangle tries to cover dark areas like in the eyes. For each blob, potentially occurring leaks will be filled. These leaks can often be found near to the eyes. As they are not skin-colored, they have a negative influence on the correct segmentation of the whole face region. The result of the filter process is shown in figure 2(d). Finally, a closing with a longish bigger ellipse removes all areas which do not have the correct size (figure 2(e)). By a bounding box R around the best-fitting ellipse, the position of the potential head candidate is specified (figure 2(f)).

3.3 Template matching

To further improve the quality of the segmentation result, we additionally apply a template matching algorithm. Therefore, a striking, invariable *region of interest* (ROI) in the facial plane has to be identified. A basic requirement for a robust tracking of this ROI should be the independence of special faces. Taking the center of the eyes as ROI results in misclassifications when the user blinks. In

this case, the eyes fuse with the rest of the face to one single blob. Moreover, the mouth drops out as a potential ROI, since it changes its form during talking. Therefore, we concentrate on the nose bridge as the key feature. For enlarging the matching criteria, we use a symmetric template including the nose bridge, the area of the eyes, and parts of the eye-brows.

For each of these head candidates, we calculate a measure of how good the template matches the current image region R. This is done by determining the match quality of the template and the input image column by column and row by row. The result of this match depends both on the quality of the template and the special kind of the matching algorithm. We use the standard gray-level correlation

$$c(x,y) = \sum_{(u,v)\in R} t(u,v)b(x+u, y+v),$$

where the template $t(x,y)$ is defined over $\mathbb{R}$ and $b(x,y)$ denotes the input image. We relate the individual gray-values to the medium gray-value and normalize them by their standard deviation. Using the gray-scale correlation instead of the sum of absolute gray-scale differences, changes in the light conditions can easily be handled. If the resulting match value is below a certain confidence measure (by default set to 0.7), the head candidate will not be accepted as a potential position of a head. If more than one candidate exceeds the threshold, the best correlation candidate is taken for further processing. This can be seen in the lower series of images in figure 2, where the right candidate is preferred.

The native segmentation phase is exited, if a head region is found. In subsequent phases, this head region is used for further calculation steps. In case the area gets to small or no blob is found anymore, the search is extended to the complete image. This principle guarantees an integral robust localization of the head and a fast tracking of the head regions in the image.

4 A Bottom-Up Rule-Based Classification

The rule-based approach (RBA) is a simple and performant implementation. Exclusively using one feature for classification, we could achieve considerable high recognition rates under the test conditions described in section 6.

4.1 Continuous feature extraction

Based on the template matching outlined in section 3.3, we can establish a local Cartesian coordinate system with its zero point in the lower left corner of the rectangle R (see figure 2(f)). Let j be an integer indexing each frame F of a video sequence. For the j-th frame F_j, let the center of the template be denoted by $\boldsymbol{c}_j$. Referring to the previous frame F_{j-1}, we can express the motion of the nose bridge by the difference vector $\boldsymbol{d}_j = \boldsymbol{c}_j - \boldsymbol{c}_{j-1}$. This vector is transformed into a polar representation using the absolute value $\|\boldsymbol{d}_j\|_2$ and the phase φ. Let

$d_{j,1}$ be the x-component and $d_{j,2}$ denote the y-component of $\boldsymbol{d}_j$, then the phase can be calculated via

$$\varphi_j = \begin{cases} 0 & , \text{if } d_{j,1} = 0 \\ \arctan(\frac{d_{j,2}}{d_{j,1}}) & \text{else} \end{cases}$$

Using φ_j and $\|\boldsymbol{d}_j\|_2$, we can specify the direction and the speed of the head motion for each frame. This forms the basis for the rule-based classification algorithm.

4.2 Classification

The head movements are modeled by means of a finite state machine (FSM) containing five motion states (*up*, *down*, *left*, *right*) and a non-motion state (*idle*). By means of the scheme depicted in figure 3, a motion direction represented by φ_j is assigned to a corresponding state of the recognizer. If $\theta = 0°$, the two-

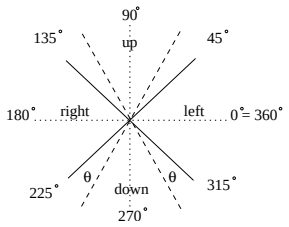


Fig. 3. The mapping scheme for the angle φ consists of four sectors (viewpoint: camera). Their size can be varied by the aperture angle θ.

dimensional space is symmetrically partitioned into four motion sectors (from the viewpoint of the camera). In test runs, we varied the partition types by applying different values for θ. A detailed description is presented in section 6.

Table 2. Classification scheme for different head gestures (HGs) according to the type and the number of changes in direction (CIDs)

HG \ CID	left→right	right→left	up→down	down→up
NOD	–	–	≥ 1	≥ 1
SHAKE	≥ 1	≥ 1	–	–
UP	–	–	1	–
DOWN	–	–	–	1
LEFT	1	–	–	–
RIGHT	–	1	–	–

With each gesture being represented by a certain series of motion and non-motion states, we can categorize the considered head gestures by their number of temporally sequent direction changes of the head motion. Table 2 shows the number and the type of direction changes for classification. Consequently, there is a set of valid and invalid state transitions for each gesture (see figure 4).

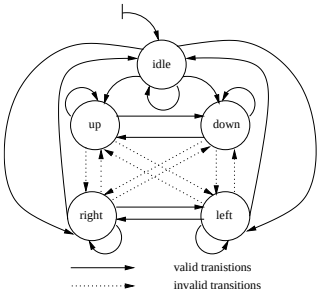


Fig. 4. Overview over the different state transitions in RBA

Initially, the FSM is in the *idle* state. Once a motion is detected, i.e. there is a change to a motion state, the recognition process is triggered. In a state history, each state is stored with respect to the current frame. If a certain number w of non-motion states is detected (the parameter w was adjusted in the evaluation period, see section 6), the system automatically starts to classify the state sequence. For this purpose, the state history is clustered: first we mark all state changes (i.e. transition states A to B with A $\neq$ B). The type and the number of identical states between two state changes are determined. If the number of identical states between two state changes is less or equal two, these states are erased from the state history. Given the state history presented in figure 5, the *up* states in frames 6 and 7, as well as the *down* state in frame 8 will be erased by this procedure. Thus we try to cope small outliers caused by inaccuracies of

state	-	left	left	left	left	left	up	up	down	left	left	left	left	left	left	left	left	left	...
frame	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18 ...

Fig. 5. Exemplary excerpt of the state history showing a set of *up* and *down* states within a motion to the *left*

the movements. After this step, the motion history is checked for invalid state transitions. If all transitions are valid, the statistics of the recognized direction types and transitions are evaluated applying table 2 to obtain the resulting gesture. On the other hand, if there are invalid state transitions (e.g. a set of *left* states followed by *up* states), the recognition result is mapped to *unknown*.

4.3 Spotting

In the current approach, only a prototypical spotting algorithm is implemented. Hence, we assume that gestures can be separated by a sequence w of *idle* states. w must be larger than the number of idle states that appear, when the head goes through the inflection points within a gesture. In our test sets, there has been a maximum of three idle frames at the inflection points (see section 6.2). If the gestures follow each other too fast, the RBA system is not able to distinguish between them. As soon as any kind of movement is detected, the recognition process is (re-)initialized. In current work, the FSM is being revised for a more efficient spotting.

5 A Top-Down Stochastic Classification

5.1 Continuous feature extraction

The tracking module calculates the spacio-temporal movements of head candidates in the image sequences and provides the basic data for the subsequent classification process. Concerning the stochastic approach (STA), we apply a hybrid combination of the Averaged Optical Flow (AOF) and the continuous template matching of the nose bridge. Hence, the template matching is used to estimate the position of the nose bridge in the tracked face region. The optical flow calculates motion vectors of certain areas of interest in subsequent images. The approach tries to find solutions to the known flow equation $\nabla I \cdot \boldsymbol{v} + I_t = 0$. Hence, $I = I(x, y, t)$ denotes the luminance, which depends of the local coordinates x, y, and the time t. Moreover, $\boldsymbol{v} = (\frac{dx}{dt}; \frac{dy}{dt})^T$ represents the vectored velocity of the head movement. For calculating $\boldsymbol{v}$, the standard Lucas-Kanade algorithm[10] is used. For reasons of system performance, we apply this local method instead of techniques operating on the whole image (e.g. the *Horn-Schunk* algorithm). In common implementations, the AOF is usually computed over a rectangle containing the whole head. Yet, we have found out that the bounding box around the head region in itself is sometimes not sufficient for adequate classification results. Concerning rotations of the head in the image plane (*yaw*- and *pitch*-axis), the resulting bounding box does not change significantly. This especially holds for segmentation results in which, for example, the bounding box comprises areas of the chest). As can be seen in figure 6, the bounding box enlarges to the area of the chest. A nod of this user could not be detected, because the total movement is completely enclosed in the primary rectangle. Thus, the result of the segmentation is only used to restrict the search area for the extraction of the features.

We apply the AOF technique to detect rotational movements around the nose bridge. Using the bounding box of the head, which results from the segmentation process, too much information from the background might be included in the calculation of the rectangles. This adulterates the results, if the bounding box covers a very large field and there is too much motion in the background. In

Fig. 6. The result of this skin-color segmentation process is a bounding box that is bigger than the primary head region and thus could, for example, not be used to classify *nod* gestures (left picture). This was a direct result from the first closing with an ellipse in the binary image (right picture).

our approach, we use the position of the nose bridge as an approximation for an element on the vertical symmetry axis of the face. Thus we are able to separate the face into a right and a left half. For each half of the face, we determine the AOF separately. Each region is marked with a square (see figure 6). By default, the square size is 40x40 pixels. If the square ranges out of the head region, it is scaled down. Thus the AOF is always calculated within the face region. As an effect of the implementation, we get two competing outputs for the AOF for each side of the face. Concerning the gesture sets GS_1 and GS_2, the AOF of the left and the right side of the face make for almost identical results. Thus the two redundant features are supposed to confirm each other.

There was also another motivation for computing the AOF within two separate face regions. Even though bend gestures (around the *curl*-axis) are not part of the evaluated test sets G_1 and G_2, this approach holds strong potential with regard to the recognition of those gestures. In this case, the speed vector of the nose bridge would not be sufficient for a classification. Concerning these *curl*-gestures, the vectors of the optical flow point into opposite directions up- and downwards, respectively, which enables a classification of such gestures.

The nose bridge is used as an origin of the local coordinate system of the rectangles bounding the face halves. In combination with the relative movement of the nose bridge, we are able to distinguish between horizontal movements of the user and head shaking itself. With horizontal movements, the AOF is zero, as the offset generated by the movement is compensated by the offset of the local coordinate system.

As a third feature, we use the difference vector of the nose bridge, just like in RBA (see section 4.1). These three features provide the basis for the classification process of STA to be described subsequently.

5.2 Classification

Modeling head gestures, a fundamental aspect is tolerance of small divergences regarding the temporal run and the duration. In the field of stochastic approaches, Hidden Markov Models very well cope with molding on time variant patterns. In addition, they show a robust behavior on small breaks during a

gesture, which are likely to appear when the head moves through the inflection point within a gesture. In the current implementation, we use Discrete Hidden Markov Models (DHMMs) composed of five states for the classification of head gestures. As mentioned in[11], DHMMs in general take more parameters, but the calculation is easier in the recognition process. The generation of the discrete symbols s_1, s_2, and s_3 that are fed into the DHMMs can be split up into two steps. First the optical flow and the arithmetic mean is computed over the regions which are in close vicinity to the nose bridge. Then both vectors as well as the speed vector of the nose bridge are discretized to integers between 0 and 5. Hence symbol 0 represents *no movement*. Symbols 1 to 4 are generated by applying the mapping scheme sketched in figure 3. These three feature symbols are canonically coded into a final symbol, using the known formula $s_1 + 5s_2 + 5^2s_3$. The classifier evaluates the symbol sequences and puts out a probability vector for each DHMM. Finally, the result is returned in terms of an n-best list.

5.3 Spotting

In the current state of development, the recognition process is automatically triggered, when any kind of head movement is detected. For this purpose, the absolute value of difference vector of the nose bridge (see section 4.1) is evaluated. If two head gestures directly follow one another, a number of five or more idle frames must be detected between these two gestures to separate them. Otherwise, the recognition process continues, which consequently leads to wrong results. The improvement of the segmentation between single gestures is part of current work. Hence we are about to implement a technique proposed by P. Morguet[12], which applies an improved normalized Viterbi algorithm for a continuous observation of the HMM output scores. This approach allows for integrated spotting and classification at a time.

6 Evaluation

The recognition module system has been implemented on an Intel Pentium IV machine with 512KByte cache and 1 GByte memory under the Linux operating system (Kernel 2.4.20). We have evaluated both the time performance and the recognition rates in the various domains. A single input frame is composed of an RGB image with 288x384 pixels.

6.1 Test environment and procedure

Both RBA and STA have been evaluated in two different application domains. One test series was run under optimized conditions in the computer-vision laboratory of the institute. We shielded the test environment from glares of the sun, and used a flicker-free light. The scene background was native consisting of different objects. During the data collection, the test subjects sat on a chair in

front of a camera (distance 60cm). They had to interact in different VR desktop scenarios, using head gestures of both test sets GS_1 and GS_2.

In the second test series, we focused to evaluate head gestures under preferably realistic conditions. The test domain was an automotive environment in a driving simulator. Driving the test car in the simulation, the trial participants had to perform head gesture interaction with different in-car infotainment devices. Yet, we did not consider any influences of artificial vibrancies or forces implicated by bumps, curves, or braking. The camera, which had the same sample rate as in the VR desktop environment, was positioned on the dash board over the steering wheel with an approximate distance of 45 cm. To simulate alternating light conditions, we shaded the laboratory, and used a set of spotlights.

In both test environments, the gestures have been evaluated in an offline analysis, using captured video sequences. In case subsequent gestures have been made, they were not manually segmented in order to analyze system behavior with respect to the prototypical spotting. For evaluating the recognition performance of the system three parameters have been chosen: the recognition rate (RR) measuring the percentage of correctly classified gestures, the false accept rate (FAR) denoting the percentage of misclassified gestures or movements that were misinterpreted as gestures, and the false reject rate (FRR) describing gestures that have erroneously not been accepted as valid.

6.2 Rule-based approach

We used a total of 153 video sequences of ten different test subjects, and 120 sequences of eight subjects in the automotive environment. During the implementation of RBA, we found out that in the regions of the inflection point of the head, a certain set of idle frames can appear. In the point of inflection, the motion of the head and thus the absolute value $\|\boldsymbol{d}\|_2$ is so small that the system does not detect any movement. In some cases, this was misinterpreted as a completion of the gesture, and two motion sequences which are actually associated were disjoined. On the other hand, RBA uses a sequence of idle frames for a temporal segmentation of the gestures (see section 4.3). With a too large value, subsequent gestures could not satisfactorily be separated (and consequently not be recognized), unless there was an accordingly larger break between the gestures. Thus, we tried to find a suitable threshold w at which the system assumes a gesture has been completed. As can be seen in table 3, a good performance is reached for $w = 5$. The reason is that the breaks at the inflection points within a

Table 3. Performance of RBA for different values of w concerning the test set GS_1 (47 sequences, VR desktop environment)

w	1	2	3	4	5	6	7
RR	33.5	52.8	80.6	90.5	96.3	86.8	75.8
FAR	57.6	32.1	11.8	0.3	0.2	5.7	14.8
FRR	8.9	15.1	7.6	9.2	3.5	7.6	9.4

gesture varied between one and three idle frames. Between subsequent gestures, there has been an averaged number of 4.7 idle frames. The recognition rate gets worse for $w \geq 6$, as now in most cases, the subsequent gesture can no longer be separated. Results in the automotive domain did not significantly differ from this realization.

In some scenarios, the gestures have not been made very exactly. Evaluating motion histories of horizontal head movements, it was remarkable that for a short instant (1-2 frames), some test subjects made a motion which the system classified as a vertical movement. In 18% of all test sets, we have observed this motion sequence short before or after the head ran through its inflection point. On the other hand, an according phenomenon has only occurred in 2% of the evaluated vertical head movements. As mentioned in section 4.2, small outliers can be erased from the state history. To further improve system behavior in this regard, we enlarged the aperture of the horizontal mapping sectors by varying the angle θ (see figure 3 in section 4.2) and studied the effect on the recognition performance. Regarding gesture set GS_1, we have got the recognition rates de-

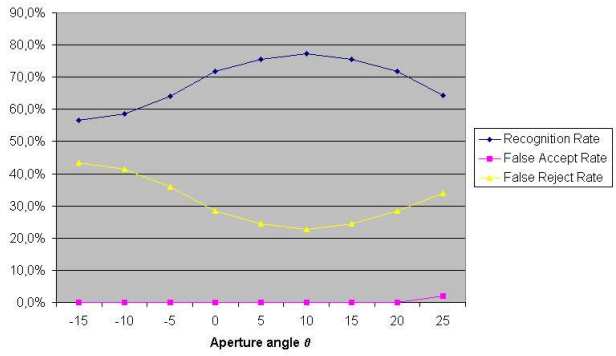


Fig. 7. Performance of RBA for different values of θ (test set GS_1, 27 sequences, VR desktop environment)

picted in figure 7. This chart shows an explicit maximum of RR with a coeval minimization of FRR for $\theta = 10°$. With other values of θ, either the horizontal or the vertical sectors get too small. The evaluation of θ in the automotive domain confirmed these results.

Using the optimized parameter settings mentioned above, the recognition performance for both gesture sets (GS_1 and GS_2) dependent on the different domains are presented in table 4. Concerning GS_1 and calculated over both domains, in 38,4% of all error cases, there was a confusion of *shake* with *left* or *right* or vice versa (29,7% confusion between *nod* with *up* or *down*). Thus expectedly, the system showed a better RR for the reduced set GS_2, where this kind of misinterpretation was a priori impossible.

Table 4. Characteristic rates of RBA in both domains

	VR desktop		Automotive	
Gesture set	GS_1	GS_2	GS_1	GS_2
RR	92.0	95.2	91.2	94.0
FAR	6.5	1.1	2.3	4.7
FRR	1.5	3.7	6.5	1.3

Considering all types of gestures, the RR is better in the VR desktop scenarios than in the automotive environment, where head movements often were less distinct. This particularly happened in cases the subjects did not have a frontal view into the camera. In 20 evaluated test sequences, the head of the test participant was initially rotated by approximately 40°. Hence, RBA had major problems in correctly classifying the gestures. Although the template matching worked well, in a large number of frames, the movement vector $\boldsymbol{d}$ of the nose bridge was close to zero, which was a straight consequence of the distortion of the head movement. Thus a large set of idle states was detected, and consequently, the gesture was erroneously rated complete.

6.3 Stochastic approach

The training corpus for the DHMMs has consisted of 32 selected symbol sequences. It contained gestures of four persons of different skin colors and one person wearing glasses. We have used the Baum-Welch method for the training. The data for test and training has been strictly disjoint.

STA has intensely been evaluated in both domains. To get a usable basis of comparison, we have fed exactly the same sequences into the STA system. In tables 5 to 6, the recognition results can be seen. Similar to RBA, in both domains, there has been a strong affinity between direction related gestures (*up*, *down*, *nod*, and *left*, *right*, *shake*, respectively). This effect has been aggravated, when the gestures are made very quickly. Then the resulting symbol sequence corresponding to the gesture has contained too few elements, so no good match for an DHMM has been found. Particularly, the very good recognition rates for the reduced set G_2 have been due to the training corpus containing a great variety of gestures of different durations.

In some cases, blinking or even moving the pupils have had a negative impact on the computation of the AOF, which has consequently lead to misclassifications. Moreover, we have observed that the AOF is more likely to be error-prone to bad light conditions than the template matching algorithm and the feature extracted from it.

With regard to inexact head movements, which were mentioned in section 6.2, STA has shown a very robust behavior. Concerning the test set in which the head of the subject was initially rotated by 40°, in 73.2% of all cases the gesture has been classified correctly.

Table 5. Recognition rates of the STA approach with regard to gesture set GS_1 in the VR desktop environment (left table) and in the automotive environment (right table). In the first column, G stands for the actual head gesture, and in the first row, H denotes the HMM modeling this head gesture.

G \ H	Up	Down	Left	Right	Shake	Nod
Up	**96.3**	0.4	0.1	0.1	0.6	2.5
Down	0.5	**96.1**	0.2	0.2	0.2	2.8
Left	0.2	0.1	**98.0**	0.8	0.8	0.1
Right	0.1	0.1	0.5	**96.6**	1.9	0.8
Shake	0.1	0.3	1.4	1.0	**97.0**	0.2
Nod	1.3	2.5	0.1	0.3	0.2	**95.6**

G \ H	Up	Down	Left	Right	Shake	Nod
Up	**95.2**	1.2	0.3	0.3	0.2	2.8
Down	1.9	**94.5**	0.4	0.3	0.1	2.8
Left	0.2	0.6	**95.0**	2.2	1.1	0.9
Right	0.5	0.7	1.2	**94.2**	3.1	0.3
Shake	0.3	0.3	2.2	2.5	**93.9**	0.8
Nod	1.5	3.0	0.5	0.2	0.6	**94.2**

Table 6. Recognition rates of the STA approach with regard to gesture set GS_2 in the VR desktop environment (left table) and in the automotive environment (right table). In the first column, G stands for the actual head gesture, and in the first row, H denotes the HMM modeling this head gesture.

G \ H	Shake	Nod
Shake	**98.2**	2.2
Nod	1.8	**97.8**

G \ H	Shake	Nod
Shake	**96.8**	3.9
Nod	3.2	**96.1**

6.4 Benchmarking

The test set for evaluating the time performance of the individual system modules is composed of 54 video sequences of eight different users, each containing a single head gesture. The sequences differ in the number of head candidates, the size of the head region and the background environment. We have collected the following time data: *basic image operations* (IO), like scaling and converting, *head localization* (HL), *optical flow* (OF), *template matching* (TM), and *loading of an image* (IL). Table 7 (left) summarizes the results.

All values are given in microseconds (usec). Thereby, the *total time* (TT) is the average time that is needed for the complete calculation cycle of a single frame. In general, it is not necessarily the sum of the individual times, since individual modules (e.g. template matching) potentially run multiple times per cycle. The values for both domains do not differ significantly.

On average, we obtained a total running time of about 18msec, which corresponds to 55 images per second. The time for displaying the image makes up additional 10msec based on the OpenCV system functions. The time for final classification (concerning both approaches) is below 1msec and thus not explicitly mentioned. Thus both implementations definitely meet real-time conditions on standard state-of-the-art hardware.

A more detailed evaluation of the HL is given in table 7 (right), specifying the morphological operations for the *first closing with the ellipse* (MO_1), the *elimination of small particles* (MO_2), the *padding of the eye region* (MO_3), estimating the *skin color* (SC), calculating the *head blob* (HB) and, finally, the morphological operation for *closing leaks in the resulting blobs* (MO_4). Again,

Table 7. Results of the benchmarking for components of a cycle (left) and for the head localization (right). All time data is represented in usec.

	IO	HL	OF	TM	IL	TT
VR	2912	956	998	2321	6591	17627
CAR	3010	1085	1004	3325	6954	19475

	MO_1	MO_2	MO_3	SC	HB	MO_4
VR	320	105	90	96	105	198
CAR	385	139	103	85	134	210

these values are specified in microseconds. In both domains, the quota of the first closing (MO_1) requires about 34 % of the whole time for the head localization.

6.5 Comparative Discussion

Each of the discussed implementations offers individual advantages due to different points of view. RBA is a fast implementation which, considering its simplicity, has very good recognition results in the reduced gesture set GS_2. Thus in both domains, it is predestinated for low-cost implementations or the use in dialogs with decision questions. As it hardly takes processing power, an application in a low CPU resource environment, like the automobile, appears to be very interesting. The individual rules of RBA are hard-coded in an implicit knowledge base, thus it is strictly limited to the given gesture vocabulary. At any time, STA can be trained offline with arbitrary additional sets of gesture models of different domains. An extension of RBA to other gesture types is rather expensive, as the FSM has to be completely revised and updated. In this context, the tracking of only a single feature might be problematic in some applications. For example, gestures that are generated by rotations around the *curl*-axis (which we excluded from our test set) can hardly be recognized by RBA, since during *curl*-rotations, the position of the nose bridge hardly changes. Hence the architecture of STA has great potential for an easy extension of the vocabulary, as we separately evaluate the optical flow in the left and right part of the face (see section 5.1).

The segmentation algorithm was absolutely stable concerning moving objects in the background or scenarios with more than one potential head candidate. In the VR field, the template matching algorithm worked extremely robust. In 98.9% of all test sets, the nose bridge has been found correctly. Both the RBA and the STA system were mainly developed in the VR environment. Using the same template in the car, the rates for the matching have been slightly worse, as the template matching algorithm is not scale-invariant with respect to larger distance changes in z-direction (depth dimension). Moreover, the difficult light conditions in the automotive environment contributed to an aggravation of the RR. In 97.4% of all automotive test sets, the template matching algorithm has been localized.

Regarding person independence, both approaches are extremely robust against different skin color types, persons wearing glasses, ear- or nose-jewelry. Yet, both systems have major difficulties categorizing head gestures of persons with strong hair-growth covering the nose-bridge or subjects having a full beard. In this case, either the head localization or the template matching algorithm, which forms the

basis for the feature extraction, does no longer work properly. Moreover, the optical flow delivers too many diametrically opposed or divergent direction vectors which leads to unemployable results, especially in the automotive environment, where difficult light conditions negatively contribute to this effect. Hence STA detected direction changes which were actually not made. This lead to the fact that gestures, like *left* or *right* were wrongly interpreted as *shake*.

If test subjects do not sit directly in front of the camera, but have their head slightly turned, STA is again more stable. Hence the optical flow of one half of the face still delivers a movement direction. From the viewpoint of the camera, the motion of the nose bridge is no longer recognizable. Thus the difference vector $\boldsymbol{d}$ (key feature of RBA) is close to zero, and consequently, the system detects idle frames. RBA is more error-prone to inexplicit and indistinct gestures than STA. In the current implementation, RBA has only a set of rigid rules by which the system is able to overcome smaller inaccuracies in the head movements (see section 4.2). Individual nuances of motion phases can hardly be covered unless massively blowing up the code book of the rules. The latter, in fact, would noticeably deteriorate system performance. On the other hand, STA is highly adaptable to both specific conditions and individual users. It has better recognition results than RBA, if gestures are made very quickly or clipped. In this regard, the implementation again benefits from the flexibility of DHMMs to time variant patterns and the broad corpus by which it has been trained.

7 Ongoing and Future Work

The systems presented here are in an intermediate state of development and thus are subject to additional changes. As mentioned above, especially the HMM approach can be enhanced with regard to recognizing further sets of head gestures. Hence we concentrate on the implementation of gestures which are generated by rotations around the *curl*-axis.

As the head gesture recognition unit is to be used as part of a multimodal system, two ways of processing the recognizer output are researched. In a current approach, the head gesture recognition unit is to be coupled with a natural speech recognizer[13], using an early feature fusion. This allows for further improvement of the overall recognition rates and benefits from the fact that many user inputs (especially confirmation and negation) are temporally overlapping[14].

In a late semantic fusion approach based on a client-server architecture[3], the outputs of the recognizers are combined in a central integration unit. Applying context knowledge, the integrator can dynamically vary the vocabulary of the head gesture recognizer via a TCP/IP socket-based communication. E.g., if a yes-no answer is expected in a system dialogue the system could instruct the recognizer to load configuration GS_2, as other input does not make sense in this system context. By this, we expect a remarkable improvement of the recognition rate and time performance.

8 Conclusions

Head gestures offer strong potential for an intuitive, efficient, and robust human-machine communication. They are an easy and helpful input unit, especially in environments, where tactile interaction is difficult or error-prone (like in the automobile). We discussed two differently motivated approaches. The strongly limited rule-based implementation allows for satisfactory categorization of head gestures. It is predestinated for the evaluation of yes-no dialogs in environments, where CPU-resources are low. On the other hand, the HMM-oriented stochastic approach has excellent means to robustly recognize even inaccurate head movements, and can easily be enhanced for recognizing further types of gestures.

References

1. Oviatt, S.L.: Multimodal interface research: A science without borders. Proc. of the 6th Int. Conf. on Spoken Language Processing (ICSLP) (2000)
2. Althoff, F., et al.: Using multimodal interaction to navigate in arbitrary virtual worlds. In WS on Perceptive User Interfaces (PUI 01) (2001)
3. McGlaun, G., et al.: A new approach for the integration of multimodal input based on late semantic fusion. In Proc. of USEWARE 2002 (2002)
4. Althoff, F., et al.: Experimental evaluation of user errors at the skill-based level in an automotive environment. Proc. of CHI 02 (2002)
5. Morimoto, C., et al.: Recognition of head gestures using hidden markov models. Proc. of IEEE Int. Conf. on Pattern Recognition (1996)
6. Davis, J., et al.: A perceptual user interface for recognizing head gesture acknowledgements. In WS on Perceptive User Interfaces (PUI 01) (2001)
7. Tang, J., et al.: A head gesture recognition algorithm. In Proc. of the 3rd Int. Conf. on Multimodal Interface (2000)
8. McGlaun, G., et al.: A generic operation concept for an ergonomic speech mmi under fixed constraints in the automotive environment. In Proc. of HCI 2001 (2001)
9. Yang, M., et al.: Detecting faces in images: A survey. In IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) **24** (2002) 35–58
10. Barron, et al.: Performance of optical flow techniques. IJCV **12:1** (1994) 43–77
11. Rabiner, L.: A tutorial on hidden markov models and selected applications in speech recognition. In: Proceedings of the IEEE. Volume 77:11. (1989) 257–285
12. Morguet, P., et al.: Spotting dynamic hand gestures in video image sequences using hidden markov models. Proc. of ICIP 99 (1998) 193–197
13. Schuller, B., et al.: Navigating in virtual worlds via natural speech. In: 9.th Int. Conf. on HCI. (2001)
14. Althoff, F., et al.: Combining multiple input modalities for VR navigation - A user study. In: 9.th Int. Conf. on HCI. (2001)