Robust Real-Time Face Tracking and Gesture Recognition

J. Heinzmann and **A. Zelinsky**
Department of Systems Engineering
Research School of Information Sciences and Engineering
Australian National University
Canberra, ACT 0200, Australia

Abstract

People naturally express themselves through facial gestures. We have implemented an interface that tracks a person's facial features robustly in real time (30Hz) and does not require artificial artifacts such as special illumination or facial makeup. Even if features become occluded the system is capable of recovering tracking in a couple of frames after the features reappear in the image. Based on this fault tolerant face tracker we have implemented real-time gesture recognition capable of distinguish 12 different gestures ranging from "yes", "no" and "may be" to winks, blinks and "asleep".

1 Introduction

To advance the field of robotics greater efforts are needed to improve the communication interface between humans and robots. An important area of this work is the recognition of gestures of the head, body and expressions of the face. Such body language is regarded as one of the most natural forms of human expression. If we are to create intelligent robots in the future, human body language must be understood. Robots that can communicate in this manner will be particularly useful for disabled people and to users who are inexperienced in working with traditional computer interfaces. A gesture interface could open the door to many applications ranging from control of machines to "helping hands" for the elderly and disabled. The crucial aspects of such systems are their robustness and their real-time capability. At present many computer vision systems are still lacking in computational speed and can't cope with tracking errors, temporary occlusion of features. We have overcome these problems by using dedicated computer vision hardware that is capable of real-time feature tracking and by developing specialised algorithms that make face tracking robust.

The principle of our vision system is image correlation (or template matching). Template matching is not directly applicable to face tracking especially under the constraints of non-homogeneous illumination and contrast enhancing makeup can't be used. The appearance (i.e shape and shade) of facial features changes when an observed person's head moves. A large difference occurs between the reference template and the features being tracked. This effect can be further compounded when people tilt their heads to the side and cause a rotation of the feature. So tracking of the facial feature fails and there is no way to recover from this error.

Implementations of gesture interfaces based on image correlation have been previously reported [Azarbayejani *et al.*, 1993; Hager and Toyama, 1996; 1995]. The work of [Azarbayejani *et al.*, 1993] reported on the manipulation of a virtual reality clone. This system automatically acquired the tracking templates at runtime using heuristic filters. With this approach it is not possible to determine the initial state of features e.g. eyes are open and the position of the face can only be measured relative to the initial image. The [Hager and Toyama, 1996; 1995] implementation uses deformable template matching performed in software to track a face. This system does not exploit the geometry of the facial features to assist in tracking or error recovery so only small rotations of the head are allowed. Related to the template based approach is the method proposed by [Maurer and von der Malsburg, 1996]. Instead of using bitmap templates for tracking this system calculates the local response to a Garbor filter of different frequencies and orientations. The video frame rate required for this method is reported to be >10Hz. Although, even using reduced images of 128x128 pixels the calculation time for one frame is 4sec. Like the Hager *et al.* system it does not exploit the geometric relationship between features. Both systems are not able to recover from tracking errors caused by temporary occlusion of features.

An alternative approach is to use specialised filters instead of templates to localise facial features. The filters are implemented as small programs that classify pixels, e.g. belonging to the iris or to the eyeball or to the eyelid of an eye. The classifier usually uses heuristics or probabilistic techniques. A system controlling a clone in virtual reality using this approach is reported by [Saulnier *et al.*, 1995]. To improve the reliability of pixel classification contrast enhancing makeup has to be used and the system only runs at 10Hz. Since the system does not exploit facial geometries the face tracking can fail when features are temporarily occluded. A system running theoretically at 100Hz is reported by [Gee and Cipolla, 1994]. This impressive performance is achieved because the system uses simple filters that are not able to validate the tracking results. Thus, the tracker can neither detect tracking errors nor recover from them. Like the other systems this tracker is not able to pick up a face that recently entered the image. Another face tracker reported by [Jacquin and Eleftheriadis, 1995] uses a oval shape model to detect the location of the face in the image. Using this general model a high robustness and person independence can be achieved, though, the required computation is high and makes the approach unsuitable to real-time applications.

Reliable and rapid tracking of the face gives rise to ability to recognise facial gestures. This should not be confused with work being done to understand facial expressions [Darrel and Pentland, 1995; Black and Yacoob, 1995; Saulnier *et al.*, 1995]. The work described in [Darrel and Pentland, 1995] is concerned with the classification of the expression into states such as happiness, anger, fear, sadness or disgust. The system uses nearest neighbour matching to a set of templates to classify the expression state. Templates can be acquired automatically or synthesised by the system. The detection of facial expression runs at about 8Hz. The aim of the system reported in [Black and Yacoob, 1995] is similar, but the classification is based on the affine transformation parameters of individual templates for different facial features. However, the system requires high resolution images (560x420 pixels) so real-time performance could not be achieved.

Due to the real-time constraints of the problem of gesture recognition based on head motion there have been few projects undertaken in this area. [Darrel and Pentland, 1995] describe a system that is able to recognise hand and facial gestures based on the template matching system developed by [Azarbayejani *et al.*, 1993]. The tracking output consists of different motion parameters of the face. Sequences of motion vector parameters can be recorded where the start and the end must be manually determined. Such a sequence forms a "space-time" gesture. During recognition the last motion vectors are compared with the stored gestures using a dynamic time warping algorithm similar to that used in speech recognition applications. After receiving each new motion vector the history of motion vectors have to be matched with all gestures stored in the library. This has a heavy impact on the calculation time. Using only two gestures of length 20 state vectors the system can only run at 10Hz. The system we describe in this paper has proved to run at 30Hz and can distinguish 12 different gestures. This is possible due to a specialised gesture model which requires little computation yet can still detect gestures robustly.

2 Real-time Vision System

The MEP tracking vision system manufactured by FUJITSU is designed for real time tracking of multiple objects in the frames of a NTSC video stream. It consists of two VME-bus cards, a video module with frame grabber and overlay memory for displaying graphics and textual information, and a tracking module which can track over 100 templates at video frame rate (30Hz for NTSC). A MC68040/25MHz processor card running VxWorks executes the application program and controls the vision system.

Objects are tracked using template correlation. There are two sizes for templates available, 8x8 or 16x16 pixels, but both sizes can be magnified by a factor between 1 and 4 in the X- and Y-direction independently. The frames of the video stream are digitised by the video module and stored in dedicated video RAM which can also be accessed by the tracking module. Next, the templates are matched within specified search windows of the image. The comparison itself of the template and an area of the digitised frame is done by a cross correlation using the mean absolute error method. The distortion, which is the sum of the absolute greyscale differences of corresponding pixels in the template and the video frame, indicates how similar the two images are. Low distortions indicate a good match and high distortions appear when the images are very different. The formula for the distortion D is shown in Formula 1 where $Size$ is the template size (8 or 16), $g_t(x, y)$ is the grey value of the template at the specified coordinates, $g_f(x, y)$ is the grey value of the pixel in the last frame, m_x and m_y are the magnifications of the template in X- and Y-direction and o_x and o_y are the offsets in the frame.

$$D = \sum_{x=0}^{Size} \sum_{y=0}^{Size} \mid g_t(x_t, y_t) - g_f(x_f, y_f) \mid \qquad (1)$$

$$\text{where} \quad \begin{array}{ll} x_t = x \cdot m_x & y_t = y \cdot m_y \\ x_f = x \cdot m_x + o_x & y_f = y \cdot m_y + o_y \end{array}$$

To track the template of an object it is necessary to calculate the distortion not only at one point in the image but at a number of points within the search area. To track the motion of an object the tracking module finds the position in the search window where the template matches with the lowest distortion. By moving the search window along according to the tracking results objects can be tracked easily This method works perfectly for objects that do not change their appearance or shade and that get occluded by other objects.

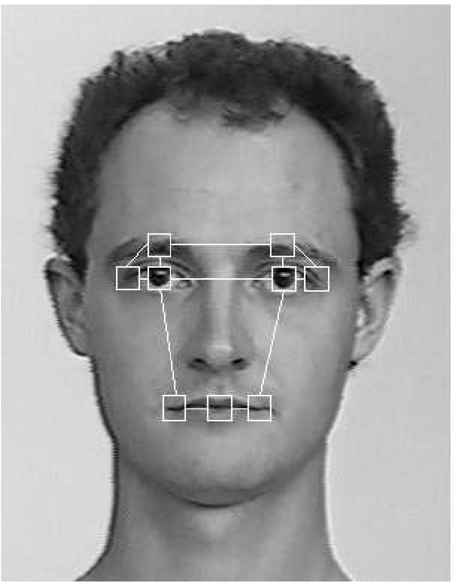

Figure 1: The tracked features and their references

3 Improving the System Robustness

Our basic idea to overcome the problem of temporarily distorted or occluded features is to have individual search windows help each other to track their features. Since the geometric relationship between the features in a face is known a lost search window can be relocated by assistance from those that are still tracking. In a first approach a two dimensional model of the face is used where the features are placed on a virtual grid. During face tracking the search windows can monitor their relative position to the other windows and readjust their coordinates if necessary. Figure 1 shows the face of a person where the boxes mark the 9 features that are used for tracking. The lines indicate the connections between the search windows.

The robustness of the tracker heavily depends on the method used to fuse the tracking results of the vision system with the geometric data derived from the 2D face model. In difficult situations, when the head is turned and all the templates match with high distortions, it is not possible to determine whether or not a feature has been lost based on the distortion reported by the vision system. Thus, a probabilistic approach must be chosen to merge the uncertain tracking data with the information derived from the 2D facial model. We believe that Kalman filters are an appropriate method to cope with this problem.

3.1 Multiple Interdependent Kalman Filters

The Kalman filter is a recursive linear estimator which merges the measurements of sensors observing the environment with a system state prediction that is derived from a system model [Bozic, 1979; Durrant-Whyte, 1995]. Kalman filters are used in many applications such as navigation of planes, missiles and mobile robots where uncertain measurements from sensors that observe landmarks are used to localise a vehicle. By merging sensor information with the internal model information, the Kalman filter guarantees an optimal estimate of the system's state vector.

The use of Kalman filters in feature tracking has been previously reported by McLaughlan *et al.* [McLauchlan *et al.*, 1994] to estimate 3D structure from motion. However, in our approach features are used to assist each other in tracking. Hager *et al.* [Hager and Toyama, 1996] proposed a similar idea using geometric constraints and feature states for tracking. However, a binary switching method between winning and loosing features is used to decide the features that guide the other tracking features. Our approach uses all the features for tracking and is weighted to the features that are tracking best. In difficult situations, when the head is turned around and all templates match with high distortions our merging method shows much more robust behaviour than binary switching. So larger rotations of the head can be tracked than with previously reported methods.

An individual extended Kalman filter is applied to each feature. The tracking results provided by the vision system are used as sensor input where the distortion is converted to a variance of this measurement. The results of the different templates are fed into a *Selection*-unit which selects the template with the lowest distortion. Only the coordinates and the associated variance of the selected template are forwarded to the Kalman filter. Instead of a system model describing the behaviour of the feature itself an estimation of the features position based on the position of the reference features is fed into the Kalman filter. Equation 2 shows how a *Merge*-unit derives a position estimation $\overline{pp}_i$ from the positions $\overline{fp}_j$ of the reference features with the according variance $P_j(t-1)$, the displacement vectors $\overline{d}_{ij}$ between the reference feature and the feature itself and the movement vector $\overline{m}$.

$$\overline{pp}_i(t) = \frac{\sum_j \frac{\overline{fp}_j(t-1)+\overline{d}_{ij}}{P_j(t-1)}}{\sum_j \frac{1}{P_j(t-1)}} + \overline{m}(t) \qquad (2)$$

The variance PP_i associated with this position estimation is calculated from the variance of the last position and the variances of the reference positions according to Formula 3.

$$PP_i(t) = PP_i(t-1) + \frac{1}{\sum_j \frac{1}{P_j(t-1)}} \quad (3)$$

Next, according to the general Kalman filter equations the position of the feature is calculated. These computations must be performed for each feature in each frame cycle. In the next cycle the position calculation is used by other features to estimate their own position.

The system described thus far will track facial features however strictly limited turning and tilting of the head is only possible. The rigid connections in the 2D facial model can force well matching templates to leave their associated features to satisfy the geometric requirements of the 2D model. We overcome this problem by introducing a projection to deal with the deformations of the 2D model. The rotational angle ϕ_f and the compressions factors c_x and c_y in X- and Y-direction of the model are calculated each frame. If the calculated values are used as raw input the system becomes unstable. However, a simple low pass filter sorts these problems out.

3.2 Further Enhancing Robustness

At this stage the system is capable of robustly tracking the face in the rotational range of about 60 degrees. Further rotations causes features to disappear and others to be highly distorted. Should all the features disappear or deform in a way that the system can't track them anymore, the tracking fails completely and is not able to recover even if the features reappear in the image. A similar situation arises when the system is started up and the person may not be in the image yet or the face is located differently from the coordinates in the initialisation file. The search windows can't lock onto the features and thus guide others to their correct locations. These situations are common in real world applications and manual initialisation of the feature coordinates in the first frame of a recorded video sequence can't deal with this problem. A mechanism is required to perform online recovery in real-time. Another desirable characteristic in this context is that the system does not need to switch between an initialisation mode trying to localise any feature and a tracking mode because this implies that the system can determine whether all features are lost or not.

The single mode solution proposed in this paper increases the system resources used to relocate a feature as the variance of its position increases. Additional search windows, called *Area search windows*, that use the same template as the feature tracking window are added in the surrounding area of the estimated feature position. The size of the area and the number of search windows used are functions of the feature's variance. Since system resources are limited, for each template we must determine the number of area search windows to allocate. Our system distributes the available search windows according to the requirements and the uniqueness of the features. Characteristic features like the eyes and eyebrows go first, while the features with poor contrast around the mouth go last. The last problem to be solved is when to believe a feature was actually lost and found by one of its area search windows or when should the feature be relocated to the position of a well matching area search window. At the moment we are using a malus factor[1] applied to the distortion of the area search window and compare the resulting distortion to the distortion of the feature tracking window. If the area distortion after application of the malus factor is smaller than the centre distortion, the feature is relocated by selecting the area search windows coordinates as one input of the Kalman filter.

Another problem to be addressed is the similarity of the two eyes and eyebrows. During the recovery phase the window of an eye can easily get stuck on the wrong eye, and the accompanying eyebrow will also match very well on the wrong eyebrow. The system remains in this state without any chance of to recover unless the eye, that is tracked by the wrong window, disappears for some reason. For this reason a third eye is introduced into the system. The search window for the 3rd eye is set to the coordinates calculated by adding the vector connecting the left eye and the right eye to the coordinates of the right eye, this is where the right eye would be if the right eye window tracks the left eye. The variance is set to the variance of the left eye and the appropriate number of area search windows is added. This variance is usually large if the left eye is being searched in the area of the ear and low if both eyes are being tracked the right way. So only when one eye appears to be lost then the area search windows are used to recover it.

The decision whether or not an error has occurred is done in the same way as for area search windows. The best distortion of the left eye is compared with the best distortion of the 3rd eye search window multiplied with a malus factor[2]. If the 3rd eye search window detects an eye, then the coordinates of the right eye are set to those of the 3rd eye and those of the left eye are set to those of the right eye.

The search position of the 3rd eye window is alternated each frame, so both sides (left and right) of the facial feature network are checked at a frequency of 15Hz.

[1] The malus factor for area search windows is 1.8.

[2] For the 3rd eye the malus factor is 3.

a) b) c) d)

Figure 2: Various deformations of the feature grid

4 Gesture Recognition

Robust real-time face tracking gives rise to the possibility of recognising gestures based on motions of the head. Gesture recognition and face tracking are implemented as independent processes. Both processes run at the NTSC video frame rate (30Hz).

The design of the algorithms for gesture recognition is determined by hard real-time constraints since dozens of gestures must be compared with the data stream from the face tracking module. Also, gesture recognition must be flexible in respect to the performance of the gesture including timing and amplitudes of the parameter. The system should assign confidence values to the gestures that it recognises. This allows higher level processes to use the gestures as input and interpret them optimally. To avoid the computationally expensive time warping method we developed a recursive method taking only the current motion vector of the face into account. A set of finite state machines implicitly store the previous tracking information.

Gesture recognition is implemented as a two layer system based on the decomposition of gestures into atomic actions. The lower layer recognises basic motion primitives and state primitives called atomic actions. The output of an atomic action consists of its activation which is a measure for the similarity of the recent motion vectors and the atomic action definition. The system incorporates 22 predefined atomic actions for which the activation is calculated in each frame cycle.

The upper layer is concerned with the recognition of patterns in the activation of the set of atomic actions. A gesture is defined by a sequence of atomic actions and time constraints for the occurrence of each of them. Each time the first atomic action of a gesture definition is activated an instance of a finite state machine is generated dynamically. This instance then observes the activation of the next atomic action in the gesture definition and does the transition to the next state if the activation occurs within a given time frame. If no activation occurs in the time frame the finite state machine instance is deleted from the system. When the state machine reaches it's final state the atomic action sequence is completely recognised and the gesture is send to output together with a confidence measure of how well the observed pattern matched the gesture definition. Figure 3 shows a selection of gestures recognised by the system. For a full description of gesture recognition refer to /citeHei96.

Nod for Yes Shake for No May be

Look Down Look Left Turn to Left

Figure 3: Example Gestures

5 Experimental Results

The complete system was extensively tested in our laboratory. Sequences of several minutes of persons performing gestures were video taped and used for analysis. The face tracker proved to be reliable under stable illumination and tolerated rotations of the face not exceeding 60°. Figure 4 shows the output of the face tracker and the recognised gestures. The recognition rate for the gestures differs significantly. Complex but easy to track gestures like *yes* and *no* have the best recognition rate (95%). Difficult to track gestures like the *turn*-gestures are sometimes not identified due to tracking errors but still have recognition rate over 70%. Short gestures such as winking and blinking are recognised whenever they are performed, though due to the shortness of the gesture the number false detections can be as high as 20%.

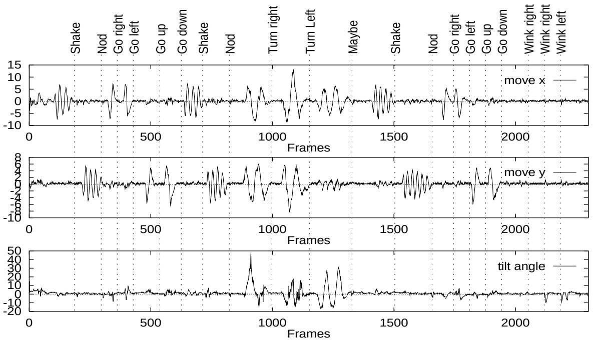

Figure 4: Activation pattern of the atomic actions eyes.open, half.open, closed during a blink

6 Conclusion and Future Work

Our system is able to track the features of the face at 30Hz without special illumination or contrast enhancing makeup. It is able to automatically initialise tracking without any restrictions to the position of the face in the first frame. It is also able to lock on temporary occluded features as soon as they are appearing in the image and even if all features where occluded the system can recover within in a few seconds.

The gesture recognition module runs in parallel and is able to recognise 12 different gestures running at video frame rate. By decomposing gestures into atomic actions it also provides ways of measuring the conformity of a performed gesture to the gesture definition.

In our future work we plan to develop a 3D adaptive model of the face and use dynamic template acquisition during tracking. Our ultimate aim is to use the facial gesture recognition system in a robotic system for the disabled.

Acknowledgements

This work has been funded by the Australian Research Council and by the generous support of Wind River Systems, supplier of VxWorks.

References

[Azarbayejani *et al.*, 1993] A. Azarbayejani, T. Starner, B. Horowitz, and A. Pentland. Visually controlled graphics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6):602–605, 1993.

[Black and Yacoob, 1995] Black and Yacoob. Tracking and recognizing rigid and non-rigid facial motions using parametric models of image motion. In *Proceedings of ICCV'95*, pages 374–381, 1995.

[Bozic, 1979] S.M. Bozic. *Digital and Kalman Filtering*. Edward Anold Publishers Ltd, 1979.

[Darrel and Pentland, 1995] T. Darrel and A.P. Pentland. Attention-driven expression and gesture analysis in an interactive environment. In *Proceedings of the International Workshop on Automatic Face- and Gesture-Recognition 95*, pages 135–140, 1995.

[Durrant-Whyte, 1995] H. Durrant-Whyte. Autonomous guided vehicle technology. In *Proceedings of the Joint Australia-Korea Workshop on Manufacturing Technology 1995*, pages 51–60, 1995.

[Gee and Cipolla, 1994] A. Gee and R. Cipolla. Non-intrusive gaze tracking for human-computer interaction. *IEEE*, pages 112–117, 1994.

[Hager and Toyama, 1995] G.D. Hager and K. Toyama. A framework for real-time window-based tracking using off-the-shelf hardware. Technical report, Yale University, Department of Computer Science, 1995.

[Hager and Toyama, 1996] G.D. Hager and K. Toyama. Xvision: Combining image warping and geometric constraints for fast vision tracking. In *Proceedings of ECCV'96*, pages 507–517, 1996.

[Heinzmann, 1996] J. Heinzmann. Real-time human face tracking and gesture recognition, *Master's Thesis*, Department of Computer Science, University of Karlsruhe, 1996.

[Jacquin and Eleftheriadis, 1995] A. Jacquin and A. Eleftheriadis. Automatic location tracking of faces and facial features in video sequences. In *Proceedings of the International Workshop on Automatic Face- and Gesture Recognition 95*, pages 142–147, 1995.

[Maurer and von der Malsburg, 1996] T. Maurer and C. von der Malsburg. Tracking and learning graphs and pose on image sequences of faces. In *Proceedings of the International Conference on Automaic Face and Gesture Recognition '96*, pages 176–181, 1996.

[McLauchlan *et al.*, 1994] P.F. McLauchlan, I.D. Reid, and D.W. Murray. Recursive affine structure and motion from image sequences. In *Proceedings of ECCV'94*, pages 217–224, 1994.

[Saulnier *et al.*, 1995] A. Saulnier, M.-L. Viaud, and D. Geldreich. Real-time facial analysis and synthesis chain. In *Proceedings of the International Workshop on Automatic Face- and Gesture Recognition 95*, pages 86–91, 1995.