MIT Media Laboratory, Perceptual Computing Technical Report #401
Submitted to CVPR November 1996

Parametrized Structure from Motion for 3D Adaptive Feedback Tracking of Faces

Tony S. Jebara and Alex Pentland
Media Laboratory, Massachusetts Institute of Technology
Cambridge, MA 02139

November 28th, 1996

Abstract

A real-time system is described for automatically detecting, modeling and tracking faces in 3D. A closed loop approach is proposed which utilizes structure from motion to generate a 3D model of a face and then feed back the estimated structure to constrain feature tracking in the next frame. The system initializes by using skin classification, symmetry operations, 3D warping and eigenfaces to find a face. Feature trajectories are then computed by SSD or correlation-based tracking. The trajectories are simultaneously processed by an extended Kalman filter to stably recover 3D structure, camera geometry and facial pose. Adaptively weighted estimation is used in this filter by modeling the noise characteristics of the 2D image patch tracking technique. In addition, the structural estimate is constrained by using parametrized models of facial structure (eigen-heads). The Kalman filter's estimate of the 3D state and motion of the face predicts the trajectory of the features which constrains the search space for the next frame in the video sequence. The feature tracking and Kalman filtering closed loop system operates at 30Hz.

1 Introduction

Facial pose, 3D structure and position provide a vital source of information for applications such as face recognition, gaze tracking and interactive environments. We describe a real-time system that automatically provides such measurements from real-world video streams. These two key attributes (real-world video and real-time) limit us to the types of image processing we can do. Computations must be fast without sacrificing generality and robustness to a wide variety of face tracking scenarios. We propose a system that involves the marriage of robust face detection and fast face tracking. The system gracefully reverts to face detection when tracking fails and re-initializes fast face tracking anew. Tracking is accomplished by minimizing normalized correlation over translation, rotation and scale. However, tracking is intimately coupled with feedback from a parametrized structure from motion framework. This allows us to overcome some limitations of linearized 2D image patches by the simultaneous recovery of underlying global 3D structure.

Motion provides a strong cue for estimating 3D structure, pose and camera geometry. However, stable and accurate structure from motion has typically been a purely bottom-up approach requiring high quality feature tracking. Moreover, structure from motion (SfM) is usually constrained exclusively by rigidity assumptions. However, it is possible to further constrain the estimation of 3D shape if the range of the 3D structures is defined a priori. In other words, if only faces are to be tracked, SfM can be limited by 3D head models of human faces so that unlikely configurations will be eliminated. We describe a global tracking framework which takes advantage of automatic initialization and 3D parametrized structural estimation to perform reliable feature tracking.

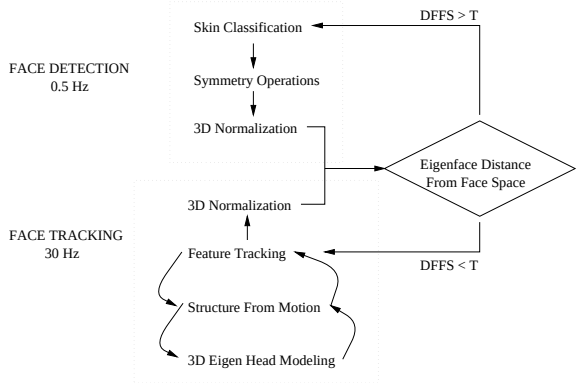


Figure 1: The Integrated System

The details of such a tracking system are discussed starting with initialization which is performed via automatic detection of facial features. The components of our face detection algorithm include skin classification, symmetry transforms, 3D normalization and eigenface analysis. Once initial locations of these facial interest points are determined, the system tracks these features using 2D SSD correlation patches (spanning rotation, scale and translation). However, such tracking alone is incapable of dealing with 3D out-of plane and other non-linear changes. Thus, the 2D tracking and its noise characteristics are coupled to a structure from motion algorithm that simultaneously recovers an estimate of the pose and of the underlying 3D structure of the face. This structure is further constrained by a training set of 3D laser-scanned heads represented as a parametrized eigenspace. This prevents invalid 3D shape estimates in the structure from motion computation. This final filtered 3D facial structure and pose estimate is fed back to control the 2D feature tracking at the next iteration

and overcome some of its inherent 2D limitations.

The fully integrated system is displayed in Figure 1. Note the fast face tracking loop and the slower face detection loop. The system switches between these two modes using eigenface measurements. If the object being tracked is a face, tracking continues. However, if the object being tracked is not face-like, reliable face detection is used to search the whole image for a new face. In addition, note the coupling of feature tracking, structure from motion and 3D eigen head modeling. This closed loop feedback prevents tracking from straying off course.

2 Facial Feature Detection

Automatic face detection and facial feature localization has been a difficult problem in the field of computer vision for several years. This can be explained by the large variation a face can have in a scene due to factors such as facial position, expression, pose, illumination and background clutter. We propose a system that uses simple image processing techniques to find candidates for faces and facial features and then selects the candidate formation that maximizes the likelihood of being a face, thereby pruning the false alarm candidates.

Starting with skin classification, the system finds blob-like regions in the image which might be faces. The symmetry transform is applied to the skin regions to find dark blobs that could be eyes and horizontal limbs that could be a mouth. Simple vertical edge detection yields an approximation for the locus of the nose. A 3D model of the average human head is then aligned to anchor points at the position of the eyes, nose and mouth and warped into a canonical frontal view. By warping the image at various anchor points and minimizing "Distance From Face Space", the system finds the most likely locations of eyes, nose and mouth from all possible candidates. The algorithm [6] is explained in further detail below.

2.1 Skin Classification using EM

Human skin forms a dense manifold in color space which makes it an easy feature to detect in images [10]. We obtain multiple training samples of skin from images of several individuals of varying skin tone and under varying illumination conditions. Each pixel in this distribution forms a 3 element vector, [R G B]. We perform clustering on this distribution of pixels using Expectation Maximization to find a probability distribution model for skin colors. This model is a mixture of Gaussians and cross-validation is used to determine the appropriate number of Gaussians to use in the EM algorithm. The probability distribution model we used is shown in Figure 2 and is described by Equation 1 where $\mathbf{x}$ is an (R,G,B) vector.

$$p(\mathbf{x}) = \sum_{i=1}^{n} \frac{\mathrm{mix_i} \exp\{-\frac{1}{2}(\mathbf{x}-\mu_i)^T \Sigma_i^{-1}(\mathbf{x}-\mu_i)\}}{(2\pi)^{(d/2)} \mid \Sigma_i \mid^{1/2}} \quad (1)$$

When a new image is acquired, the likelihood of each pixel is evaluated using this model and if it is above a threshold of probability, it is labeled as skin. Then, a connected component analysis is used to determine the regions of skin pixels in the image. This process is demonstrated in Figure 3. The largest skin blob is then processed further to search for facial features. It is possible to consider the smaller skin blobs as well in case the face is not the largest skin-colored object in the scene.

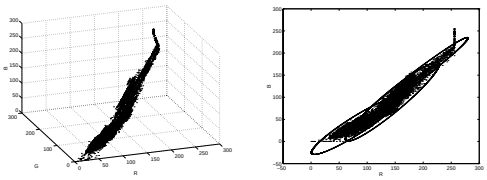


Figure 2: The Skin Color Distribution and the Gaussian Mixture Model

Figure 3: Skin Classification

2.2 Symmetry Transformation

Using the detected skin contour and some simple heuristics, a window can be defined which is expected to contain the eyes. We then propose the use of the dark symmetry transform [3] [7] [9] [6]. This is an annular sampling region which detects edge configurations that enclose an object. However, unlike template matching, a perceptual measure of symmetric enclosure is computed and blob centers are detected. When applied at the appropriate scale within a window defined by the skin contour, this transform consistently detects the eyes in the face. The dark symmetry transform is computed from a phase and edge map by wave propagation (for computational efficiency). For each point in the image p, at each scale or radius r and for each symmetry orientation ψ we find the set of cocircular pairs of edges $\Gamma_{r,\psi}(p)$. The magnitude of axial symmetry in the (p, r, ψ) space is as follows:

$$S_{r,\psi}(p) = \sum_{\lambda_i,\lambda_j \epsilon \Gamma_{r,\psi}(p)} \|\lambda_i\| \, \|\lambda_j\| \, (\sin\phi/2)^{w_1} \quad (2)$$

where $\|\lambda_i\|$ and $\|\lambda_j\|$ are the edge intensities of the two co-circular edges and ϕ is the angle separating their normals.

Then, radial symmetry is determined from the axial symmetry map as in Equation 3 and Equation 4. Finally, the symmetry map undergoes Gaussian smoothing and local maxima are determined.

$$S_\psi(p) = \max_{r=0}^{r_{max}} S_{r,\psi}(p) \quad (3)$$

$$I(p) = \sum_{\psi_i,\psi_j} S_{\psi_i}(p) S_{\psi_j}(p) (\sin(\psi_i - \psi_j))^{w_2} \quad (4)$$

The strongest peaks of dark symmetry are candidates for eye positions. Simple heuristics are used to reject pairs of eyes that have insufficient intra-occular distance (w.r.t the skin blob) and that form an angle larger than 20 degrees from the horizontal. The interest map resulting from the dark symmetry transform is shown in Figure 4.

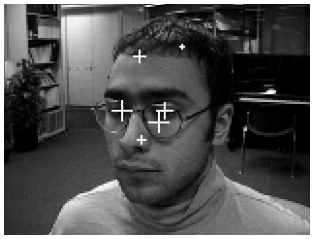


Figure 4: Symmetry Transform's Possible Candidates for Eyes

Horizontal limb extraction is performed to find the mouth from the dark axial symmetry map. The longest linked limb is selected as the mouth.

Additionally, a coarse estimate for the nose's vertical location is found by searching for the strongest vertical gradient in the intensity image that lies in a region bracketed by the eyes and the mouth.

At this stage, a variety of candidates have been detected as possible facial features. These candidates must be tested by more discriminating techniques to discard false alarms and to refine localization.

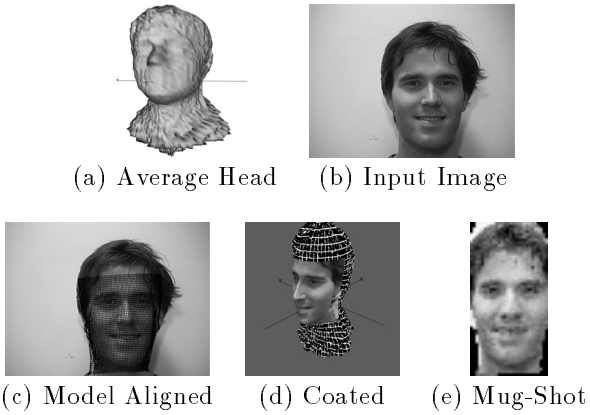
2.3 3D Facial Pose and Directional Illumination Normalization

We begin by considering a set of candidate anchor points for the facial features (eyes, nose and mouth). These may be detected in a variety of configurations. The loci of these feature points gives an estimate of the pose of a face. Unfortunately, not all faces will be facing the camera in a canonical frontal view and this prevents us from using techniques such as eigenspace analysis where correspondance is important. We thus propose to warp a detected face into frontal view using a 3D model of a head.

A 3D range data model of an average human face is formed off-line from a database of range data and is depicted in Figure 5(a). Several Cyberware range models were averaged in 3D to obtain this average head. The eyes, nose and mouth were located on the models and used to align them via a 3D mapping and a vertical stretch into a standard pose. This alignment was done by manually selecting the 4 points and then using a least-squares iterative fit of the 3D anchor points.

Using the computed average 3D model, a Weak-Perspective-3-Points [1] computation can then be used to align its eyes and nose to the ones found in a 2D image. The model is also iteratively deformed by a vertical stretch so that its mouth is also properly aligned with the mouth in the 2D image as shown in Figure 5(c).

Once the optimal 4-point alignment is found, the 2D image's intensity data is mapped onto the 3D structure that now overlaps it. Thus, the 3D mesh is 'coated' with the appropriate intensity values of the underlying 2D image. This coated 3d model is shown in Figure 5(d). If parts of the 3D structure are occluded due to excessive rotation, we use symmetry to mirror the face intensities across the midline of the 3D structure. The 3D structure is then rotated into a normalized frontal view and projected to form a segmented, mug-shot image of the face. Thus, we generate a frontal,



(a) Average Head (b) Input Image

(c) Model Aligned (d) Coated (e) Mug-Shot

Figure 5: Normalizing with 3D Warping and Histogram Fitting

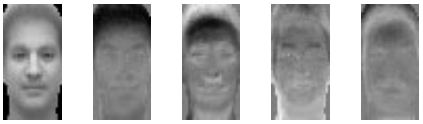


Figure 6: The Mean Face and the First 4 Eigenfaces

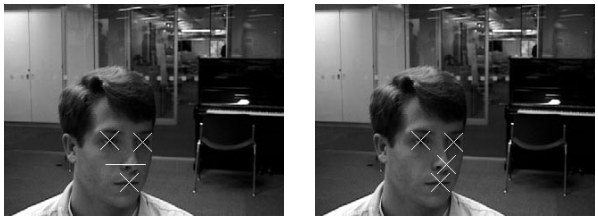
colour mug-shot of the individual from the original image and 4 anchor points corresponding to facial features.

Each side of this new 2D face undergoes histogram fitting to normalize its illumination [11]. Two transfer functions are computed: one for mapping the left half of the face to a desired histogram (i.e. a histogram of a well-illuminated face) and the other for mapping the right half of the face. A weighted mixture of these transfer functions is used as we traverse from the left side of the face to the right side, smoothly removing directional shading of the face. Furthermore, the generation of the transfer functions is windowed to avoid facial hair and head hair so that illumination normalization does not over-brighten mug-shots of bearded men and so on. The fully normalized face is shown in Figure 5(e).

2.4 Eigenspace Distance Measures on 3D Warped Faces

A database of colour face images was collected and for each image the locations of the facial features were manually identified. These loci were then used to generate normalized mug-shots as explained above. In addition, the loci were perturbed with random spatial noise to generate multiple mug-shots of each face with slightly misaligned feature locations. This makes the eigenspace sligthly less sensitive to precise feature localization. A colour eigenspace of these normalized mug-shots is constructed and the mean face and the first 4 eigenfaces are shown in Figure 6.

By projecting a new mug-shot into the span of these eigenvectors, we can compute its coefficients in this new basis as well as the residual error. We also approximate its distance to the training set of faces (distance to face-space) or how 'face-like' it is using this represen-



(a) Initial Localization (b) Final Localization

Figure 7: Localization

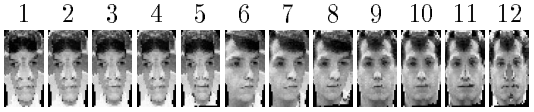


Figure 8: The 3D Normalized Faces for Various Trial Nose Positions and their Corresponding DFFS

tation [8]. The training set of faces is mapped into this eigenspace and the distribution of the coefficients and residuals is modeled as a Gaussian density. The maximum likelihood estimate for the probability of a data point fitting this model is computed using this Gaussian. This gives us a measure of the 'faceness' or how face-like a given mug-shot is (or, conversely, an image with 4 anchor points as it is warped into a mug-shot).

Now, refer to Figure 7(a). Up until now, detection should have recovered a combination of eyes, mouth and nose vertical height. However, it is still uncertain where the exact horizontal position of the nose was on the face. Thus, we attempt 12 different normalizations and K-L projections along the horizontal line across the nose's bottom. The 12 candidate nose anchor points along this line generate 12 normalized mug-shots and their 'distances to face-space'. These are shown as we test for a nose along each point on the horizontal line (Figure 8). Face 0 is generated by setting the nose anchor point all the way to the left of the nose-bottom-line and Face 12 is generated by the anchor point on the right tip of the line. The normalized face vector with the highest 'faceness' probability corresponds to the best possible nose localization (i.e. minimal DFFS).

The final position of the eyes, nose and mouth are shown in Figure 7(b). If time is not critical, we suggest using search or optimization techniques to refine the position of these locations by searching locally for the 3D normalization that minimizes distance to face-space.

The time required for detecting facial feature points is of the order of 1 second. Having found a face and facial feature points that meet a threshold on our 'faceness' measure, we can initialize the tracking system appropriately. Note that, if the face detector was slower than 0.5 to 1 Hz, the tracking could not be initialized properly because the face will probably move away from the localization during the time the detection was being computed.

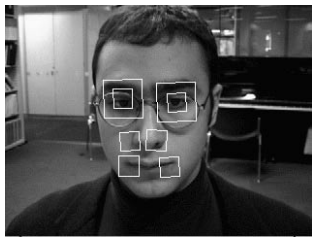


Figure 9: Initialized Correlation Based Trackers

3 2D Feature Tracking

Having determined the locations of facial features in the image, it is now possible to define a number of windows on the face which will be used for template matching via SSD correlation [5]. Using a simple mapping, a set of windows are overlayed upon the face automatically from the data gathered in the face detection stage. A typical initialization result is shown in Figure 9. Eight tracking windows are initialized on the nose, the mouth tips and the eyes automatically as shown. These windowed correlation trackers acquire templates from the image and minimize the SSD of the underlying image patch from one frame to the next. The image patches first undergo contrast and brightness compensation. Registration of the image patch from one frame to the next is accomplished by minimizing the normalized correlation over translation, scaling and rotation parameters. A linear approximation of the behaviour of the image patch under small translation, scaling and rotation perturbations can be used to recover the motion of the image patch. Only simple linear computations are required for this (i.e. no explicit searching) rendering the computation quite efficient.

Given an image $I(\mathbf{x}, 0)$ at time 0, we wish to find μ that minimizes $O(\mu)$ defined in Equation 5.

$$O(\mu) = \sum_{\mathbf{x} \epsilon \Re^2} (\mathrm{I}(\mathbf{f}(\mathbf{x}, \mu), \tau) - \mathbf{I}(\mathbf{x}, 0))^2 \qquad (5)$$

Where $\mathbf{f}(\mathbf{x}, \mu)$ is a motion parametrized by vector μ which allows translation, rotation and scaling. In other words, $\mu = (\mathrm{T_x}, \mathrm{T_y}, \theta, \mathrm{scale})$. Solving for μ in an optimal ℓ_2 sense is performed by computing the pseudo-inverse of a matrix composed of the motion templates. Such a solution for μ is only valid for small displacements and smoothing is used to extend the applicable range of the solution.

The minimum value of $O(\mu)$ is also recovered by the process which gives us a cue for the reliability of the resulting optimal μ.

Unfortunately, minimizing $O(\mu)$ over rotations, scaling and translations cannot account for other 3D or complex changes in the image region. Such changes might be induced by 3D out of plane rotations, occlusions or noise and could easily mislead the estimate of μ. Thus, the correlation window typically loses track of the feature being tracked if it undergoes excessive change beyond the span of the 2D motion model. In addition, due to the local nature of the tracking algorithm, it would be extremely unlikely for feature tracking to recover from this failure without external assistance. Even if multiple features are being tracked, without a strong coupling feature tracking will even-

tually fail. As unpredictable effects such as 3D structure, occlusion and noise, interfere with the 2D tracking, each of the feature trackers will stray off in turn and yield invalid spatial trajectories.

What is desired is a global framework that overcomes some of the difficulties inherent in simple 2D tracking by coupling the individual trackers to a global 3D structure. The outputs of the trackers are integrated appropriately to achieve a global explanation of the scene which can be fed back to constrain their individual behaviour and avoid feature loss.

4 Structure from Motion

Recently, structure from motion has been reformulated into a stable recursive estimation problem and been shown to converge reliably [2]. By remapping the data into a new parametrized representation, what was essentially an under-constrained problem becomes uniquely solvable with no numerical "ill-conditioning".

4.1 Stable Representation for Recursive Estimation

The objective of SfM is to recover 3D structure, motion and camera geometry. These form the "internal state vector", $\mathbf{x}$ of the system under observation. These internal states are to be recovered by observation measurements of the system. For a thorough justification of the internal state vector representation, consult Azarbayejani and Pentland [2]. One internal state parameter is the camera geometry. Instead of trying to estimate focal length to describe the camera, we estimate $\beta = \frac{1}{f}$. The structure of points on the 3D object is represented with one parameter per point instead of an XYZ spatial location. The mapping from this 3 Cartesian form to one parameter is described in Equation 6 where α is the new representation of structure and u and v are the coordinates of the point in the image plane when tracking is initialized.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} (1+\alpha\beta)u \\ (1+\alpha\beta)v \\ \alpha \end{bmatrix} \quad (6)$$

In addition, we define translation as $(t_X, t_Y, t_Z\beta)$. Rotation is defined in terms of $(\omega_X, \omega_Y, \omega_Z)$ which are the incremental Euler angles for the interframe rotation. This representation of rotation overcomes the normality constraints of the quaternion representation by linearizing with a tangent hyper-plane on the unit hyper-sphere formed by the quaternion representation.

The final representation of the internal state vector has a total of $7+N$ parameters where N is the number of feature points being tracked (each of which requires one scalar depth value to determine 3D structure):

$$\mathbf{x} = (\mathrm{t_X}, \mathrm{t_Y}, \mathrm{t_Z}\beta, \omega_\mathrm{X}, \omega_\mathrm{Y}, \omega_\mathrm{Z}, \beta, \alpha_1, \alpha_2, ..., \alpha_\mathrm{N}) \quad (7)$$

At each time step, we also have a measurement or observation vector, $\mathbf{y}$ of size $2N$ with the following form:

$$\mathbf{y} = (\mathrm{X}_1, \mathrm{Y}_1, \mathrm{X}_2, \mathrm{Y}_2, ..., \mathrm{X_N}, \mathrm{Y_N}) \quad (8)$$

Where (X_i, Y_i) are the positions of a feature point currently being tracked in the image. Unlike other formulations which are underdetermined at every time step, the above parametrization of the SfM problem is well-posed when $2N \geq 7 + N$ or when $N \geq 7$. Thus, if 7 or more feature points are being tracked in 2D simultaneously, a unique, well-constrained solution can be found for the internal state and a recursive filter can be employed.

Due to the non-linearities in the mapping of state vector to measurements, an extended Kalman filter is used as the estimator. The dynamics of the internal state are trivially chosen to be identity with Gaussian noise for each time step.

4.2 Mapping 2D Feature Tracking into the Kalman Filter

As was discussed previously, each feature tracker recovers an optimal μ motion parameter by minimizing $O(\mu)$. However, since the 2D feature tracking in question was being used to recover translation, rotation and scale, the μ vector has 4 degrees of freedom (not merely 2). We can represent these 4 degrees of freedom as 2 point features that are free to translate independently. In other words, two arbitrary points on the correlation window are selected (i.e. 2 opposing corners) and it is trivial to compute their locations from a corresponding μ transformation (translation, scale and rotation). This mapping goes both ways and we can model the 2D tracking for each image patch with the SSD model using μ or using the positions of 2 distinct feature points somewhere within the window (X_1, Y_1, X_2, Y_2). For M correlation-based windows, we compute the (X,Y) location of $N = 2M$ points. These feature points are then arranged into the $\mathbf{y}$ vector for input into the EKF.

4.3 Mapping Residuals to Spatial Uncertainty

At each iteration, one more output can be recovered from the 2D correlation based tracker in addition to μ. The output in question is the actual value of the residual $O(\mu)$. This residual can be used to weight the input measurements feeding into the Kalman filter. Thus, if a feature has a very high residual, the filter should trust its spatial information less and focus on other feature tracks. In addition, if a feature is lost or occluded, its correlation window will have a very high residual error and the Kalman filter should essentially ignore its contribution to the estimation.

Recall that Kalman filtering uses a noise covariance matrix to describe the expected noise on input measurements. Traditionally, the noise covariance matrix is denoted R and is $n \times n$ where n is the number of measurements in the observation vector $\mathbf{y}$. The role of R in the computation of the Kalman gain matrix described by Equation 9. Adaptive Kalman filtering [4] proposes the use of a dynamically varying R matrix that changes with the arrival of new observation vectors to model the confidence of the new data. By changing R using the values of the residuals of the 2D correlation based trackers, we can assign a weight on the observations they provide and end up with a more robust overall estimate of internal state.

$$K = P^- H^T [H P^- H^T + R]^{-1} \quad (9)$$

At this stage, we address the issue of relating residuals from the correlation-based trackers to the noise covariance matrix on the feature points being tracked for Kalman filtering. We propose fitting a function that models the residual as a function of spatial uncertainty.

Consider, first, the simple case of SSD tracking with only translational motion. We observe the residuals between an image patch $I(\mathbf{x}, 0)$ and the same image patch

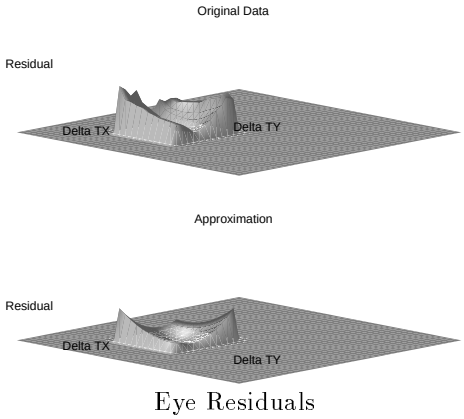


Figure 10: Residuals for a Correlation Template and Parabolic Approximation

after a given translation $I(\mathbf{f}(\mathbf{x},\mu),0)$. The residual is expected to grow as alignment errors increase and this value is plotted over various perturbations in x and y translation $(\Delta x, \Delta y)$ as shown in Figure 10. We can model this residue function with respect to $(\Delta x, \Delta y)$ as a 2D paraboloid centered at $(0,0)$ by sampling various values of $(\Delta x, \Delta y)$ and fitting in a least-squares sense. The result of this fitting is a fitted paraboloid as shown in Figure 10. Note that these residual error functions or paraboloids have different shapes for different textures over which normalized correlation is to be applied.

Extending this concept to 4D (the true dimension of μ in our application), we can compute a 4D paraboloid which maps the spatial error in alignment to correlation residue error. This process is performed on all image patches being tracked each time the system is initialized. The perturbations on μ are computed for a variety of $\Delta\mathbf{X} = (\Delta \mathrm{X}_1, \Delta \mathrm{Y}_1, \Delta \mathrm{X}_2, \Delta \mathrm{Y}_2)$ perturbations and a 4D paraboloid of the form in Equation 10 is found.

$$\sqrt{SSD} = \Delta\mathbf{X} \begin{bmatrix} a_{xx} & a_{xy} & a_{xm} & a_{xn} \\ a_{xy} & a_{yy} & a_{ym} & a_{yn} \\ a_{xm} & a_{ym} & a_{mm} & a_{mn} \\ a_{xn} & a_{yn} & a_{mn} & a_{nn} \end{bmatrix} \Delta\mathbf{X}^{\mathrm{T}} \quad (10)$$

Having solved this 4D paraboloid, we can find the 4D ellipsoid that corresponds to a given value of residue directly. The surface defined by the 4D ellipsoid is essentially the error window on the current estimate of (X_1, Y_1, X_2, Y_2) from the correlation based tracking. Under the paraboloid noise model, it is straightforward to show that the 4D iso-residual surface (the ellipsoid) is also a 4D iso-probability surface for a Gaussian model of the spatial noise on the current estimate of (X_1, Y_1, X_2, Y_2). Thus, the Gaussian error on the current feature points can be estimated by the following 4×4 covariance matrix in Equation 11.

$$C \propto \sqrt{SSD} \begin{bmatrix} a_{xx} & a_{xy} & a_{xm} & a_{xn} \\ a_{xy} & a_{yy} & a_{ym} & a_{yn} \\ a_{xm} & a_{ym} & a_{mm} & a_{mn} \\ a_{xn} & a_{yn} & a_{mn} & a_{nn} \end{bmatrix}^{-1} \quad (11)$$

For each of the N correlation windows in the tracking, a 4x4 sub matrix of the form of C can be computed and these are placed into the matrix R in the Kalman filter. For feature i, we compute a noise covariance C_i and place it into R which becomes block-diagonal as shown in Equation 12.

$$R = diag(C_1, C_2, ...C_N) \quad (12)$$

At each iteration, the rotation, scaling and residue of a correlation window determine the rotation and scaling of the covariance sub-matrix C_i associated with it. Thus, R is adaptively adjusted to reflect the noise on the spatial position of the feature points being tracked. In addition, these covariances are determined by a sensitivity analysis and are specialized to the noise characteristics of the particular texture being tracked.

Thus, at each iteration, we have an appropriate weighting of feature tracks determined by the current orientation and scale of the correlation trackers as well as their residual values and the spatial sensitivity of the textures they have been initialized to track. The Kalman filter abstracts the rest of the estimation and returns the structure, motion and camera geometry optimally from the weighted set of inputs.

5 Initialization and Parametrization of the Kalman Filter State Vector

Since the particular objects being tracked by the system are faces, we can initialize the system with a 3D model of the structure of a head to speed up convergence of true structural motion. In addition, during tracking and estimation, a more constrained set of 3D configurations for the structural estimate in the SfM solution is expected. Only faces are being tracked so we do not wish to allow the structural estimate of the SfM computation to diverge to another shape. Thus, we propose filtering the estimated 3D structure computed by the EKF to avoid any unreasonable estimates. This is done by constructing an eigenspace filter from a set of previously scanned 3D structures.

Recall the set of cyberware heads used to generate the average 3D human head for face detection. These 3D models have all been aligned into frontal view. When automatic face detection determines the loci of eyes, nose and mouth, it aligns a 3D average head model to these locations. Thus, it automatically has an estimate of the depth map of the face and the depth values at the positions of the feature points to be tracked are sampled. In addition, the system has an estimate for the 3D pose of the face $(T_X, T_Y, T_Z, \theta_X, \theta_Y, \theta_Y)$. The SfM state vector can thus be initialized (camera geometry is arbitrarily set to $\beta = 0.5$) using much of the information from the previous face detection stage which gives us $\mathbf{x}_{\mathrm{t}=0} = (\mathrm{T_X}, \mathrm{T_Y}, \mathrm{T_Z}\beta, \theta_\mathrm{X}, \theta_\mathrm{Y}, \theta_\mathrm{Y}, \alpha_1, \alpha_2, ..., \alpha_\mathrm{N})$.

During tracking, the structural estimate can also be filtered to prevent any non-face-like structural estimates. Recall that the average 3D head model was aligned to the locations of the eyes, nose and mouth. Susbequently, the 3D model of the average head generates a depth map to find the initial values for $(\alpha_1, \alpha_2, ..., \alpha_N)$. This is also done for each of the other cyberware heads so that multiple vectors of $\alpha = (\alpha_1, \alpha_2, ..., \alpha_\mathrm{N})$ are generated. We perform a Karhunen-Loeve decomposition on 12 such α vectors from our 12 cyberware 3D head models and obtain a parametrized representation of the structure.

The eigenspace is computed each time the system is initialized since the parametrization of structure

$(\alpha_1, \alpha_2, ..., \alpha_N)$ depends on initial feature positions in the image plane. However, due to the small size of the training set, this computation is trivial.

A linear subspace is formed from the first 4 eigenvectors of this eigenspace (the eigen-α-structures). At each time time step, we project the Kalman filter's current estimate of structure into this eigenspace. Thus, the N degrees of freedom in the structural estimate are constrained by the 4 degrees of freedom in our linear subspace of facial structure. Equation 13 maps the current structure vector into an eigenspace parametrization by projection onto the eigenvectors $\mathbf{e}_i$. Equation 14 reconstructs the filtered structure vector, $\hat{\alpha}$. Thus, constraints are introduced into the loop by filtering the recovered SfM information with an eigenspace.

$$c_i = \alpha \cdot \mathbf{e}_i \tag{13}$$

$$\hat{\alpha} = \sum_{i=1}^{i=4} c_i \mathbf{e}_i \tag{14}$$

6 System Integration and Feedback

We now go over the implementation details of the system integration and the feedback process. The system begins with the face detection loop and repeats until a face is detected and satisfies a threshold on distance from face-space. The facial features detected are eyes, nose and mouth. From these features, a set of templates can be placed on the face (one on each tip of the mouth, one on each side of the nose, and two for each eye). These acquire the underlying texture and then a sensitivity analysis is performed to obtain the mapping between spatial uncertainty and correlation residual. A depth map of the face is obtained by fitting a 3D model to the position of the features and this is used to initialize the depth parameters of a Kalman filter that recovers structure from motion.

The correlation-based feature trackers begin by tracking in a nearest-neighbour sense and search locally for the facial features. However, at each iteration, the Kalman filter computes an estimate of the rigid 3D structure that could correspond to the motion of the set of 2D SSD trackers. This global estimate is weighted using the noise characteristics and residuals of the 2D tracking. Once this structure is computed and an estimate of orientation and camera focal length are found, the 3D structure is filtered using an eigenspace of 3D head shape. The final 3D structure, motion and focal length are used to projected feature points back onto the image to determine an estimated position of the 2D feature trackers. Then, at the next frame in the sequence, correlation-based search is performed starting at this 3D estimated position as well as starting at the original destination of the feature track. The best match of these two searches is then fed back into the Kalman filter as the 2D spatial observation vector and the loop continues. Two searches are performed for each SSD tracker since the EKF may possibly perform worse than straight neareset-neighbour searching before structural convergence. The feedback from the adaptive Kalman filter maintains a sense of 3D structure and enforces a global collaboration between the separate 2D trackers.

In addition, at each iteration, the orientation of the face is computed and is used to warp the face image back into frontal view to compute 'distance to face space'. If DFFS is below a threshold, tracking continues. Otherwise, the system reverts back to the initial detection stage.

7 Testing and Performance

The full detection and tracking loop was tested on live video streams. Typically, detection found a face within 1 or 2 loops and was able to handle $\pm$ 20 degrees rotation in-plane as well as roughly $\pm$ 20 degrees rotation out-of-plane. This flexibility is due to the rather lax constraints on feature detection and the heauristics in the search. However, the consequent false alarms are eliminated by using 3D normalization and a strict eigenspace DFFS technique. Thus, subjects do not need to look explicitly at the camera for tracking to commence since detection can handle non-frontal views. Detection has been tested successfully in a wide variety of backgrounds, under many views and with numerous subjects. The system was used to detect facial features in the Achermann face database (courtesy of the University of Bern in Switzerland) and obtained over 90% success even though the skin classification stage was not used (the images were gray-scale). The database contains 30 individuals in 10 different views (of which 8 involve significant out-of-plane rotation).

Real-time tracking was tested on the live video sequence shown in Figure 11. Roughly 2000 frames were tracked without feature-loss (over 1 minute of tracking in real-time). The filtered tracking windows are shown projected on the face. The normalized mug-shot (after 3D warping and illumination correction) is shown at the bottom of Figure 11.

As can be seen, the subject is undergoing large in-plane and out-of plane rotations in all axes as well as partial occlusion (in frame 827). Out-of-plane rotations of over $\pm$ 45 degrees are tolerated without feature loss. Even though almost half of the correlation-based trackers may be occluded under large, out-of-plane rotations, the global EKF filtering maintains tracking using the visible features. Unless very jerky motion is used or extreme out-of-plane rotations are observed, the system maintains tracking and does not exhibit instability. The system has been tested on multiple subjects from live video streams and tracking performance is consistent.

Figure 12(a) displays the typical residual correlation error of a tracking window. However, this noisy behaviour is filtered and a stable estimate of depth structure is obtained in Figure 12(b). The EKF converges quickly to the true underlying 3D geometry despite noisy feature tracking. We also measured the SSD residual between the initial mug-shot (at frame 0) and the current normalized face. Figure 12(d) displays the DFFS value over the sequence which is used as a cue to stop tracking (when DFFS is too large). In this sequence, the threshold was set to a generous value of 0.5 and face detection was not re-used since tracking did not fail. However, if the DFFS value were to exceed 0.5, tracking would stop and detection would search for a new face.

8 Conclusions

We have presented an integrated system for detecting, modeling and tracking faces in real-time. The system uses detection to automatically initialize a tracking system and to re-initialize upon failure. The track-

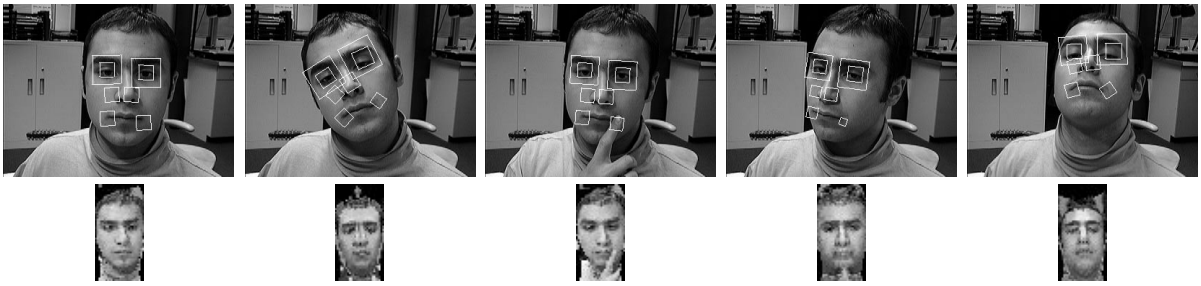


Figure 11: Real-Time Closed-loop tracking of a sample video sequence.

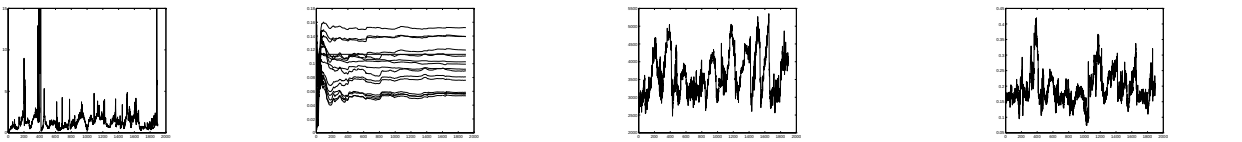


Figure 12: EKF Estimates and Residual Errors.

ing system uses a feedback approach to stabilize 2D correlation-based trackers by recovering structure from motion and constraining structure with learned 3D facial geometry. Adaptive Kalman filtering is used to weight features by determining a mapping between 2D spatial tracking accuracy and textures and correlation residuals. The system achieves greater stability under 3D variations, occlusion and local feature failure since a global estimation framework links the individual trackers by acquiring the underlying 3D structure of the face. The system is demonstrated on live video sequences where it tracks large out-of-plane rotations stably.

We are currently investigating more sophisticated representations of the 3D model of facial structure to better constrain the structure from motion problem. In particular, it is possible to place the model's structural parameters (i.e. the coefficients of the eigenspace) directly into the EKF as parameters in its internal state vector. This would replace the current estimation and post-processing of point-wise depth structure. The linearization in the EKF would be performed on our eigenspace of 3D heads directly and would be used to form the Jacobians for the estimation of internal state.

References

[1] T.D. Alter, "3D Pose from 3 Corresponding Points under Weak-Perspective Projection", *A.I. Memo No. 1378*, 1992.

[2] A. Azarbayejani and A. Pentland, "Recursive Estimation of Motion, Structure and Focal Length", *IEEE Pattern Analysis and Machine Intelligence*, June 1995.

[3] M. Bolduc, G. Sela and M.D. Levine, "Fast computation of multiscalar symmetry in foveated images", *Proceedings of the Conference on Computer Architectures for Machine Perception*, pp. 2-11, 1995.

[4] A. Gelb, "Applied Optimal Estimation", M.I.T. Press, 1996.

[5] G.D. Hager and P.N. Buelhumeur, "Real-Time Tracking of Image Regions with Changes in Geometry and Illumination", *IEEE Conference on Comnputer Vision and Pattern Recognition*, pp. 403-410, 1996.

[6] T. S. Jebara, "3D Pose Estimation and Normalization for Face Recognition", Bachelor's Thesis*McGill Centre for Intelligent Machines*, 1996.

[7] M.F. Kelly and M.D. Levine, "Annular Symmetry Operators: A Method for Locating and Describing Objects", *Fifth International Conference on Computer Vision*, pp. 1016-1021, 1995.

[8] B. Moghaddam and A. Pentland, "Probabilistic Visual Learning for Object Detection", *Fifth International Conference on Computer Vision*, pp. 786-793, 1995.

[9] D. Reisfeld and Y. Yeshurun, "Robust detection of facial features by generalized symmetry", *11th IAPR International Conference on Pattern Recognition*, Vol. 1, pp. 117-120, 1992.

[10] B. Sheile and A. Weibel, "Gaze Tracking Based on Face Color", *International Workshop on Face and Gesture Recognition*, Zurich, July 1995.

[11] P.J. Phillips and Y. Vardi, "Data-driven Methods in Face Recognition", *International Workshop on Face and Gesture Recogntion*, pp. 65-70, 1995.