On-line Face Tracking Using a Feature Driven Level-set

Derek Magee
School of Computing,
University of Leeds,
Leeds, LS2 9JT , UK.
drm@comp.leeds.ac.uk

Bastian Leibe,
Department of Computer Science
E.T.H.
Zurich, CH-8092, Switzerland
leibe@inf.ethz.ch

Abstract

An efficient and general framework for the incorporation of statistical prior information, based on a wide variety of detectable point features, into level set based object tracking is presented. Level set evolution is based on the maximisation of a set of likelihoods on mesh values at features, which are located using a stochastic sampling process. This evolution is based on the interpolation of likelihood gradients using kernels centred at the features. Feature detectors implemented are based on moments of colour histogram segmented images and learned image patches located using normalised correlation, although a wide variety of feature detectors could be used. A computationally efficient level set implementation is presented along with a method for the incorporation of a motion model into the scheme.

1 Introduction

Level Set methods (or 'Geodesic Active Contours') [13, 8, 7] are becoming increasingly popular tools for the segmentation of medical images and volumes. Such techniques are based on modelling an evolving surface. The dynamics of these surfaces are defined based on image/volume data. Such methods are less popular in the area of object tracking and dynamic scene analysis, with more conventional parametric models such as snakes [6] or Active Shape/Appearance Models [3] being favoured. There are two possible reasons for this lack of popularity; The computational cost of level set methods and the difficulty in incorporating prior information in the level set framework. The computational problem can be overcome using multi-scale approaches [12] and approximate methods such as narrow banding and fast marching [13]. The problem of incorporating shape, appearance, and other priors has also been tackled (e.g. [7, 8]). Against these disadvantages the disadvantages of parametric methods must be weighed. The principal disadvantage of parametric methods is the definition of the parameterisation. This is always a trade-off between generality and specificity. This is often a tough trade-off if there is large variation within the data class of interest, and it is often not possible to represent valid members of the class and possible to represent invalid members of the class.

In this paper an efficient and general framework for incorporating prior information into level set based object tracking is presented. This is based around a feature driven level set method, where the evolution of the level set mesh is based on probabilistic priors on the location of, and level set mesh values relating to, detected features. The nature of the features used in this framework is constrained only by the fact that priors on level

BMVC 2003 doi:10.5244/C.17.43

set mesh values relate to a single point for each feature. Features could be based on appearance, edges, colour, shape, texture, wavelet decomposition etc. etc. In our implementation we have used features based on moments of a binary image segmented using a (HSV) colour histogram, Learned image patches identified using normalised correlation and shape features which use no image information (see section 5). An efficient level set mesh approximation method is presented and a method for the incorporation of a motion model is also presented.

2 Background: Level Sets, Priors and Object Tracking

Level set methods [13] are a type of finite element approach used for the modelling of evolving curves or surfaces. These methods have been widely used in the fields of fluid mechanics and material science for some time and are popular within the field of machine vision for segmentation problems (e.g. [10]). The principal idea behind level set methods is the definition of a static, evenly spaced mesh in an image or volume space. The values at each point on the mesh relate to the proximity of the mesh point to an evolving curve or surface with the 'level set of zero' defining the location of the curve or surface (this can be thought of as like a contour line on a map). Mesh points contained within the evolving surface are given negative values and mesh points outside the surface are given positive values. A 'speed function' for the movement of the curve or surface is defined and mesh values are updated (from an initial value) using a discrete time finite element update as described in equation 1.

$$\psi_{t+1} + F|\bigtriangledown \psi_t| = 0 \tag{1}$$

Where ψ_t is the matrix of mesh values at time t, F is a speed function and $\bigtriangledown$ is a suitable spatial difference operator. The definition of the speed function F is crucial to the operation of the method. Recently Leventon *et al.* [7, 8] have tried to re-pose level set mesh evolution as the optimisation of a probabilistic likelihood function (based on a learned prior probability distribution) with respect to observed data. In such cases F is defined as a local differential that will maximise this probability. The applications described are in the domain of medical volume segmentation using priors on shape and image edge/gradient strength [8] and intensity and curvature [7]. In [8] shape is represented by the mean and distribution of the mesh values (using Principal Components Analysis) which are translationally and rotationally normalised. This is essentially an Eigenimage [14]. The computational (and other) problems of the Eigenimage approach are well understood, which makes them a poor choice for an on-line tracking application such as ours.

Paragios and Deriche [12] present a level set based system for object tracking using inter-frame difference images. The aim is to classify pixels as foreground or background based on priors relating to the difference image and mesh values at local neighbourhoods. No shape or appearance prior information is included in this scheme.

In this paper we propose a framework to include arbitrary shape and appearance information priors into a level set based object tracking system, by encoding these priors as distributions relating to expected mesh values at detected 'features' (localised configurations of the image or level set mesh). This framework could also potentially be applied to image/volume segmentation, however this is beyond the scope of this paper.

3 Feature Driven Level Sets

In this section a general 'feature based' framework for incorporating statistical priors into Level Set mesh evolution/update is presented. This work differs from previous work on incorporating priors into Level set evolution, in that priors are distributions relating to expected (interpolated) mesh values at detected features (which may lie at any point on or off the zero level set), rather than functions based on zero level set points [7] (and their neighbourhoods) or priors on the set of level set mesh values themselves [8].

In our implementation features in an image are identified by a stochastic sampling process (section 3.1), although any appropriate search strategy may be used within this framework. Identified features have an associated prior probability distribution on the value of the level set mesh at their location which relates to the particular feature/feature type located. Ideally this is learned from segmented training data (section 6). We approximate this prior as a 1D Gaussian distribution, however any suitable distribution (that is easily differentiable) could be used. From the prior and the current mesh value at the feature a mesh probability gradient ($\frac{dP(\psi)}{dN}$) may be defined along the mesh normal at the feature:

$$\frac{dP(\psi)}{dN} \approx \frac{dP(\psi)}{d\psi} \times 1 = \frac{dP(\psi)}{d\psi} \tag{2}$$

The approximation given by equation 2 is valid as the relationship between the mesh value (ψ) and the distance along the normal (N) is approximately linear with a unit scale factor. In the general case mesh normals may be estimated from the mesh (see [13], page 70). However, for near circular objects, such as the human head, the vector from the centroid of the object to the feature is a computationally more efficient measure to calculate and, in addition, is less sensitive to local noise that the mesh based normals. The number of steps required to update the mesh to the mean of $P(\psi)$ for feature n is calculated as:

$$S_n = (\bar{\psi_n} - \psi_{x,y}) \,/ \frac{dP(\psi_{x,y})}{d\psi} \tag{3}$$

To update the level set mesh the probability differentials (projected into the normal direction of the zero level set point) are interpolated at the zero level set points using Kernels ($K_n(\psi)$) centred at the features:

$$F_m = \frac{\sum_{n=1}^{N_f} K_n(\psi_m) \frac{dP(\psi_n)}{dN_n} (N_n \cdot N_m)}{\sum_{n=1}^{N_f} K_n(\psi_m)} \frac{\sum_{n=1}^{N_f} S_n}{N_f} \tag{4}$$

Where $(N_n \cdot N_m)$ is the dot product of the unit normal at feature n with the unit normal at zero level set point m (clipped above zero). We use a kernel of $K_n(\psi) = k/(k + |\psi - \bar{\psi_n}|)$ for this interpolation, although different interpolating kernels have not been widely explored. To ensure a smooth mesh evolution F_m is limited to 1 standard deviation above/below the mean over all zero level set points. Mesh values are updated using the value of F_m from the nearest zero level set point as is standard (equation 1).
Note: The level set mesh need not have the same resolution as the image. If resolutions differ locations and mesh values must be scaled using a linear factor (we use a 320x240 image resolution and a 160x120 mesh resolution).

3.1 Location of 'Features' Using Stochastic Sampling

Given we have a set of 'feature detectors' (section 5), that are essentially binary point classifiers, an efficient search procedure is required to locate instances of these 'features' in a novel image. As stochastic search procedures are currently widely used for this purpose, and generally regarded to have acceptable performance, (e.g. the Condensation algorithm [5]) we have taken this approach in our implementation. However, our architecture does not preclude alternative search methods such as hill climbing / gradient ascent/descent [2] or Levenberg-Marquardt optimisation [4] if a feature detector suits them more (they do not for our classifiers).

Our method is as follows:

- Select a feature n to locate from an occurrence prior distribution $P_{occ}(n)$
- Sample a single location of this feature based on a location prior P_{loc_n}
- If the feature is located add this feature (and it's location) to a list of feature instances used to update the level set
- Reduce the prior occurrence probability of $P_{occ}(n)$ by multiplying by a constant factor K_p (and re-normalising the distribution)
- Repeat until a fixed no. of features have been found or a fixed number of iterations have passed

4 An Efficient Level Set Mesh Representation for Object Tracking

Level set based methods, in general, are not highly computationally efficient, as calculations must be performed over all points on a fixed mesh. This is a disadvantage for on-line tracking applications. In image analysis applications mesh values usually relate to pixels (or Voxels in the case of 3D data), which can lead to relatively large mesh sizes. In order to combat this problem narrow band and fast marching approximations to level set methods have been developed (see [13]). Narrow band methods only calculate mesh values that are close to the zero level set and fast marching methods calculate mesh value approximations by working outwards from the zero level set. For the narrow band method mesh points not evaluated at the time of level set evolution are simply marked as such. If these values are required at a later stage (as in our system) they must be calculated as +/- the distance from the nearest zero level set point (as in the standard update), or approximated as the distance from the nearest calculated level set mesh point. Either option requires a search to determine which is the nearest zero level set point or calculated mesh point, which is a computationally expensive operation when repeated numerous times.

Our alternative approach to narrow band and fast marching methods is in fact an approximation to a narrow band method, however the narrow band is defined as a rectangular area around the zero level set (figure 1), rather than an exact narrow band. We shall refer to this as the 'local mesh'. This method requires marginally more mesh points to be evaluated than a true narrow band method, however it allows uncalculated mesh points to be estimated using a simple lookup and distance calculation as illustrated in figure 1 (a much more efficient procedure).

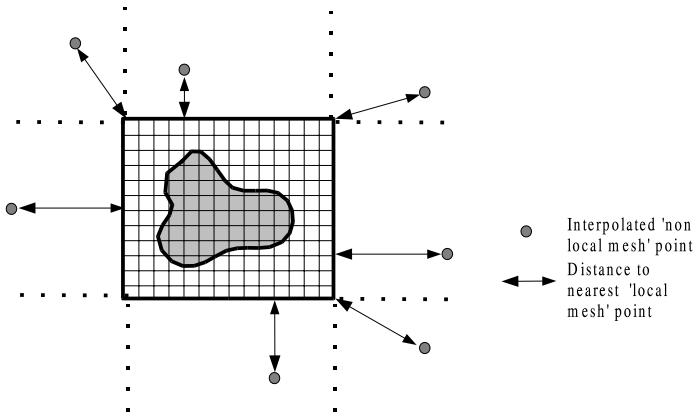

Figure 1: Interpolation of non local mesh values

Non local mesh values may be approximated (as and when needed) as the value of the nearest local mesh point (a simple lookup operation) plus the distance from the non local mesh point to this mesh point. After each level set mesh update (move or feature update) the scope of the local mesh values (min/max x,y) to be explicitly calculated is revised, with uncalculated columns or rows of mesh values estimated from their neighbours using the non local mesh approximation described. The scope is increased or decreased in each direction until all the mesh values in the rows/columns directly outside the scope would be above a maximum level set value (ψ_{max}) and at least one of the mesh values in the rows/columns directly inside the scope is below ψ_{max}. The choice of ψ_{max} depends on the magnitude of the change in level set mesh values at each step, ψ_{max} should be larger than the largest of these (typically we use $\psi_{max} = 5$). An identical approach could be taken to the interior of the object, however it has not currently proved necessary to implement this.

4.1 Level Set Motion Based Update

Most successful object trackers include a dynamic/motion model and a method for incorporating this into the spatial estimation process. Our implementation is no exception. Perhaps the most commonly used motion model is the Kalman filter (see for example [2]). This can estimate future position/configuration (or derivatives of these) based on past values and a fixed motion model (e.g. constant velocity, constant acceleration etc.). We take an alternative approach of estimating velocity from a separate process using the current and previous frame only. This is achieved by calculating the centroid of a binary image segmented using a colour histogram method (required for our Colour Histogram Moment Features, section 5.1). Although the centroid of this image is not an accurate estimate of the centroid of the face, the differential of this centroid over time is a very accurate estimate of the differential of the true face centroid. This is due to the fact there is very little shape change (due to pose change, lighting etc.) from one frame to the next. This method of velocity estimation greatly outperforms the Kalman filter in our scenario, as a single fixed motion model assumption is far from accurate. However, this approach is far from globally applicable, as it relies on a single object of distinctive colour being tracked. This motion model could easily be replaced with another motion model (such as the Kalman filter) within our framework. Once a motion estimate is obtained the Level set mesh can simply be updated using the following equation:

$$\psi_{x,y,new} = \psi_{x-dx,y-dy,old} \tag{5}$$

Where $\psi_{x,y}$ represents the level set mesh values, and dx, dy is the estimated object velocity. It should be noted that if the velocity exceeds the maximum level set value (ψ_{max} in section 4) care must be taken to preserve an accurate representation of the object shape in the level set mesh values. This is because the set of values $\psi_{x-dx,y-dy,old}$ for all mesh points x, y within the explicitly estimated local mesh may lie completely outside the object for larger dx, dy (i.e. the local mesh then represents only an area where the object is not present). Solutions to this are i) Performing the update in smaller steps, ii) Moving the scope of the explicitly calculated level set mesh values w.r.t. the estimated velocity. Of these ii) is the least computationally intensive.

5 Feature Learning and Selection

The framework presented within this paper allows for the inclusion of arbitrary types (and numbers) of feature detectors. In this context a feature detector is any algorithm that can locate a point within an image that relates to a salient feature (e.g. a human eye in our chosen example). Within the scope of the sampling scheme described in section 3.1 feature detectors are binary point classifiers that give a positive output if a feature is believed to be present. The following sections describe a number of feature detectors that we have implemented.

5.1 Colour Histogram Moment Features

This feature detector is based on a binary image formed using a colour histogram constructed from skin coloured patches. It has been widely observed (see for example [11]) that pixels relating to skin form a compact region in colourspace, in particular in the normalised HSV (Hue-Saturation-Value) colourspace we use. It should be noted that colour values with particularly high or low brightness (V) values provide unreliable colour information and are discarded in the training phase. Pixels are classified as skin colour if the probability of the histogram bin relating to the pixel colour is greater than a threshold value.

Figure 2: Skin coloured pixels classified using a HSV histogram

From figure 2 it can be seen that the colour histogram method does a reasonable job of classifying skin/non-skin pixels, although there is some mis-classification of hair and clothing. The neck area is also classified as skin (as would be expected) if visible. Normalised moments (about the x or y axes) may be calculated as:

$$M_x(x) = \frac{\sum_{n=-S}^{S} C_{x,y+n} \times n}{\sum_{n=-S}^{S} |n|} \quad , \quad M_y(y) = \frac{\sum_{n=-S}^{S} C_{x+n,y} \times n}{\sum_{n=-S}^{S} |n|} \tag{6}$$

Where $C_{x,y}$ is the binary classification of pixel x, y (1=skin, 0=not skin), and S is the scope/range of the moment (in the range of 10-50 pixels in our examples). To form binary

classifier, which may be used as a feature detector, upper and lower limits are put on the value of the moment (typically separated by only a small constant e.g. 0.05). If the moment around a given pixel is between these limits then the feature is present. This feature detector is useful for identifying the top, middle and sides of a face.

5.2 Normalised Correlation Features

A set of correlation kernels based on image patches is learned using a method inspired by the work of Agarwal and Roth [1]. From a set of example images (around 15-20 examples each of 10 individuals) and corresponding segmentations (from an early version of the tracker) image patches (lying within the segmentation) of size 9×9 pixels are extracted centred at locations identified using the Harris interest point detector. Starting with each patch as a separate cluster, agglomerative clustering is performed: the two most similar clusters C_1 and C_2 are merged as long as the average similarity between their constituent patches (and thus the cluster compactness) stays above a certain threshold t:

$$similarity(C_1, C_2) = \frac{\sum_{p \in C_1, q \in C_2} NGC(p,q)}{|C_1| \times |C_2|} > t, \tag{7}$$

Where the similarity between two patches is measured by Normalised Greyscale Correlation (NGC).

$$NGC(p,q) = \frac{\sum_i (p_i - \overline{p_i})(q_i - \overline{q_i})}{\sqrt{\sum_i (p_i - \overline{p_i})^2 \sum_i (q_i - \overline{q_i})^2}} \tag{8}$$

This clustering scheme guarantees that only those patches which are visually similar are grouped, and that the resulting clusters stay compact. For each resulting cluster, we compute the cluster centre for use as a correlation kernel. Kernels with frequency of occurrence above a threshold and (normalised) location standard deviation below a second threshold are selected as 'salient kernels' (figure 3).

Figure 3: 9x9 Correlation Kernels Learned and Selected

A feature detector is formed by performing normalised correlation of an image patch with these kernels at a point of interest. If the correlation value is above a threshold (we use 0.4) the feature is determined to be present.

5.3 Shape Prior (Featureless) Features

Shape prior features differ from the other features described, in that their outputs are not a function of the input image. Essentially shape prior features are a binary classifier that always detects the presence of a feature. The location of these features is a function of i) Learned position (a Gaussian distribution relating to the scale and positional normalised feature position), and ii) Scale and Object Centroid Location estimated from other features and the level set as:

$$C_{xy} = \frac{1}{N_{lsp}} \sum_{n=1}^{N_{lsp}} L_{xy}(n) \tag{9}$$

Where C_{xy} is the estimated centroid, $L_{xy}(n)$ is the location of zero level set point n, and N_{lsp} is the number of zero level set points.

$$Sc = \frac{1}{N_{sf}} \sum_{n=1}^{N_{sf}} S_{est}(n) \tag{10}$$

Where Sc is the estimated scale, N_{sf} is the number of features at previous timestep providing scale estimates, and $S_{est}(n)$ is the scale estimate from feature n. The scale estimates $S_{est}(n)$ are estimated from the relative location of the feature and the mean of its location prior:

$$S_{est}(n) = \frac{|F_{xy}(n) - C_{xy}|}{|\delta_{xy}(n)|} \tag{11}$$

Where $F_{xy}(n)$ is the location of instance of feature n and $\bar{\delta}_{xy}(n)$ is the mean offset from the centroid of feature n (at scale 1). In fact the estimated scale and centroid are used in the same way to determine sample location for all features in our implementation.

6 Learning Priors for Features

Priors required by the system fall into three categories; i) feature Level-set mesh value priors, ii) feature occurrence priors and iii) feature location priors.

Two elements are required in order to learn priors relating mesh values at features; i) A set of example feature locations, ii) a set of corresponding 'correct' level sets. If these are available it is simply a matter of taking the statistics (e.g. mean and standard deviation for a Gaussian approximation) of the mesh values for each feature. In general we have neither feature locations nor correct level sets, so these must be obtained. Features are obtained using a tracker bootstrapping process, in which mesh, occurrence and location priors for features are set to sensible (but very general) defaults by hand. The tracker (described in sections 3 and 4) is then run to obtain feature locations over a set of images/frames. The correct level sets are obtained from binary hand segmentations of example images/frames containing faces. To form a 'correct' level set mesh an initial level set is created with mesh values of -1 assigned to mesh values relating to segmented (face) pixels, and values of 1 assigned to mesh values relating to non-segmented (background) pixels. From this initial mesh a set of zero level set points, which lie on the edges between mesh points, can be found by linear interpolation in the standard way (see [13]). From these zero level set points each level set mesh point can be re-estimated as +/- the distance from the nearest zero level set point (in the standard way):

$$\psi_{x,y,final} = \min_{n=1}^{N} (dist([x,y],[F_{n,x},F_{n,y}])) \times sign\left(\psi_{x,y,initial}\right) \tag{12}$$

Where $\psi_{x,y}$ is the mesh value at x,y, and $F_{n,x},F_{n,y}$ is the location of feature n. Given the feature locations (w.r.t. the tracked centroid and estimated scale), and correct mesh values statistics (means and standard deviations) can be gathered to form location and mesh value priors. The relative frequency of occurrence of features could also be used to estimate the occurrence priors, however in the implemented system a uniform prior is used.

7 Results and Evaluation

Figure 4 illustrates the tracker in operation.

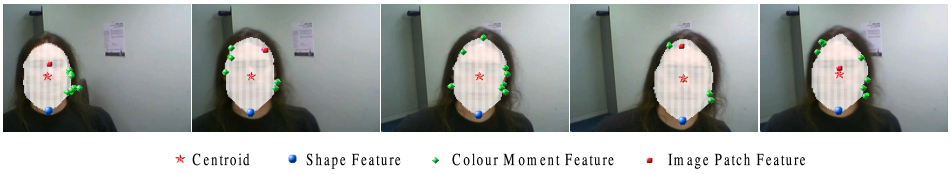

Figure 4: Tracked Sequence Showing Level Set and Features Located (5 frame / 0.5s intervals)

A qualitative evaluation of the tracker using live input (USB webcam) was carried out. 11 users were asked to use the system for 2-3 minutes with little instruction. Positional tracking was maintained for all subjects over this period (tracking at around 5-6 fps). The quality of the segmentation could not be measured objectively from this experiment, so movies were taken of two subjects at random and hand segmentations performed at 0.5s (5 frame) intervals for a number of seconds. The results of this quantitative evaluation are presented in figure 5.

Tracker vs. G.T. Area Overlap	G.T. vs. Tracker Area Overlap
0.96 [s.d. 0.03]	0.76 [s.d. 0.05]

Figure 5: Comparison of Tracker output to Ground Truth (G.T.)

Figure 5 illustrates that the ability of the tracker to segment the face is near perfect (illustrated by the 0.96 tracker to ground truth overlap). However, there is over-segmentation (illustrated by the 0.76 ground truth to tracker overlap). This equates to an average error of around 3 to 4 pixels at the perimeter at 320x240 image resolution. The small standard deviations of both metrics indicate this error is more systematic than random, which is encouraging from a systems point of view. This error is largely related to the different shapes of faces in the training and test data, which results in this systematic error.

8 Discussion and Future Work

The example on-line system implemented performs face location/segmentation/tracking at around 5-6 fps on a PIII 1GHz, which is sufficient to maintain on-line tracking (from a camera input) for extended periods of time. Segmentation performance is qualitatively good, with a small (but significant) systematic over-segmentation observed under quantitative analysis. This over segmentation is most prominent around the area of the lower jaw and chin, where no true features are located (segmentation relies on a single shape feature at the chin). Accuracy would be improved if a reliable feature detector was available for this area.

The use of priors for feature occurrence and location priors in the implemented system is fairly basic. These priors could be updated dynamically using current feature occurrence information (see for example the scheme in [9]) or using feature co-occurrence/co-location information (for example that learned in [1]). Configuration information from previous timesteps could also be used. This is a rich area for future research.

From a theoretical point of view the level set evolution presented is imperfect as there is no unique optimal mesh configuration for a given set of features, except for mesh values

calculated from zero level set points that are the closest zero level set point to a feature. The state of other mesh points is based on the dynamics of the system. In practise this is rarely a problem as feature locations change from one frame to the next, and the dynamics of nearby points are similar. However, surface roughness and imperfections can be observed on occasion due to this. A solution to this would be to introduce a standard curvature based speed function [13] or a probabilistic curvature prior [7] at the zero level set points, which would implicitly define a single optimal mesh configuration. This would have the added benefit of smoothing curvature, however would come at extra computational cost.

9 Acknowledgements

This work was funded by the European Union as part of the CogVis project (Contract IST-2000-29375).

References

[1] S. Agarwal and D. Roth. Learning a sparse representation for object detection. In *Proc. European Conference on Computer Vision*, volume 4, pages 113–127, 2002.

[2] A. Blake, R. Curwen, and A. Zisserman. A framework for spatiotemporal control in the tracking of visual contours. *International Journal of Computer Vision*, 11:127–145, 1993.

[3] T. Cootes, G. Edwards, and C. Taylor. Active appearance models. In *Proc. European Conference on Computer Vision*, volume 2, pages 484–498, 1998.

[4] A. Fitzgibbon. Robust registration of 2d and 3d point sets. In *Proc. British Machine Vision Conference*, pages 411–420, 2001.

[5] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29:5–28, 1998.

[6] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. In *Proc. First International Conference on Computer Vision*, pages 259–268, 1989.

[7] M. Leventon, O. Faugeras, W.E. Grimson, and W. Wells. Level set based segmentation with intensity and curvature priors. In *Proc. IEEE Workshop on Mathematical Methods in Biomedical Image Analysis*, pages 4–11, 2000.

[8] M. Leventon, W.E. Grimson, and O. Faugeras. Statistical shape influence in geodesic active contours. In *Proc. Computer Vision and Pattern Recognition*, pages 316–323, 2000.

[9] D. Magee. A sequential scheduling approach to combining multiple object classifiers using cross-entropy. In *Proc. International Workshop on Multiple Classifier Systems*, pages 135–145, 2003.

[10] R. Malladi, J. Sethian, and B.C. Vemuri. Shape modeling with front propagation: A level set approach. 17(2):158–175, Feb. 1995.

[11] S. McKenna, S. Gong, and Y. Raja. Modelling facial colour and identity with gaussian mixtures. *Pattern Recognition*, 31(12):1883–1892, 1998.

[12] N. Paragios and R. Deriche. Geodesic active contours and level sets for the detection and tracking of moving objects. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(3):266–280, 2000.

[13] J.A. Sethian. *Level Set Methods and Fast Marching Methods*. Cambridge University Press, 1999.

[14] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3:71–86, 1991.