in Proc. International Conference on Pattern Recognition, Vienna, 1996

Hermite Deformable Contours

D.M. Gavrila
Computer Vision Laboratory, CfAR,
University of Maryland
College Park, MD 20742, U.S.A.
gavrila@umiacs.umd.edu
http://www.umiacs.umd.edu/users/gavrila/

Abstract

We propose the Hermite representation for deformable contour finding. This representation compares favorably in terms of versatility and controlability with other local contour representations that have been used previously for this purpose. The Hermite representation allows a compact representation of curved shapes, without the smoothing out of corners. It is also well suited for both interactive and tracking applications.

The Hermite representation is used to formulate the contour finding problem as an optimization problem using a maximum a posteriori energy criterion. Optimization is performed by dynamic programming. Our approach to contour tracking decouples the effects of transformation and deformation, using a template matching strategy to robustly account for the transformation effect. We demonstrate these ideas on a variety of images from different domains.

1 Introduction

Image segmentation by boundary finding is one of the central problems in computer vision. This is because amongst features that can be used to distinguish objects from their backgrounds, such as color and texture, shape is usually the most powerful. For detecting instances of objects with fixed and known shape, the Hough-transform or a template matching technique is well suited (see [3]). For cases where there exists some flexibility in the object shape (either w.r.t. a previous frame in a tracking application, or w.r.t. a user supplied shape in an interactive object delineation setting) deformable contour models have found widespread use.

Deformable contours (also called active contour models, or "snakes") are energy-minimizing models for which the minima represent solutions to contour segmentation problems. They can overcome problems of traditional bottom-up segmentation methods, such as edge gaps and spurious edges, by the use of an energy function that contains shape information in addition to terms determined by image features. The additional shape information can be seen as a regularization term in the fitting process. Once placed in image space, the contour deforms to find the most salient contour in its neighborhood, under the influence of the generated potential field.

An extensive amount of work has been reported on deformable contours since their emergence in the late eighties; among others [1], [4]-[6], [8]-[12], [14]-[16]. A useful way to characterize the different approaches is along the following dimensions:

- contour representation
- energy formulation (internal and external)
- contour propagation mechanism (spatial and temporal)

We review the various contour representations that have been used in Section 2. A new local representation is proposed for the deformable contour framework, based on Hermite interpolating cubics, see Section 3. Its use has several advantages, as will become apparent. The main plus is that it handles both smooth and polygonal curves naturally.

We formulate the solution to the contour finding problem by a *maximum a posteriori* (MAP) criterion. This leads to an internal energy formulation which contains squared terms of deviations from the expected Hermite parameter values. The external energy terms describe the typical image gradient correlations. See Section 4.1. The resulting energy minimization is performed by dynamic programming which gives the optimal solution to contour finding for a certain search region, see Section 4.2.

One of the well-known limitations of deformable contours is that their initial placement has to be close to the desired object boundary in order to converge. In tracking applications, this assumption might be violated. To keep the problem computationally tractable, we propose to decouple the effects of transformation and deformation, see Section 4.3.

Experiments on a variety of images are presented in Section 5, after which we conclude in Section 6.

2 Contour Representations - A Review

Contour representations can be roughly divided into two classes, depending on whether they are *global* or *local*. Global representations are those where changes in one shape parameter affect the entire contour, and conversely, local change of the contour shape affects all parameters. These representations are typically compact, describing shape in terms of only a few parameters. This is an advantage in a recognition context, i.e. when trying to recover these parameters from images, because of lower complexity. A useful class of shapes easily modeled by a few global parameters are the super-quadrics [15], which are general-

izations of ellipses that include a degree of "squareness". To these shapes, one can add global deformations, such as tapering, twisting and bending [2]. A more general global representation is the Fourier representation [14]. It expresses a parametrized contour in terms of a number of orthonormal (sinusoidal) basis functions. Arbitrary contours can be represented in any detail desired, given a sufficient number of basis functions.

Local representations control shape locally by various parameters. This flexibility makes local representations well suited in a shape reconstruction context, as is the case when deforming a contour to fit image data. The simplest contour representation is an ordered list of data points. More compact representations describe contours in terms of piecewise polynomials. Each segment of the parametrized contour $(x_i(t), y_i(t))$ is described by a polynomial in t. The lowest-degree interpolating polynomial is of degree one, leading to a contour representation by polylines and polygons. More flexibility is possible by the use of higher order polynomials, generally cubic polynomials; they are the lowest degree polynomials for which derivatives at the endpoints can be specified. Higher order polynomials tend to bounce back and forth in less controlable fashion and therefore are used less frequently for interpolation purposes.

Natural cubic splines are piecewise third degree polynomials which interpolate control points with C^0, C^1 and C^2 continuity. The natural cubic spline parameters depend on all control points, which makes it a global representation. B-splines on the other hand, have a local representation, where contour segments depend only on a few neighboring control points. This comes at a price of not interpolating the control points. The same C^0, C^1 and C^2 continuity as natural splines is now achieved at the join points of connecting segments. By replicating control points, one can force the B-spline to interpolate the control points. A last interesting property is that the B-spline can be specified such that it performs a least-squares fit on the available data points.

In previous work, three local representations have been used for deformable contour finding: point representations, polygonal chains and uniform B-splines. These representations have the following disadvantages when used for the contour finding task.

Manipulating contours on the fine scale offered by pixel-by-pixel representations leads typically to high computational cost (for example, note the high complexity incurred in [8]). The incorporation of a-priori shape information in this featureless representation is difficult. If, on the other hand, a contour is represented by a few (feature) points, and contour finding only considers image data in the local neighborhood of these points, no use is made of data at intermediate locations which makes the approach prone to image noise.

The polygonal chain representation [6] overcomes some of these problems. However, it is not well suited to represent curved objects well, requiring many control points to be adequate. In an interactive object delineation setting, this is tedious. For tracking applications, the placement of control points close to each other, typical also of point representations, leads to stability problems. This is because for most contour finding approaches using local representations, a-priori shape information is encoded for each control point w.r.t. to its neighboring control points (i.e. curvature [9][12] [16], affine coordinates [10]). If control points are close together, small deviations due to image noise or contour propagation will result in large changes of local shape properties.

B-splines present an efficient and natural way to represent smoothly curved objects. For objects with sharp corners they are less suited; the C^2 continuity smooths out any regions of high curvature of a contour. The fact that B-splines do not interpolate the control points can be considered a drawback in an interactive object delineation setting (think of a physician pointing to specific locations in medical images). The before mentioned use of control point duplication can take care of this, but then straight line segments appear around the newly C^0 continuous control point. Without user intervention, when to duplicate control points becomes a difficult decision; for example, Menet [11] duplicates control points in regions where after M steps of contour deformation, the curvature is higher than a user-supplied threshold θ.

3 The Hermite Representation

The previous considerations lead us to propose the Hermite representation for deformable contour finding. Hermite contours are piecewise cubic polynomials, which interpolate the control points $\mathbf{p_0}, ..., \mathbf{p_N}$. In each interval, the Hermite cubic $Q(s,t) = [x(s,t)\ y(s,t)]$ is specified by the positions $\mathbf{p_{i-1}}$, $\mathbf{p_i}$ and tangent vectors $\tau^+_{\mathbf{i-1}}$, $\tau^-_{\mathbf{i}}$ at the endpoints.

Let $\mathbf{Q}$ be an arbitrary cubic polynomial

$$\mathbf{Q} = \mathbf{T} \cdot \mathbf{C} \tag{1}$$

where

$$\mathbf{T} = [t^3\ t^2\ t^1\ 1] \qquad \mathbf{C} = \begin{bmatrix} a_x & a_y \\ b_x & b_y \\ c_x & c_y \\ d_x & d_y \end{bmatrix}$$

with tangent vector $\mathbf{Q'(t)}$

$$\mathbf{Q'} = \mathbf{T'} \cdot \mathbf{C} = [3t^2\ 2t\ 1\ 0]\ \cdot\ \mathbf{C} \tag{2}$$

Given hermite parameter matrix

$$\mathbf{H_i} = [\mathbf{h_{i_x} h_{i_y}}] = [\mathbf{p_{i-1}}\ \mathbf{p_i}\ \tau^+_{\mathbf{i-1}}\ \tau^-_{\mathbf{i}}]^T \tag{3}$$

the corresponding Hermite coefficient matrix $\mathbf{C_{H_i}}$ can be derived as [7]

$$\mathbf{C_{H_i}} = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \mathbf{H_i}$$

We collect all the hermite parameters in state vector $\mathbf{H}$ for later use

$$\mathbf{H} = \begin{bmatrix} \tau_{\mathbf{0}}^{-} \\ \mathbf{p_0} \\ \tau_{\mathbf{0}}^{+} \\ \cdots \\ \tau_{\mathbf{N}}^{-} \\ \mathbf{p_N} \\ \tau_{\mathbf{N}}^{+} \end{bmatrix} \quad (4)$$

When considering the same criteria of usefulness for the contour finding problem as discussed in previous section for the point-, polygon- and spline-based representations, we note that the Hermite representation

- can efficiently represent both smooth and sharp contours. This is because smooth contours are well represented by the Hermite interpolating cubics, while at the same time, arbitrary sharp corners can be easily generated at the control points by the adjustment of the left and right tangent vector parameters
- interpolates the control points
- is explicit in those features that can be measured from image data: position and direction of gradient at control points. This allows to prune the search space during contour finding, as we will see in next section.

4 Contour Detection

4.1 MAP formulation

A *maximum a posteriori* (MAP) criterion is formulated for the solution of the contour finding problem. The aim is to find from all possible contours the contour which matches the image data best, in a probabilistic sense. Let $\mathbf{d}(x,y)$ be the original normalized image and $\mathbf{t_H}(x,y)$ be the image template corresponding to the Hermite parameters $\mathbf{H}$. We want to find $\mathbf{H_{MAP}}$ which maximizes the probability that $\mathbf{t_H}$ occurs given $\mathbf{d}$, e.g. $P(\mathbf{t_{H_{MAP}}}|\mathbf{d})$. $\mathbf{t_{H_{MAP}}}$ is then the *maximum a posteriori* solution to the problem. Bayes rule gives

$$\begin{aligned} P(\mathbf{t_{H_{MAP}}}|\mathbf{d}) &= \max_{\mathbf{H}} P(\mathbf{t_H}|\mathbf{d}) \\ &= \max_{\mathbf{H}} \frac{P(\mathbf{d}|\mathbf{t_H})\ P(\mathbf{t_H})}{P(\mathbf{d})} \end{aligned} \quad (5)$$

where $P(\mathbf{d}|\mathbf{t_H})$ is the conditional probability of the image given the template, and $P(\mathbf{t_H})$ and $P(\mathbf{d})$ are the prior probabilities for template and image, respectively. Taking the natural logarithm on both sides of eq.(5) and discounting $P(\mathbf{d})$, which does not depend on $\mathbf{H}$, leads to an equivalent problem of maximizing objective function U

$$\begin{aligned} U(\mathbf{t_{H_{MAP}}}, \mathbf{d}) &= \max_{\mathbf{H}} U(\mathbf{t_H}, \mathbf{d}) \\ &= \max_{\mathbf{H}} (ln P(\mathbf{t_H}) + ln P(\mathbf{d}|\mathbf{t_H})) \end{aligned} \quad (6)$$

The above equation describes the trade-off between a-priori and image-derived information.

If the image is considered as a noise corrupted template with a additive and independent noise that is zero-mean Gaussian, we have $P(\mathbf{d}|\mathbf{t_H}) = P(\mathbf{d}|\mathbf{t_H} + \mathbf{n}) = P(\mathbf{n}|\mathbf{d} - \mathbf{t_H})$, thus

$$P(\mathbf{d}|\mathbf{t_H}) = \prod_{\mathbf{t_H}(x,y)} \frac{1}{\sqrt{2\pi}\sigma_n} e^{-\frac{(d(x,y) - t_H(x,y))^2}{2\sigma_n^2}} \quad (7)$$

and

$$ln\ P(\mathbf{d}|\mathbf{t_H}) = constant + \sum_{\mathbf{t_H}(x,y)} \frac{(d(x,y) - t_H(x,y))^2}{2\sigma_i^2} \quad (8)$$

This last term can be replaced by correlation term $\mathbf{d} \cdot \mathbf{t_H}$, approximating $||\mathbf{d}||$ and $||\mathbf{t_H}||$ by constants. For $||\mathbf{d}|| = 1$ and $||\mathbf{t_H}|| = 1$ we obtain

$$\max_{\mathbf{H}} ln\ P(\mathbf{d}|\mathbf{t_H}) \approx \min_{\mathbf{H}} (1 - \mathbf{d} \cdot \mathbf{t_H}) = \min_{\mathbf{H}} E_{ext} \quad (9)$$

A similar derivation was described by Rosenfeld and Kak [13] and Staib and Duncan [14].

We model the prior probability for a Hermite contour $\mathbf{H}$ as

$$P(\mathbf{H}) = P(\mathbf{H}|\overline{\mathbf{H}}) = constant \cdot \prod_i e^{-\frac{(\mathbf{H}_{[i]} - \overline{\mathbf{H}}_{[i]})^2}{2\sigma_i^2}} \quad (10)$$

where $\overline{\mathbf{H}}$ represents an expected contour. $\overline{\mathbf{H}}$ is typically obtained as the sample mean of contours generated in a training phase, or as the contour obtained by prediction during tracking. σ_i acts as a weighing measure for the various dimensions. In case of an open contour, we set the σ's of τ_0^- and τ_N^+ equal to infinity.

In tracking applications the contour typically undergoes a tranformation T (for example, translation, rotation and scale) for which one does not want to penalize. The above modeling assumes that any transformation on the contour which one does not want to penalize has already been performed, before eq.(10) is applied. Any further contour change is considered as deformation from an expected contour and thus penalized.

Taking the natural logarithm gives

$$\max_{\mathbf{H}} ln P(\mathbf{t_H}) = \min_{\mathbf{H}} \sum_i \frac{(\mathbf{H}[i] - \overline{\mathbf{H}}[i])^2}{2\sigma_i^2} = \min_{\mathbf{H}} E_{int} \quad (11)$$

4.2 Dynamic Programming

There are many ways to solve the resulting minimization problem

$$\min_{\mathbf{H}} E = \min_{\mathbf{H}} (E_{int} + E_{ext}). \quad (12)$$

Variational calculus methods have been used extensively for continuous parameter spaces where derivative information is available [5] [9] [11] [12] [14] [15].

For discrete search spaces one possibility is to use A.I. search techniques (e.g. best-first, simulated annealing, genetic algorithms). We will use a discrete enumeration technique based on dynamic programming (DP) which was popularized by Amini *et al.* [1], and used since by [8] [10]. The advantages of dynamic programming w.r.t. variational calculus methods are in terms of stability, optimality and the possibility to enforce hard constraints [1]. For dynamic programming to be efficient compared to the exhaustive enumeration of the possible solutions, the decision process should be *Markovian*. This is typically the case if the a priori-shape component E_{int} contains a summation of terms which only depend on parameters which can be derived locally along the contour.

For the case of open contours, our objective function can be written as

$$\begin{aligned} E &= E_1(\mathbf{p_0}, \tau_{\mathbf{0}}^+, \tau_{\mathbf{1}}^-, \mathbf{p_1}) + \ldots + \\ & \quad E_N(\mathbf{p_{N-1}}, \tau_{\mathbf{N-1}}^+, \tau_{\mathbf{N}}^-, \mathbf{p_N}) \end{aligned} \tag{13}$$

Applying the dynamic programming technique to our formulation involves generating a sequence of functions of two variables, s_i with $i = 0..N-1$, where for each s_i a minimization is performed is over two dimensions. s_i are the optimal value functions defined by

$$\begin{aligned} s_0(\tau_{\mathbf{1}}^-, \mathbf{p_1}) &= \min_{\mathbf{p_0}, \tau_{\mathbf{0}}^+} E_1(\mathbf{p_0}, \tau_{\mathbf{0}}^+, \tau_{\mathbf{1}}^-, \mathbf{p_1}) \\ s_i(\tau_{\mathbf{i+1}}^-, \mathbf{p_{i+1}}) &= \min_{\mathbf{p_i}, \tau_{\mathbf{i}}^+} (s_{i-1}(\mathbf{p_i}, \tau_{\mathbf{i}}^+) + \\ & \quad E_i(\mathbf{p_i}, \tau_{\mathbf{i}}^+, \tau_{\mathbf{i+1}}^-, \mathbf{p_{i+1}})) \\ & \quad i = 1 \,..\, N-1 \end{aligned} \tag{14}$$

If $\mathbf{p_i}$ and $\tau_{\mathbf{i}}^-$ ($\tau_{\mathbf{i}}^+$) range over N_P and N_T values at each index i, the complexity of the proposed algorithm is $O(NN_P^2N_T^2)$.

The above formulation is for open contours. For closed contours, where the first and last control point are defined equal, we apply the same algorithm as for the open contour case, yet repeat it for all N_P possible locations of the first (last) control point, while keeping track of the best solution. The complexity increases to $O(NN_P^3N_T^2)$.

Speed-up can be achieved by a multi-scale approach. Here contour finding is first done on a lower resolution image to find an approximated contour. This can be done with a coarse discretization of the parameter space (i.e. requiring smaller N_P and N_T for the same parameter range). At the finer level, the originally desired discretization can be achieved by decreasing the parameter range to lie around the solution found at the coarse level.

At the same scale, the algorithm can be sped up by discounting unprobable control point locations before starting the DP search. A measure of "unprobability" can be specified in terms of weak image gradient strength or dot product between measured and expected gradient directions (the latter are explicit in the Hermite representation). If all the candidate control point locations are rated similarly (e.g. standard deviation of ratings below a threshold), it is more robust to consider all.

In addition, for closed contours, one can use only a single pass DP for closed contour and to optimize for the remaining $E_0(\mathbf{p_N}, \tau_{\mathbf{N}}^+, \tau_{\mathbf{0}}^-, \mathbf{p_0})$ while assigning to $\mathbf{p_0}$ and $\mathbf{p_N}$ the optimal values found for the open contour case. Of coarse, all these speed-up procedures loose the optimality property of DP. Nevertheless, the last two methods which were implemented performed satisfactory in practice.

4.3 Contour Tracking

The high computational cost of dynamic programming, and of other search methods which do not get stuck in the closest local minimum, makes search only feasible in a limited neighborhood. For interactive contour delineation this is fine, since the user is likely to place well-positioned control points, very close to the desired contour. In tracking applications this requirement is often unrealistic. On the other hand, it is our observation that the effects of deformation are often small from frame to frame once rigid motion is accounted for.

We therefore decouple the effects of motion and deformation on the contour, searching first for transformation parameters $T = [\mathbf{t}, \phi, s]$ with $\mathbf{t}$, ϕ and s denoting translation, rotation and scaling. T is found w.r.t. the undeformed contour, after which search continues for the deformation parameters. The first stage is robustly performed by template matching (or Hough Transform [10]) on a Gaussian-blurred gradient image. The second stage is the DP approach described earlier. Both stages use motion prediction methods; template matching at time $t+1$ searches in a parameter range centered around predicted transformation $\overline{T}(t+1)$ using predicted template $\overline{\mathbf{H}}(\mathbf{t+1})$. $\overline{\mathbf{H}}(\mathbf{t+1})$ is also the initial contour of DP search.

For simplicity, we currently use $\overline{T}(t+1) = T(t)$ and $\overline{\mathbf{H}}(t+1) = \mathbf{H}(\mathbf{t})$. More general, $\overline{T}(t+1) = p(t+1)$ where p is a best fitting n-th order polynomial at (t-M, T(t-M)) ..., (t-1, t). Similar consideration holds for $\overline{\mathbf{H}}(\mathbf{t+1})$. If the time-span M in which a n-th order model holds is large it is efficient to use a recursive predictor such as the Kalman filter.

5 Experiments

We have performed experiments with the proposed combination of Hermite representation and template-plus-DP search in both interactive as tracking settings. The associated template matching parameters were range and discretization of the transformation parameters (translation, rotation and scale). DP-related parameters included the initial values of the Hermite parameters, their range and discretization, as well as the weighing parameters. The locations considered around control point $\mathbf{p_i}$ lied on a rectangular grid with x-axis perpendicular to $\mathbf{p_{i+1} - p_{i-1}}$.

The Hermite gradients $\tau_{\mathbf{i}}$ were described in terms of length l_i and direction ϕ_i. Typically, $N = 5$, $N_P = 9$ ($N_P = 4$ after pruning), $N_l = 3$, $N_\phi = 9$.

Figure 1 demonstrates versatility of the Hermite representation. Different initial contours are placed by the user as shown in Figure 1a. Figure 1b shows the search region covered by DP for the initial control point placement; for each contour segment the Hermite cubics are shown corresponding to ($\phi_{i_{max}}$, $\phi_{i+1_{max}}$) and ($\phi_{i_{min}}$, $\phi_{i+1_{min}}$) for fixed (initial) control point locations and $l = l_{max}$. Many different Hermite contours which lie within this search region are not displayed. Figure 1c shows the result of contour finding by DP. One observes a wide variety of shapes that have been accurately described by the Hermite representations, from the smoothly varying contour of the mug rim to the sharp corners of the square pattern, with a curved horizontal segment joining at the corner. It compares favorably with a possible representation by polygonal chains, splines or Fourier descriptors.

For completeness, we also show in Figure 1d the conditioned Sobel gradient image, which is used by the DP algorithm. We use a conditioned image instead of the original Sobel image in order to amplify weak but probable edges. This is done based on local considerations, taking into account mean μ and standard deviation σ in a $n \times n$ neighborhood. A linear remapping is applied on the image data at (x, y) if σ is greater than a user specified threshold.

Figure 2 shows different instances of initial placement and contour detections on a MR image of the human brain. Finally, Figure 3 shows a tracking sequence of a head using the proposed combination of coarse-scale template matching and DP.

6 Conclusion

We have advocated the use of the Hermite representation for deformable contour finding. It was shown to have advantages over point-, polygonal- and spline-based representations in terms of versatility, stability and controlability. A decoupled approach to contour tracking was proposed based on template matching on a coarse scale to account for motion effects, and a DP formulation on a finer scale to account for the deformation effects.

7 Acknowledgements

The author thanks Larry Davis for his continued support.

References

[1] A.A. Amini, T.E. Weymouth and R. Jain, "Using Dynamic Programming for solving variational problems in vision," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* vol.12, no.9, pp.855-867, 1990.

[2] A. Barr, "Global and local deformations of solid primitives," *Comput. Graphics,* vol.18, pp.21-30, 1984.

[3] D.H. Ballard and C.M. Brown, *Computer Vision*, Prentice-Hall, Eaglewood Cliffs, 1982.

[4] A. Blake, R. Curwen and A. Zisserman, "A Framework for Spatiotemporal Control in the Tracking of Visual Contours," *Int. J. Computer Vision,* vol.11, nr.2, pp.127-145, 1993.

[5] A. Chakraborty, M. Worring and J.S. Duncan, "On Multi-Feature Integration for Deformable Boundary Finding," *Proc. ICCV,* pp.846-851, 1995.

[6] P. Delagnes, J. Benois and D. Barba, "Active contours approach to object tracking in images sequences with complex background," *Pattern Recognition Letters,* vol.16, pp.171-178, 1995.

[7] Foley *et al., Computer Graphics,* Addison-Wesley, 1996.

[8] D. Geiger et al., "Dynamic Programming for Detecting, Tracking, and Matching Deformable Contours," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* vol.17, no.3, 1995.

[9] M. Kass, A. Witkin and D. Terzopoulos, "Snakes: Active Contour Models," *Int. J. Comp. Vis.,* pp.321-331, 1988.

[10] K.F. Lai and R.T. Chin, "Deformable Contours: Modeling and Extraction," *Proc. IEEE Computer Vision and Pattern Recognition,* pp.601-608, 1994.

[11] S. Menet, P. Saint-Marc and G. Medioni, "B-snakes: implementation and application to stereo," *Proc. Image Understanding Workshop,* pp.720-726, 1990.

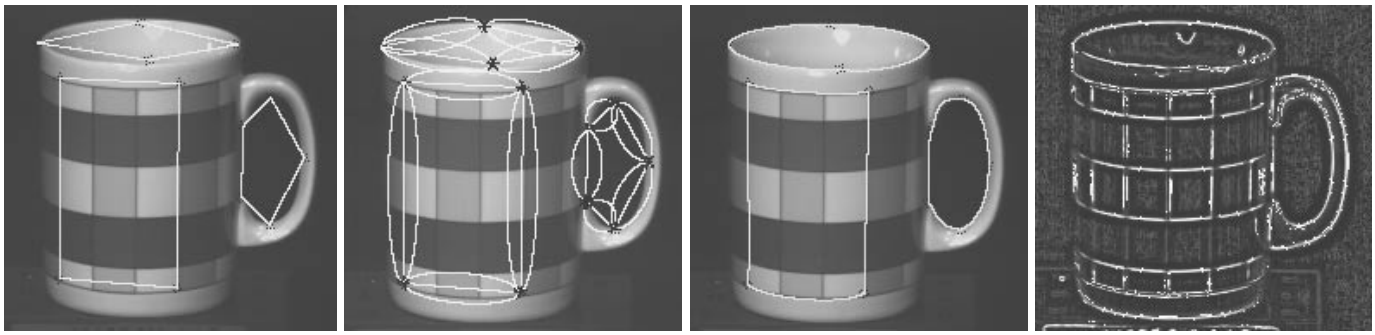
[12] P. Radeva, J. Serrat and E. Marti, "A Snake for Model-Based Segmentation," *Proc. ICCV,* pp.816-821, 1995.

[13] A. Rosenfeld and A. Kak: *Digital Picture Processing*, Academic Press, 1982.

[14] L.H. Staib and J.S. Duncan, "Boundary Finding with Parametrically Deformable Models," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* vol.14, no.11, pp.1061-1075, 1992.

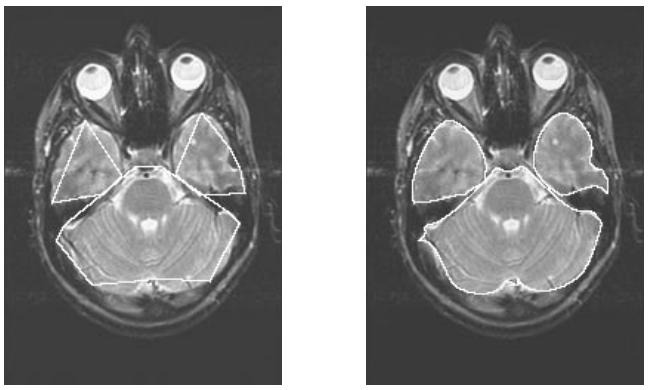
[15] D. Terzopoulos and D. Metaxas, "Dynamic 3D Models with Local and Global Deformations: Deformable Superquadrics," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* vol.13, no.7, pp.703-714, 1991.

[16] N. Ueda and K. Mase, "Tracking Moving Contours Using Energy-Minimizing Elastic Contour Models," *Proc. ECCV*, 1992.



(a) (b) (c) (d)

Figure 1: Mug image (a) contour initialization (b) search region (c) contour detection (d) conditioned Sobel image



(a) (b)

Figure 2: Brain MR image (a) contour initialization (b) contour detection

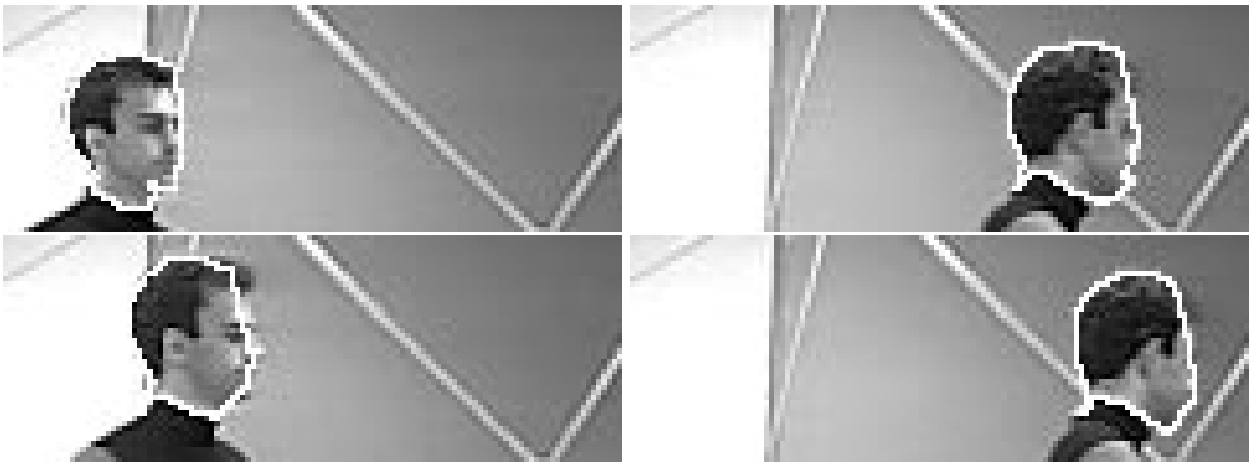


Figure 3: A tracking sequence of a head (t = 0, 8, 24, 30)