Tracking of Tubular Objects for Scientific Applications

B. Parvin, C. Peng and W. Johnston
Imaging Technologies Group
ICSD
Lawrence Berkeley Laboratory
Berkeley, CA 94720

M. Maestre
Biological Structures Group
LSD
Lawrence Berkeley Laboratory
Berkeley, CA 94720

Abstract

In this paper, we present a system for detection, tracking and representation of tubular objects in images. The uniqueness of the proposed system is twofold: at the macro level, the novelty of the system lies in the integration of object localization and tracking using geometric properties; at the micro level, in the use of high and low level constraints to model the detection and tracking subsystem. The underlying philosophy for object detection is to extract perceptually significant features from the pixel level image, and then use these high level cues to refine the precise boundaries. In the case of tubular objects, the perceptually significant features are anti-parallel line segments or, equivalently, their axis of symmetries. The axis of symmetry infers a coarse description of the object in terms of a bounding polygon. The polygon then provides the necessary boundary condition for the refinement process, which is based on dynamic programming. For tracking the object in a time sequence of images, the refined contour is then projected onto each consecutive frame. In addition, the system provides an axis of symmetry representation of object for subsequent scientific analysis.

1 Introduction

Object detection and tracking has been a major area of research in the computer vision community. Typical applications include target tracking, autonomous navigation, process control, and scientific data analysis. In this paper, we focus on a specific tracking problem for scientific applications. The main thrust in such scientific applications is either to gain new insights about the behavior of an object or to establish-and-verify certain hypotheses [15, 4]. We constrain our domain to objects that are elongated and maintain parallel symmetries. We refer to these objects as "tubular objects" that may or may not be terminated at both ends. In addition to man-made objects, such objects occur abundantly in nature: coronary arteries [13], bacteria, chromosomes [12] and macro structures in human lungs. In this paper, we illustrate the performance of our approach through an implementation that detects and tracks DNA molecules (a class of tubular objects) obtained through epi-fluorescence microscopy. An epi-fluorescence microscope is a typical scientific imaging device that operates at the extreme limits of the optical system and the photo conversion limits of the receptor system [15, 4, 3].

The main novelty of our system lies in its use of high level geometric features for object detection and the subsequent use of this information in the localization of individual molecules, followed by the re-use of this information for tracking. High level constructs are extracted from *incomplete* low level information, and then used to refine and *complete* the low level results. In the application domain, the high level constructs, or the *most perceptually significant* features, are ribbon-like structures defined by a collection of substructures called U-shapes and anti-parallel segments. These isolated segments are grouped with respect to the object model where a coarse description of the molecule is obtained in terms of a bounding polygon. This global representation is then refined using local pixel activities. The refinement is achieved through dynamic programming that ensures global optimality. The refined contour is then projected and updated in every consecutive frame. Furthermore, to facilitate subsequent quantitative analysis, the refined contour is represented by its axis of symmetry. This representation is then used to study the motion and mechanical properties of the molecules. The general architecture of this system is shown in figure 1, and consists of detection, tracking, and shape representation subsystems. In the next section, we briefly review the relevant work in the area, and point out to the similarities as well as unique aspects of our system with respect to previous work. Then, in sections 3, 4 and 5, we outline the details of each of the above subsystems and provide examples to demonstrate their performance on real data. Finally, we conclude our paper in section 6 by discussing the strengths and weaknesses of our approach.

2 Past work

Most of the previous papers in the vision literature aim at either detection *or* tracking of objects of interest in the scene. However, very little attention has been given to combining these techniques for real world applications. From this perspective, several papers have dealt with detection of ribbons [7], generalized cones [2], and more recently, generalized tube model [6]. A brief review follows.

Huertas, et al., [7] developed a system to detect runways in aerial images. They used a hypothesis-and-verification strategy to achieve their goal. The initial hypotheses are obtained by detecting peaks in the histogram of edge orientation images. The underlying idea is that such man-made objects will have a *global* orientation. *Local* hypothesis formation is initiated by detecting anti-parallel segments from low level edge information. These isolated anti-parallel segments, referred to as *apars*, are then joined based on their continuity, collinearity, and consistency with global orientation and certain knowledge about airport engineering. *Lowe* [11] developed a system called SCERPO, whose goal is to recognize and locate 3D rigid parts from a single

*Research was supported by the U.S. Dept. of Energy, Office of Scientific Computing, under Contract DE-AC03-76SF00098, and in part by NIH, under grant number AIO8427-25.

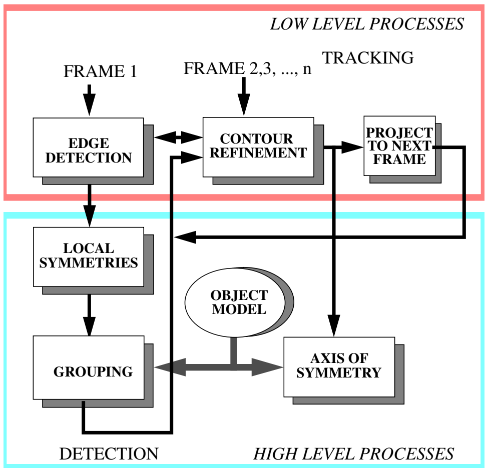


Figure 1: System architecture

gray-scale image. High level features such as instances of collinearity, end-point proximity, and parallelism are built based on straight lines formed by low level processes. These features are then combined into larger, more complex structures, in a stepwise fashion. Finally the structural patterns are used to limit the search space by hypothesizing the position of the part, which is verified by back projecting onto the edge data. Brooks [2] developed a system called ACRONYM, which has been influential in several vision systems. A typical application of this system is to identify airplanes in aerial images of airfields. In this system, objects are modeled as subpart hierarchies in terms of generalized cones (which are two dimensional shapes, and consist of a cross section swept across a spine). More recently, Huang and Stockman [6] introduced the generalized tube model, where the tube model is locally similar to GC. Their tube model combines contour- and shading-based techniques in a unique fashion. The contour information is used to hypothesize local tubes that are verified by matching their shading properties with dynamically generated optimal filters. The recognition step then sweeps across locally verified tubes using a best fit constraint.

In the context of tracking, most of the previous work is based on a variation of the *Snake* model [8]. This model assumes that an initial contour, defined interactively, is placed near the desired boundaries by means of a closed polygon or spline. This initial contour is then refined using a cost function that integrates internal and external energies, i.e.,

$$E_{snake} = \alpha E_{external} + \beta E_{internal} \tag{1}$$

where the external energy is measured by image forces such as gradients or zero crossings, and the internal energy is measured by the amount that the contour is being stretched or bent. Several refinement strategies based on relaxation [16], variational calculus [8, 9], and dynamic programming [1] have been proposed. The relaxation technique optimizes a local cost function, where the other two aim at global optimization of the cost function. There are some numerical anomalies in the variational approach that can be side-stepped by the other two techniques. In general, the Snake model assumes that the initial contour need not be placed near the desired boundaries[1] and the model is capable of deforming its initial contour into a state where the cost function is optimized.

Our system operates on the 2D flat world, and in some ways has been influenced by the work of Brook [2] and Huertas [7]. Its similarity to ACRONYM is in the use of an object model to drive the search process. And its commonality with Huertas's work is in the use of high level features for grouping. Still, our formulation for the particular application is unique in terms of the consistent labeling of the high level features. The detection phase of our system provides a coarse approximation, in terms of a bounding polygon, for each molecule. The fact that this description is very close to the actual boundary of the molecule, and that the motion of each molecule between consecutive frames is only several pixels, has enabled us to design a simple tracking mechanism that significantly limits the scope of search during refinment. Furthermore, our cost function is simpler than traditional Snake model; it does not rely on the rectangular tessellation of the pixel layout to compute the corresponding energy function, and it eliminates the scaling parameters for the internal and external forces.

3 Detection subsystem

In this section we summarize different components of the detection process as shown in figure 1, and we present intermediate results for each important step of the process. The edge detection is based on Canny's approach [5], which is inherently a gradient operator. The resulting edges are linked, curve segments are extracted, and polygon representation of these curve segments are obtained. The edge gradient and directions are computed at a coarse and fine scale. The coarse scale information is used in the detection phase for better noise immunity, and the fine scale information is used in the tracking phase for better localization. In general, due to noise and variation in contrast, the edge detection technique produces broken and undesirable curve segments. The objective is to group these curve segments such that the individual objects can be extracted from background. This is accomplished by representing curve segments by richer representations which are referred to as "ribbons" in the computer vision literature. A ribbon is a polygon formed by a set of anti-parallel segments or U-shapes. In Brook's definition [2], a ribbon is a two dimensional analog of a generalized cone. These local ribbons correspond to local symmetries that provide the foundation for grouping them into a larger structure based on a suitable object model. The ribbon representation is difficult to compute, but it simplifies the process of high level reasoning.

3.1 Local symmetries

Local symmetries are identified either from U-shapes or antiparallel segments. U-shapes correspond to the ends of a molecule or points of maximum curvature, and antiparallel segments correspond to the main body of the molecule.

A U-shape is a collection of connected line segments whose total accumulation of the angles between adjoining segments is approximately 180 degrees. There are several ways to extract U-shapes from polygon approximations of curve segments. One way of doing this is by summing the angles θ_i between adjacent line segments l_i and l_j at multiple scales, where scale refers to the number of line segments that are used in the computational process. The alternative approach is to search for line segments with

[1] In the sense that the scope of the search may include the entire scene.

opposite directions in the same curve segment. Both of these approaches can generate multiple U-shapes. This is resolved by imposing the constraint that a U-shape should be formed with minimum number of consecutive line segments.

An anti-parallel line is a pair of parallel line segments with opposite directions that can be projected onto one another. This construct is sometime referred to as an "apars" in the literature [7]. There are two types of anti-parallel lines in our system. These are the anti-parallel lines formed by line segments that belong to the *same* curve segment, and the anti-parallel lines that are formed by the line segments from *different* curve segments. The reason for this partitioning is that anti-parallels formed by the same curve segment are perceptually more significant than those formed from different curve segments. This is due to the fact that the anti-parallels formed between two different curve segments may not correspond to a single object. If the anti-parallel segments occur between different pieces of the same curve segments then the search process (for anti-parallel) is initiated from the position of the U-shape. On the other hand, if an anti-parallel segment occurs between different curve segments, then they can only be discovered by scanning normal to a line segment and forming a correspondence histogram.

3.2 Grouping

We express the grouping problem as a function of three entities:

DNA = Group(features, object model, geometric constraints)

In this formulation, *features* correspond to nodes with labels, and some attributes where a label can be either a U-shape or anti-parallel, and the attributes that encode its invariant geometric measures. The *object model* represents the relationship between the nodes of an idealized DNA model as described by U-shapes and anti-parallels. The *geometric constraints* encode the rules by which the nodes can be linked together. The goal of the grouping is to connect high level features, represented as nodes of a disconnected graph, in such a way that labeling of these nodes remains consistent with respect to the object model subject to the geometric constraints. In this approach, the grouping function is self-correcting since a single node might be mislabeled due to various perturbations in the lower level processes, and errors can be corrected as the grouping process continues. We now provide a detailed description of the object model, geometric constraints, and the grouping strategy to link a disconnected graph in a coherent fashion.

3.2.1 DNA object model

The object model is based on our a priori knowledge about the occurrence of objects in an image that may include shape, texture and context. A DNA molecule is distinguished by its symmetric-elongated shape with the following features:

1. Every DNA has two U-shapes, one at each end;
2. There are anti-parallel lines between these two U-shapes;
3. The variation in the width of a DNA is small; and
4. If we represent the U-shapes and anti-parallels as nodes, and the relationships between them as a link, then the links must be consistent. We refer to this as "Link Consistency" that will be explained later in more detail.

We represent the relationship between the nodes of a DNA object as a graph as shown in figure 2. As we pointed out, the significant characteristic of DNA molecules is their symmetric shape, and these symmetries can be localized by detection of *ribbon*-like structures that are computed through incremental coercion of the high level representation.

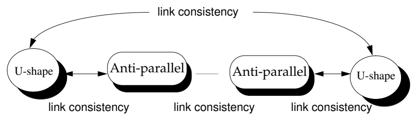


Figure 2: DNA Model

3.2.2 Geometric constraints

The goal of the grouping process is to connect high level features in such a way that labeling of these features remains consistent with the geometric constraints of the object model. These geometric constraints can be described as:

1. *Multiple-overlapping ribbons cannot be formed by the same line segments:*
   This constraint is implied by the DNA model, and is used to filter out irrelevant high level features and reduce the search space.
2. *Ribbons share a common curve segment:*
   Ribbons that are formed by the same curve segment are likely to belong to the same object, and hence, they can be included in the same group.
3. *Ribbons are collinear with each other;* and
4. *Ribbons have uniform width.*

These geometric constraints, together with the object model and high level features, establish the basis for the grouping process.

3.2.3 Geometric grouping

This step of the computational process attempts to assign a tag (unique ID) to each DNA object. This is accomplished by utilizing the geometric constraints in a stepwise fashion. However, the labeling consistency constraint defined by the model may not be guaranteed in this process. Hence, a set of verification steps are required to ensure the labeling consistency.

1. *U-shape consistency verification:*
   Every DNA has only two U-shapes and these are located at its ends. If a set of curve segments form more than two U-shapes, then some of these U-shapes must be erroneous. We distinguish these false U-shapes by a width property that is inconsistent with the width of other U-shapes.
2. *Linking consistency checking:*
   The two U-shapes must be connected by anti-parallel lines. We call this the "link consistency constraint" which is also used to remove false U-shapes.

Figure 3 shows various steps in the grouping process, and a list of illegal junctions is shown in figure 4.

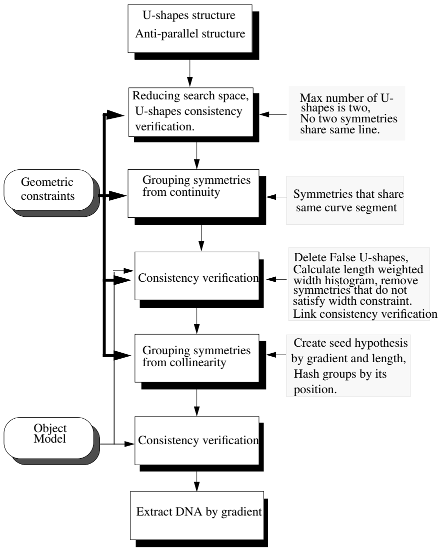


Figure 3: Grouping Process

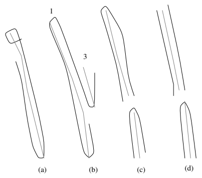


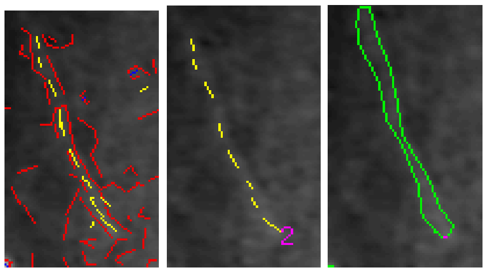
Figure 4: Link inconsistency: (a) T junction; (b) Concave boundary between 1 & 3; (c) Two co-directional U-shapes; (d) An inconsistent U-shape and anti-parallel.

3.3 Examples

We now illustrate the performance of our system with some examples. In these examples, the first image corresponds to edges and local symmetries; the second image corresponds to global symmetries; and the third image represents the inferred polygons. Figure 5 shows a single molecule with fragmented contours, fragmented symmetries, inconsistent symmetry and a number of multiple overlapping symmetries. The second example in figure 6, with the hot spot in the lower part of the image, shows several linear molecules and a curved one. Note that for the curved molecule, there is no local symmetry where the molecule bends. In this case, a local symmetry is inferred to maintain continuity. The last example, figure 7, is the case of a low contrast image with broken edge contours. Again, through the use of high level symmetries, we have extracted the object from the background.

4 Tracking subsystem

Extraction of *closed* contours that correspond to real object boundaries has been a major area of research in low level vision. As we have already pointed out, this is a difficult and complex task in the absence of any high level constraints. In some cases, high level constraints can be entered interactively. But, regardless of whether these high level constraints are incorporated interactively or computed automatically, some form of *refinement* is needed. We have formulated the refinement process as a cost function that encodes both the low level and high level constraints in a unique fashion. The actual optimization of the cost function is achieved via dynamic programming, which ensures global optimality in the search process. In this context, the system allows a coarse description, expressed by a bounding polygon, to snap back into the original contour and construct a refined representation. The refined contour is then convolved with a Gaussian kernel and projected upon each subsequent frame so that the tracking is advanced.



(a) (b) (c)

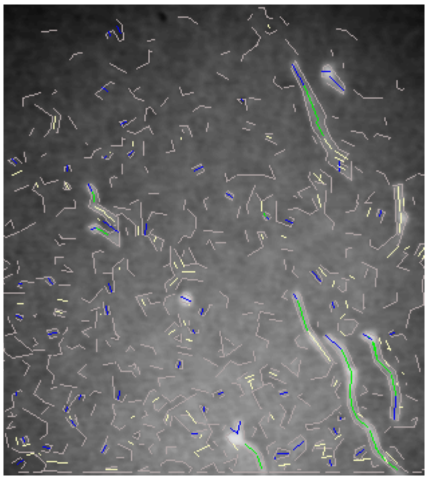
Figure 5: (a) Edges and local symmetries for a single molecule; (b) Global symmetry; (c) Inferred polygon.

4.1 Cost function

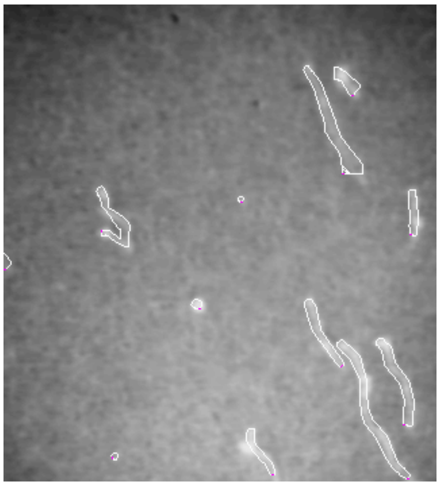
The cost function should encode the desirable properties of the refined contour. In our formulation, this is defined in terms of high and low level feature activities. The low level features refer to pixel level information such as local edge magnitude and direction. The high level features refer to the measurements that can be made from the bounding polygon. In our system, the high level constraints (bounding polygon) affect the contour refinement in two ways. The first one is a geometric constraint, which we will discuss later, and the second one is by limiting the scope of the search. This is accomplished by smoothing the initial polygon with a Gaussian kernel and bounding the refined contour to lie in a small neighborhood as defined by the normal lines to the smooth curve. The relationships between the smooth bounding polygon, normal lines, and the refined contours are shown in figure 8.

Conceptually, each point on the refined contour should have high gradient and good continuity with both the low and high level features. This is reflected in the cost function in terms of making a decision for a particular path. The cost function indicates the accumulation of making a decision to link two points from two consecutive normal line segments as shown in figure 9. Let:

1. g_i be the gradient magnitude for a point p_i on line L_i,
2. θ_i be the direction of the gradient at location p_i,
3. $\Delta\theta_{ij}$ be the difference between the direction of gradient for points p_i and p_j,

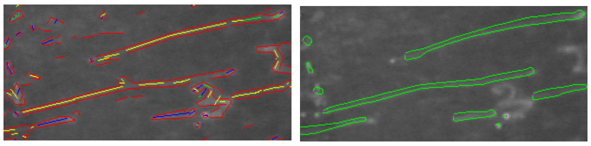


(a)



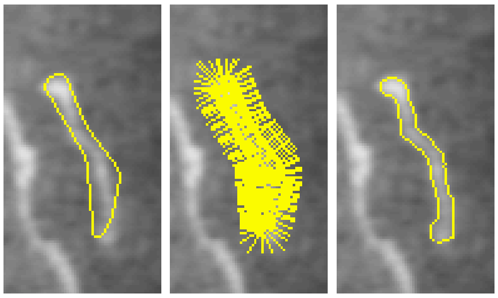
(b)

Figure 6: (a) Edges and local symmetries for multiple molecules; (b) Inferred polygons.



(a) (b)

Figure 7: (a) Edges and local symmetries for multiple molecules; (b) Inferred polygons.



(a) (b) (c)

Figure 8: Steps in refinement for a single molecule: (a) Gaussian-smooth bounding polygon from the detection step; (b) Normal lines to the smooth polygon; (c) Refined polygon.

4. γ_i be the angular difference between local edge direction and local direction of high level constraint,
5. η_{ij} be the angle between the gradient direction at point p_i and the vector connecting point p_j to p_i,
6. w_{ij} be the distance between points p_i and p_j located on two consecutive normal line segments, and
7. α_{ij} be the directional deviation between two consecutive points defined as $MAX\ (\Delta\theta_{ij}, \eta_{ij})$.

We formulate the cost function in such a way that the gradient is maximum, directional differences are small and the deviation for a particular path from the bounding polygon (high level constraint) is also small. One way to express such a cost function is as follows:

$$MAX_{i,j,k} \sum_{k} \sum_{i} \sum_{j} \frac{1}{w_{ij}} g_i e^{-tan(\frac{\gamma_i}{s})} \cos\left(\alpha_{ij}\right) \qquad (2)$$

Where k is the length of the contour; i and j are the pixel location on normal line to the smooth polygon. This formulation allows the curve to expand and shrink as necessary. Furthermore, by expressing the cost function as a product of internal and external energies, as opposed to the sum in equation (1), we have eliminated scaling coefficients that are used in the Snake formulation. The above cost function is optimized with the dynamic programming principle.

4.2 Examples

Figure 10 shows an example of the refinement process through four frames of a sequence. These frames are roughly 120ms apart in time. These molecules are being deformed as they are perturbed by an electric field. Note that some of the molecules are very near each other, and the tracking subsystem continues to track them without any interference. The major assumption in our formulation is that each normal line should intersect the DNA contour at least at one point. This will not be an issue as long as the motion between consecutive frames remains small.

5 Shape Representation

Our system aims to detect and track DNA over a sequence of images. We are interested in representing the shape of

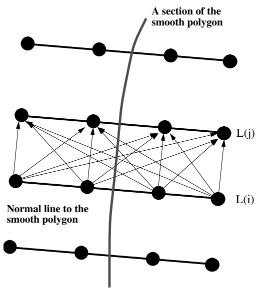
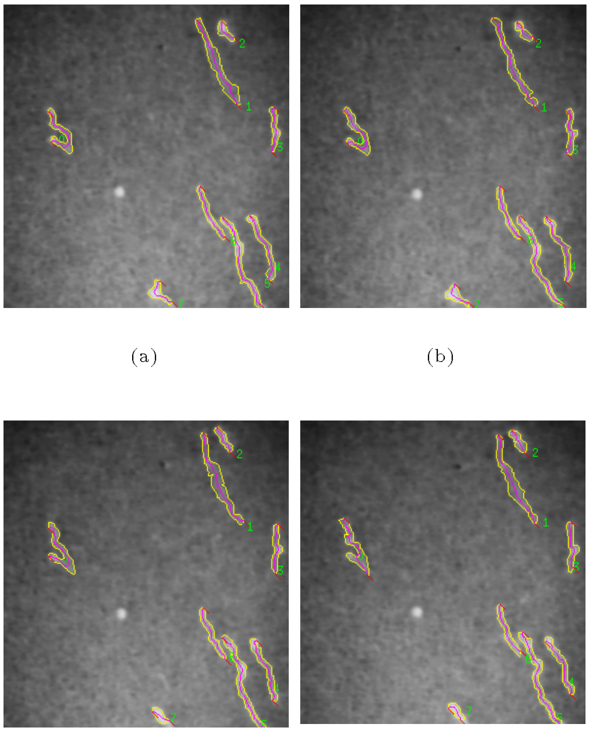


Figure 9: Possible paths for each point on a normal line segment



(a) (b)

(c) (d)

Figure 10: Four frames of the tracking sequence

DNA in such a way that the significant shape measures are captured. This is another step in the data reduction process that is used to visualize the temporal motion of the object. The simplest method to represent the shape of a DNA molecule is by its medial axis, which is sometimes referred to as "thinning" [14]. Yet, most thinning techniques ignore high level information and fail to generate desired results. Here, we propose a simple technique that is adequate for our application domain.

Our approach to thinning is to use the high level information about the shape of DNA to guide extraction of medial axes. This is accomplished by utilizing the end points of a DNA to partition the corresponding contour into two halves where the medial axis can be computed

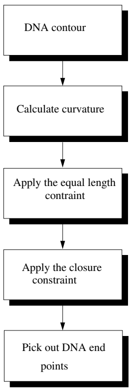


Figure 11: The process of finding the end points of a molecule

directly. A brief description for localizing the end-points of a molecule during tracking is followed.

5.1 Locating the end points

From a geometric point of view, under ideal conditions, the end points of DNA correspond to the location of maximum curvature along the contour. Yet, contour information is generally noisy, as a result the positions of maximum curvatures do not necessarily correspond to the end points of DNA. In other word, curvature information by itself is not a sufficient constraint. To resolve this problem, we add *equal length* and *closure* constraints as two additional constraints. Each pair of hypotheses, corresponding to the curvature peaks, divides the length of the contour into two halves, where equal length and closure constraints are applied.

Each of these steps, i.e., computation of curvature and application of various constraints, are designed to reduce the number of hypotheses in a stepwise fashion. The whole process is depicted in figure 11, and discussed below:

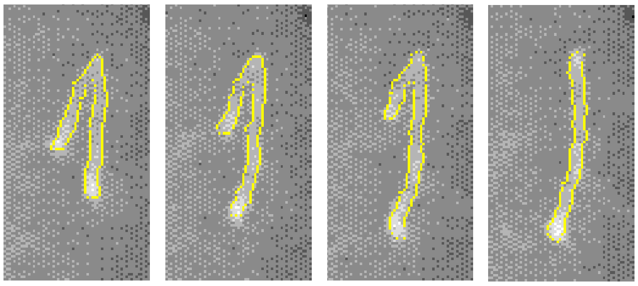
1. *Apply the curvature constraint*
   The curvature is computed along the length of the contour. Peaks are identified and ranked. Only the most prominent peaks are propagated into the next step of the process. The curvature along the contour is computed as follow: Let the contour be represented parametrically as $(x(t), y(t))$ and let $g(t)$ be a Gaussian kernel. The curvature is then defined as:
   $$c(t) = \frac{x'y'' - y'x''}{\sqrt{(x'^2 + y'^2)^3}} \tag{3}$$
   where
   $$x' = x(t) \bigotimes \frac{\partial g}{\partial t} \quad x'' = x(t) \bigotimes \frac{\partial^2 g}{\partial^2 t}$$
   $$y' = y(t) \bigotimes \frac{\partial g}{\partial t} \quad y'' = y(t) \bigotimes \frac{\partial^2 g}{\partial^2 t} \tag{4}$$
   Some of the curvature peaks can be eliminated through the use of Curvature-Duality theorem [10]. In this context, we can check whether the axis of symmetry is inside the polygon defined by the closed contours.

2. *Apply the equal length constraint*
   Every pair of curvature peaks should approximately divide the contour into equal lengths. This is accomplished by setting a soft threshold to accept curvature peaks for further analysis.

3. *Apply the closure constraint*
From a computational perspective, this is the most expensive constraint, and therefore is the final step of the computational process. This step ensures that the lines connecting the correspondence points, between each half curve, is inside the DNA object. The correspondence points are the points equidistant from the axis of symmetry. With this constraint, we can select a pair of points that correspond to end points of the DNA.

5.2 Examples

An example of applying the above constraints to locate end points of a DNA object, and extracting the axis of symmetry was shown in the previous section in figure 10. Here, we show the motion of axis of symmetry for a single molecule over 140 frames of data. Four frames of this sequence are shown in figure 12. Note that the curved molecule stretches, and eventually becomes linear under electric field forces. This is graphically shown in figure 13 for better visualization of the molecular behavior as a function of time.



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

Figure 12: Changes in the shape of a molecule over 140 frames

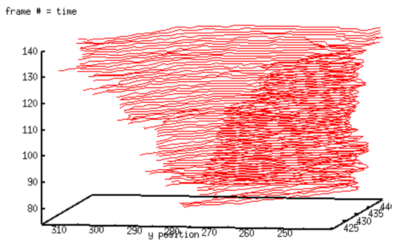


Figure 13: Changes in the shape (axis of symmetry) of a molecule as a function of time

(a)

6 Conclusion

In this paper, we have illustrated the performance of our system in detection, tracking, and shape representation of tubular objects. One final result was the dynamic analysis of DNA molecules being stretched by an electric field, which demonstrates that they behave much like a spring or a helix under excitation. This insight can eventually enable the scientists to manipulate a molecule in a gel under forces of an applied electric field.

We now summarize the limitations of our system. The detection subsystem assumes that local symmetries are correctly computed. In this sense the system is unable to probe the image further and infer additional missing local symmetries. In the tracking subsystem, the system assumes that the normal lines to the smooth polygon intersect the actual boundary of the molecule. If the object deforms rapidly, then the tracking subsystem will not compute the correct boundary along the length of the molecule. Finally, shape representation is based on finding the end points of a molecule, where the end points are hypothesized by the curvature peaks along the contour. These peaks may not be locally accurate due to noise and convolution of the contour with a Gaussian kernel (scale dependent localization error). The problem is further complicated when the molecule folds and the end point becomes almost smooth. (Folded molecules are beyond the current scope of our research.)

Acknowledgement: The authors thank B. Crowley, M. Gelbaum, and N. Ostrofe for the preparation of this manuscript.

References

[1] A. Amini, S. Tehrani, and T. Weymouth. Using dynamic programming for minimizing the energy of active contours in the presence of hard constraints. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 95–99, Tampa, Florida, 1988.

[2] R. A. Brooks. Model-based three-dimensional interpretations of two-dimensional images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2):140–150, 1983.

[3] C. Bustamante. Direct observation and manipulation of single dna molecules using fluorescence microscopy. *Annual Review of Biophysics and Biophysical Chemistry*, 20:415–446, 1991.

[4] C. Bustamante, T.W. Houseal, D. Beach, and F. M. Maestre. Fluorescence microscopy of the dynamics of supercoiling, folding, and condensation of bacterial chromosomes, induced by acridine orange. *Journal of Biomolecular structure and Dynamics*, 8:643–655, 1990.

[5] J. F. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, November 1986.

[6] Q. Huang and C. Stockman. Generalized tube model: Recognizing 3d elongated objects from 2d intensity images. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 104–109, New York, June 1993.

[7] A. Huertas, W. Cole, and R. Nevatia. Detecting runways in complex airport scenes. *Computer Vision, Graphics, and Image Processing*, 51(2):107–145, August 1990.

[8] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 259–268, London, England, June 1987.

[9] F. Leymarie and M.D. Levin. Tracking deformable objects in the plane using an active contour model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15:617–635, 1993.

[10] M. Leyton. Symmetry-curvature duality. *Computer Vision, Graphics, and Image Processing*, 38:327–341, 1987.

[11] D. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31:355–395, 1987.

[12] M.A. Nederlof, A. Witkins, and D.L. Taylor. Knowledge-driven image analysis of cell structures. In *Society of Photo-Optical Instrumentation Engineers*, volume 89, pages 1388–1392, June 1991.

[13] S. Rake. Three-dimensional reconstruction of arteries from biplane angiograms. *Medical Informatic*, 16:195–212, 1991.

[14] A. Rosenfeld. Axial representations of shape. *Computer Vision, Graphics, and Image Processing*, 2(33):156–173, 1986.

[15] L. Song and M. Maestre. Unhooking dynamics of u-shaped dna molecule undergoing gel electrophoresis. *Journal of Biomolecular structure and Dynamics*, 9:87–99, 1991.

[16] D. J. Williams and M. Shah. A fast algorithm for active contours and curvature estimation. *Computer Vision, Graphics, and Image Processing*, 55(1):14–26, 1992.