Motion Trajectory Matching of Video Objects

William Chen and Shih-Fu Chang

Dept. of Electrical Engineering, Columbia University
New York, New York 10027
E-mail: {bchen, sfchang}@ctr.columbia.edu

ABSTRACT

In this paper, we propose an efficient wavelet-based approach to achieve flexible and robust motion trajectory matching of video objects. By using the wavelet transform, our algorithm decomposes the raw object trajectory into components at different scales. We use the coarsest scale components to approximate the global motion information and the finer scale components to partition the global motion into subtrajectories. Each subtrajectory is then modeled by a set of spatial and temporal translation invariant attributes. Motion retrieval based on subtrajectory modeling has been tested and compared against other global trajectory matching schemes to show the advantages of our approach in achieving spatio-temporal invariance properties.

Keywords: Content-based video retrieval, motion representation, object trajectory, video database, wavelet descriptor.

1. INTRODUCTION

Video databases require indexing and search mechanisms that can capture its dynamic content. We have demonstrated in VideoQ [1] that motion characterization is an effective attribute in searching video databases. In VideoQ, video objects are segmented and tracked over the duration of a video shot. During the tracking process, global motion compensation algorithms are applied so that the final object trajectory is independent of camera motion. We store the raw object trajectory as a sequence of points in the database.

Prior techniques in motion trajectory modeling used simple chain-codes or B-splines. These do not completely capture the motion trajectory characteristics. Instead, we have identified two major modes of an object trajectory: spatio-temporal and spatial. The spatio-temporal mode can be mathematically represented as a parametric curve (x(t),y(t)) in the two-dimensional image plane or equivalently as a curve (x,y,t) in three-dimensions. As shown in Figure 1(a), the spatio-temporal mode captures the movement and speed of the object as it twists and turns in time. The spatial mode can be mathematically represented as a one-dimensional function y = f(x). As shown in Figure 1(b), the spatial mode is a projection of the spatio-temporal trajectory onto the image plane. It describes the shape of the object trajectory. In this paper, we model the spatio-temporal mode of the object trajectory. We note that both modes represent the object's affine motion which differs from the object's true motion in three-dimensional space (x,y,z).[1]

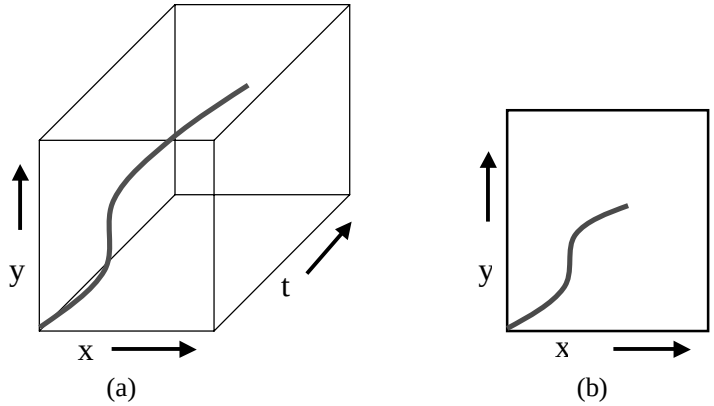


Figure 1. Two major modes of an object trajectory: (a) Spatio-temporal and (b) Spatial.

[1] If global camera motion estimation is accurate, the x and y coordinates can be approximated, and only the depth (z coordinate) information is lost.

In VideoQ, the user formulates a query based on a visual sketch. The user composes a visual sketch by adding video regions to a canvas and assigning features to each region, such as color, texture, shape, and motion. The user assigns motion by adding an arbitrary polygonal trajectory to the video region. As shown in Figure 2, most users assign simple trajectories, such as straight lines and parabolic curves with constant speed, but the trajectories stored in the database are often complex. The complexity arises from two sources:

1) Object trajectories are extracted from an automatic tracking algorithm. The algorithm tracks motion from frame to frame based on low-level optical flow. Given the optical flow information, a linear regression algorithm estimates the affine motion of the centroid of each video object. In addition to this, the algorithm also compensates for camera motion by adjusting each point in the trajectory by the motion of the background.
2) The video objects in the real world undergo complex movements. While the user might describe the action of a high-jumper as parabolic, the system records all the actions leading up to the jump, such as the accelerated running action followed by the parabolic jumping action.

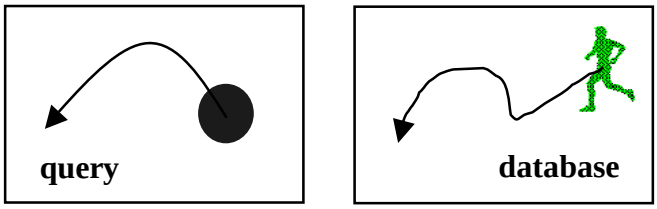


Figure 2. Object trajectory of a high jumper as sketched in a query and stored in the database.

We attempt to bridge the differences between the user's motion representation with those stored in the database. We have developed an efficient wavelet-based approach to achieve flexible and robust motion trajectory matching of video objects. We summarize our approach as follows:

- **Smoothing** Since the tracking and motion compensation are performed automatically, object trajectories are often imperfect and noisy. We use a wavelet-based approach to smooth the trajectories, using the coefficients from the coarse bands to reconstruct the global motion.
- **Segmentation** Video objects can undergo complex movements. We segment the object trajectory into subtrajectories with constant acceleration. We use the detailed wavelet coefficients to estimate the object acceleration at various resolutions.
- **Modeling** We model each subtrajectory as a feature vector of acceleration, velocity (including both magnitude and direction), arclength, order, and multiscale edge points, and support similarity matching based on this feature vector.

2. RELATED WORK

A number of researchers have proposed using motion to characterize video databases. We summarize some of their key contributions to object trajectory modeling.

In Dimitrova [2], motion trajectories are obtained by tracing the position of a macroblock. Several trajectory representation schemes are considered:

- Raw trajectory representation: The raw trajectory is a set of points tracing the relative or absolute position of the object in the image plane.
- Curve representation: The curve representation fits the trajectory with a B-spline curve and uses the parameters of the B-spline curve to represent the trajectory.
- Chain code representation: Chain code approximates the trajectory with piecewise linear segments and represents each segment with a directional primitive. The directional primitives quantize the space into eight basic directions and encode the eight basic directions from 0-7.
- Differential chain code representation: Differential chain code approximates the trajectory with piecewise linear segments and represents each segment with a directional primitive that is relative to the previous segment. The directional primitives include right and left and segment length.

In Dagtas et al [3], an explicit formalism is developed to describe many of the spatial and temporal properties of the object trajectory. The spatial and temporal properties include:

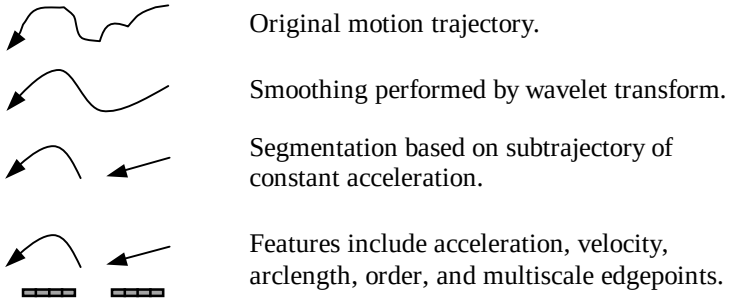
- Spatial scale invariance: The spatial scale refers to the arc length of an object trajectory in the image plane. Spatial scale invariance is achieved by normalizing the object trajectory by its arc length
- Spatial translation invariance: The spatial translation refers to the starting position of an object trajectory in the image plane. Spatial translation invariance is achieved by shifting the starting position of the trajectory to the origin.
- Spatial rotation invariance: The spatial rotation refers to the orientation of an object trajectory in the image plane. Spatial rotation invariance is achieved by aligning the major axis of a trajectory to the x-axis.
- Temporal scale invariance: The temporal scale refers to the temporal duration of an object trajectory. Temporal scale invariance is achieved by interpolating each motion trajectory to the same number of points.
- Temporal translation invariance: The temporal translation refers to the starting time of an object trajectory (as an object can enter a scene at different times in a video). Temporal translation invariance is achieved by segmenting a trajectory into subtrajectories and matching subtrajectories.

In Jasinschi et al [4], motion descriptors for camera motion and object trajectories have been recommended to the MPEG-7 standardization effort. Motion is described as an essential feature in characterizing animated data. In particular, the authors propose to model an object trajectory from a physics-based point of view. Descriptors include initial position, initial velocity, and acceleration.

3. OBJECT TRAJECTORY MODEL

As shown in Figure 3, we propose an algorithm that will segment the complex object trajectories into simple subtrajectories and then extract features from each subtrajectory to support similarity matching. We achieve all three goals with a single wavelet-based approach. In the next sections, we describe the algorithm in detail:

Figure 3. Overview of the object trajectory model.



3.1 Object Trajectory Smoothing

The first stage in the algorithm is to construct a multiresolution representation of the object trajectory. We separate the motion trajectory into two one-dimensional signals (x(t),y(t)) and apply a wavelet transform to each one. The wavelet transform of a signal decomposes it into a hierarchy of coefficients taken at different resolutions. We use the coarsest coefficients to approximate a smoothed trajectory as shown in Figure 4.

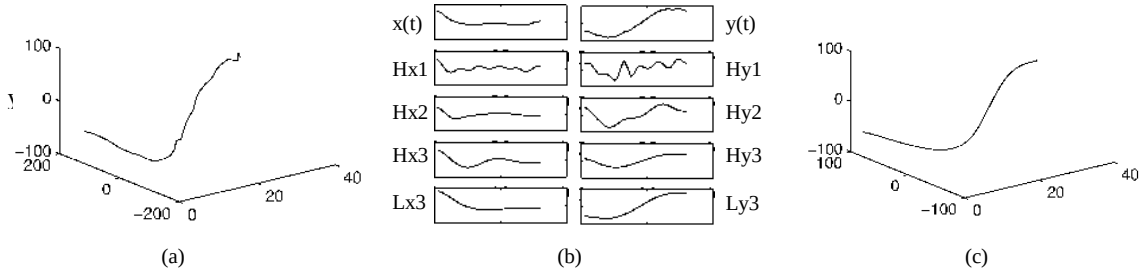


Figure 4. Wavelet transform of a motion trajectory. The original motion trajectory (a), the wavelet transform (b), and the smoothed trajectory that approximates the original (c).

3.2 Object Trajectory Segmentation

We implement the discrete wavelet transform as introduced by Mallat[6]. Since the mother wavelet is defined as the second order derivative of a smoothing function, the detailed coefficients estimate the acceleration of the trajectory at various resolutions. We combine acceleration measurements from the x and y trajectories to calculate the magnitude of the acceleration and segment the trajectory at points of maximum acceleration. As shown in Figure 5, a threshold and merge algorithm detects the peaks in acceleration, which are separated by a minimum interval.

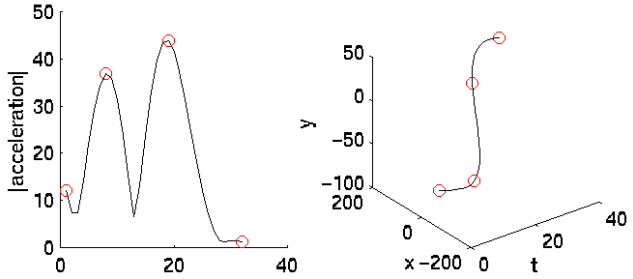


Figure 5. Peaks in the magnitude of acceleration are detected (a) and used to segment the smoothed motion trajectory (b).

3.3 Object Trajectory Modeling

The smoothing and segmentation process divides each object trajectory into subtrajectories where the acceleration does not vary greatly. We fit each subtrajectory with a second order polynomial:

$$\mathbf{r}(t) = (x(t),y(t)) = 0.5\mathbf{a}t^2 + \mathbf{v_o}t$$

$$\mathbf{a} = (a_x,a_y) \text{ acceleration, } \mathbf{v_o} = (v_x,v_y) \text{ velocity}$$

We model each subtrajectory as a feature vector that includes the acceleration, initial velocity, arclength, order, and multiscale edgepoints. The acceleration and initial velocity describe the curvature of the subtrajectory while the arclength measures the spatial distance between endpoints (in 2D-image plane). The order term indexes the spatial and temporal position of the segment within the entire trajectory (first, second, etc). This allows us greater flexibility in similarity matching. Finally, the multiscale edgepoints measure the number of segments at higher resolutions that fall within the endpoints of the segment at the lowest resolution. It is a good measure of the fine variation that exists in the subtrajectory.

Our object trajectory model supports both spatial and temporal translation invariance. We achieve spatial translation invariance by shifting the starting point of a subtrajectory to the origin and extracting features that are independent of absolute spatial position. A user, for instance, might search for the motion of a skier from any starting point in the image plane. We achieve temporal translation invariance by supporting subpattern matching according to the rules described in the next section. Subpattern matching allows the user to query for dominant motions, which are often embedded in more complex trajectories in the database.

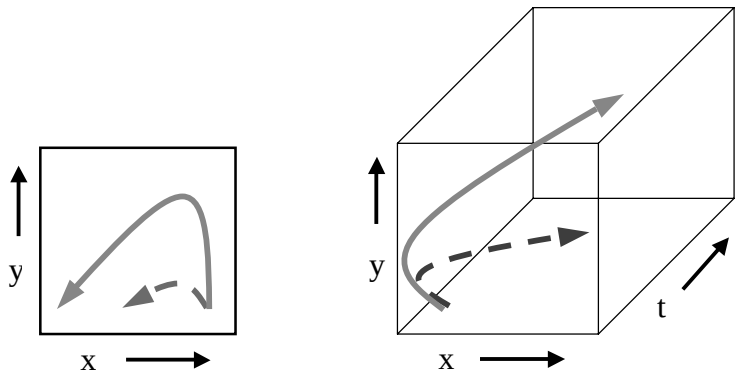


Figure 6. Spatial scale (a) and temporal scale (b) properties of a motion trajectory.

Our object trajectory model does not support spatial and temporal scale invariance. Figure 6 compares the spatial and temporal scale properties between motion trajectories. Figure 6(a) shows one large and one small parabolic curve in the 2D-image plane. We use spatial distance to distinguish between the large parabolic motion of a high jumper jumping over a bar from the small parabolic motions of a person simply walking or jogging. Also, Figure 6(b) shows one parabolic curve and its stretched version over time. We use acceleration and velocity to distinguish between the fast, curving motion of a slalom skier from the slow, winding motion of a recreational one.

4. OBJECT TRAJECTORY SEARCH

Object-based search has been described previously in [7]. Following this approach, we develop a subtrajectory-based search that matches local objects first and then combines these matches according to a global criterion. As shown in Figure 7, once the user inputs a motion trajectory, the algorithm segments it into subtrajectories. The features of each subtrajectory are matched against those in the database. We use a Mahalanobis metric to calculate distance between feature vectors. Then lists of candidate subtrajectories are generated for each one specified in the query. The candidate lists are merged and a global distance is calculated for the matching result. The global distance is used to sort and display the final return results.

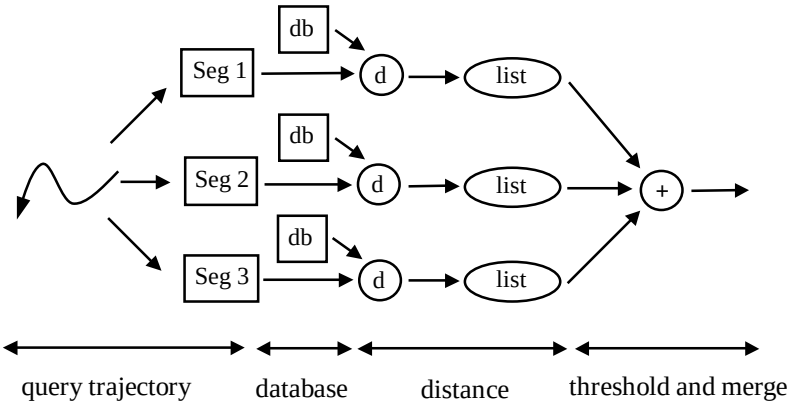
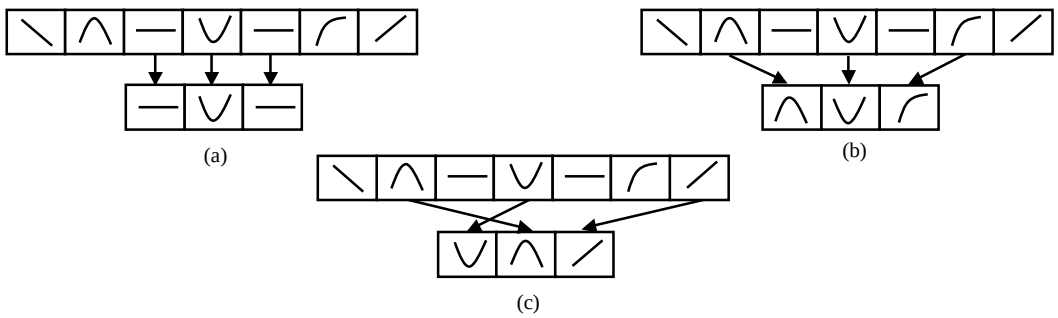


Figure 7. Subtrajectory-based similarity search.

Subtrajectory-based search allows for greater flexibility in matching motion trajectories. A number of global matching criteria can be applied in the final stage to match the user's preferences. Similar to the flexibility described for video indexing [8], we support three distinct types of global matching: dominant segment matching, partial trajectory matching, and full trajectory matching.

1) Dominant segment matching: Only the dominant subtrajectory is used to match those in the database. Dominant subtrajectories can be identified as segments with the largest spatial distance or the largest acceleration term.
2) Partial trajectory matching: A subset of subtrajectories is selected to match those in the database. Partial trajectory matching can be further specified in terms of order as shown in Figure 8.
3) Full trajectory matching: All subtrajectories in the original one must match those in the database as described above in Figure 7.

Figure 8. Partial trajectory matching: (a) strict (b) relative and (c) loose.



Furthermore, we can allow three distinct types of partial trajectory matching: strict, relative, and loose.

1) Strict partial matching: The exact order of the selected objects between the query and database must be the same.
2) Relative partial matching: The relative order of the selected objects between the query and database must be the same.
3) Loose partial matching: No constraints on the order of selected objects. Just match a subset of objects between the query and database trajectories.

5. EXPERIMENTAL RESULTS

5.1 Smoothing, Segmentation, and Modeling

We tested our algorithm on a collection of 1108 object trajectories extracted from the VideoQ object database. Our trajectory database consists of 40 trajectories of high jumpers, 68 trajectories of slalom skiers, and 1000 trajectories of random objects. Figure 9 shows three trajectories after smoothing and segmentation and compares it against the noisy, original ones.

We mark the boundaries of each subtrajectory with a small circle. Each subtrajectory is modeled with the following feature vector:

[order, acceleration in x, acceleration in y, velocity in x, velocity in y, arclength, multiscale edgepoints]

The order component is an index to the temporal and spatial position of the subtrajectory. The partial matching schemes described above use the order component to determine if the subtrajectory satisfies strict, relative, or loose criteria. The acceleration, velocity, and arclength model the spatio-temporal mode of the subtrajectory. Figure 10 compares a reconstruction based on these feature components against the smoothed version. The distance between subtrajectories is calculated by using a Mahalanobis metric on the last six components of the feature model.

5.2 Retrieval Examples

We implemented a content-based retrieval system that queries for motion trajectories by example. The user selects a motion trajectory and the system returns the best six matches. The top left trajectory is the query, and the remaining return results are ranked and displayed from left to right in scan-line order. We plot all trajectories in its three-dimensional spatial-temporal mode. Figure 11 shows some retrieval examples.

In Figure 11(a), we show an example of partial trajectory matching. The two segments in the query are matched to similar ones in the return results. In the third and fifth trajectories, we note that partial matching has been achieved. In Figure 11 (b), we show an example of full trajectory matching. All three segments in the query are matched to exactly three segments in the return trajectories.

5.3 Performance Results

We measure the performance of our systems in two ways. First, we measure the traditional precision/recall curves. Second, we compare the advantages and disadvantages of subtrajectory matching to other global ones, such as simple pointwise and cubic B-spline matching.

5.3.1 Precision vs. Recall

We calculate the precision/recall curves for our trajectory matching schemes. Using our database, we separated 40 object trajectories belonging to the class of high jumpers. These trajectories have distinctive parabolic motions. We query for these motion trajectories and calculate the precision/recall values for subtrajectory matching, pointwise matching, and cubic B-spline matching. Precision/recall values are calculated by varying the return size from 9 to 20. Maximum recall value is 0.5. Figure 12 shows that subtrajectory matching (shown with a solid line) outperforms both pointwise (shown with a dashed line) and cubic matching (shown with a dashdot line).

5.3.2 Pointwise Matching

We compare full-trajectory matching to pointwise matching. The pointwise method normalizes each trajectory. The entire trajectory is shifted to the origin and resampled to 64 points. The raw samples between two trajectories are compared on a point-to-point basis using a Euclidean metric. No smoothing or segmentation is applied. As shown in Figure 13, pointwise matching does not capture the spatial shape of the curve. A steep parabolic trajectory is matched to flat lines. Also, pointwise matching does not capture the temporal features either. The return results vary in temporal duration from the query.

5.3.3 Cubic B-spline Matching

We compare full-trajectory matching to cubic B-spline matching. We implement the cubic B-spline wavelet as described in [9]. Similar to our algorithm, a cubic B-spline wavelet decomposes a trajectory into coefficients at different resolutions. We perform matching directly on the coarse coefficients, using a Mahalanobis metric. Figure 14 compares return results.

The cubic B-spline method performs similarly to the pointwise method. For return results ranked 3-6, neither the extent of the curvature in the image plane or the number of twists and turns in time is matched. Global descriptors such as these cannot capture the local variations both in the image plane and in time. In Figure 15, the full-trajectory matching scheme matches each subtrajectory separately. This ensures that local variations are captured in the return results.

6. CONCLUSIONS

We have presented a robust and flexible algorithm for motion trajectory matching. We propose a smoothing, segmentation, and feature extraction method that can be computed with a single wavelet transform. Such features satisfy several matching criteria for spatial-temporal invariance. We show that subtrajectory matching offers greater flexibility. We also note that if a user is interested in finding just fast trajectories, subtrajectory matching can focus its search on the acceleration and velocity terms. If the user is interested in only the jumping motion of a high jumper, subtrajectory matching can achieve this through a number of partial matching schemes. This eventually allows for intermediate-level queries.

7. REFERENCES

1. S.F. Chang, W. Chen, J. Meng, H. Sundaram, and D. Zhong, " A fully automated content based video search engine supporting spatio-temporal queries," IEEE Transactions on Circuit and Systems for Video Technology, Vol. 8, No. 5, September 1996.
2. N. Dimitrova and F. Golshani, "Motion Recovery for Video Content Classification," ACM Transactions on Information Systems, Vol. 13, No. 4, October 1995, Pages 408-439.
3. S. Dagtas, W. Al-Khatib, A. Ghafoor, and R. Kashyap, "Models for Motion-based Video Indexing and Retrieval," Submitted to IEEE Transactions on Image Processing, Special Issue on Image and Video Processing for Digital Libraries.
4. R. Jasinschi, A. She, T. Naveen, A. Tabatabai, S. Jeannin, and B. Mory, "Motion Descriptors for Content Based Video Representation," Image and Communication Journal 1999.
5. G. Chuang and J. Kuo, "Wavelet Descriptor of Planar Curves: Theory and Applications", IEEE Transactions on Image Processing, Vol. 5, No. 1, January 1996.
6. Stephane Mallat, "Zero-Crossings of a wavelet transform," IEEE Transactions on Information Theory, Vol. 37, No. 4, pp. 1019-1033, July 1991.
7. J.R. Smith and S.F. Chang, "Integrated Spatial and Feature Image Query," Multimedia System Journal, Springer-Verlag. Vol. 7, No. 2, pp. 129 -- 140, March 1999.
8. K. Shearer, S. Venkatesh, and D. Kieronska, "Spatial Indexing for Video Databases", Journal of Visual Communication and Image Representation, Vol. 8, No. 3, September 1997.
9. E. Stollnitz, T. DeRose, and D. Salesin. Wavelets for Computer Graphics: Theory and Applications. Morgan Kauffman, San Francisco, 1996.

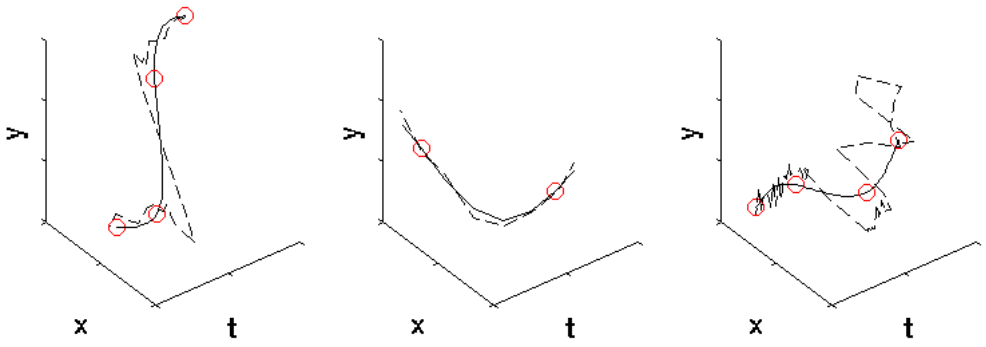


Figure 9. Smoothing and segmentation of three object trajectories. The smoothed trajectories are plotted with a solid line. The original noisy trajectories are plotted with a dashed line. The segmentation points are marked with a small circle.

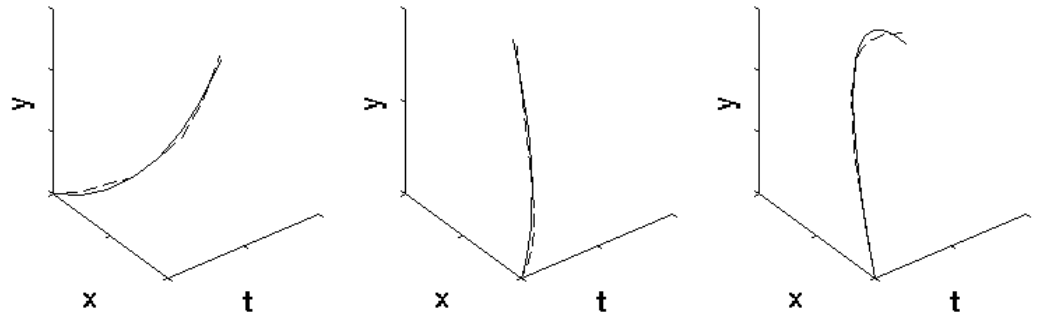
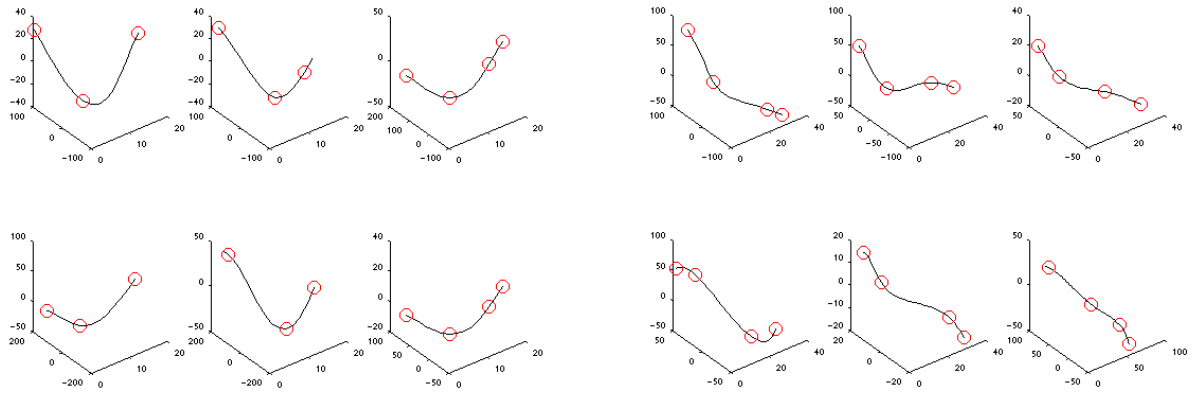


Figure 10. Reconstruction of three subtrajectories from the feature model [a_x,a_y,v_x,v_y,arclength]. The reconstructed segments are plotted with a solid line. The smoothed segments are plotted with a dashed line. These segments belong to the first motion trail of Figure 9.



(a) (b)

Figure 11. An example of (a) partial trajectory matching and (b) full trajectory matching.

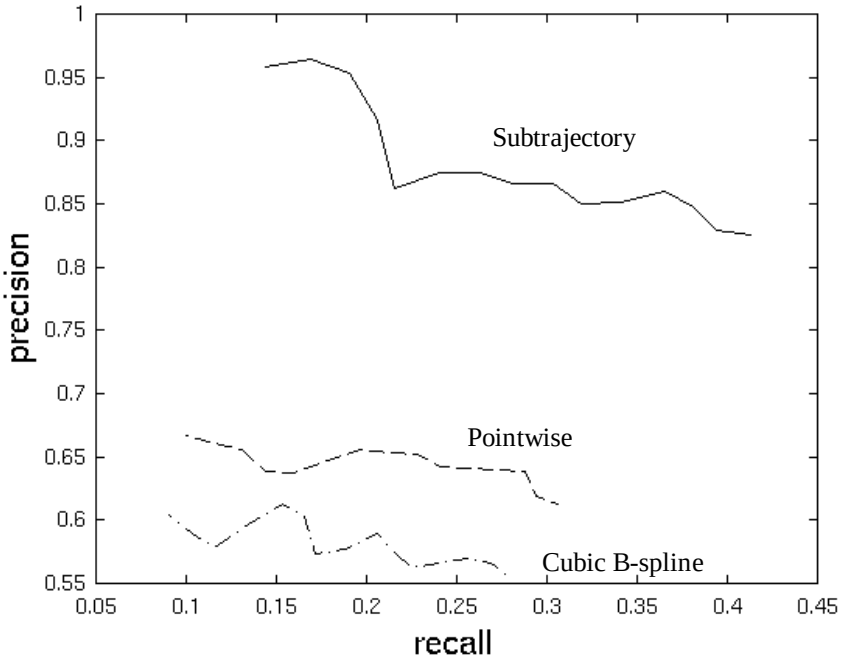


Figure 12 Precision/recall values comparing subtrajectory matching to pointwise and cubic B-spline matching.

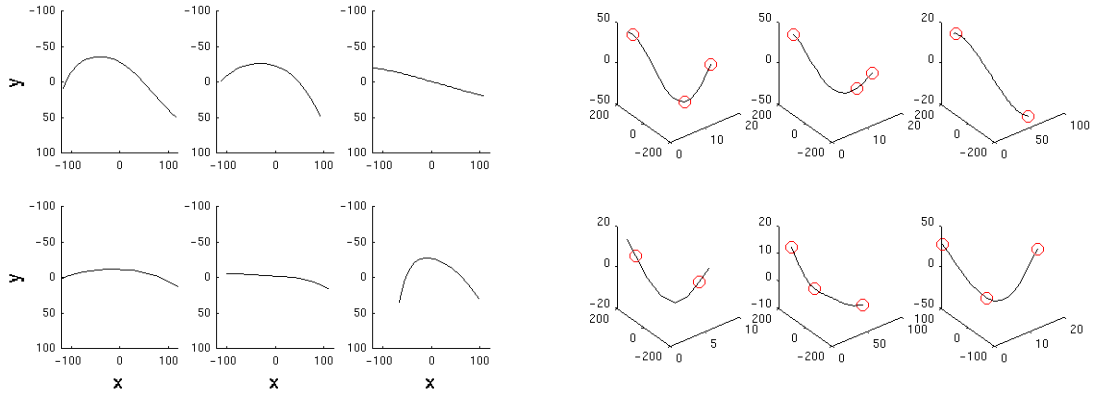


Figure 13 Pointwise matching results. Both spatial and spatio-temporal modes shown.

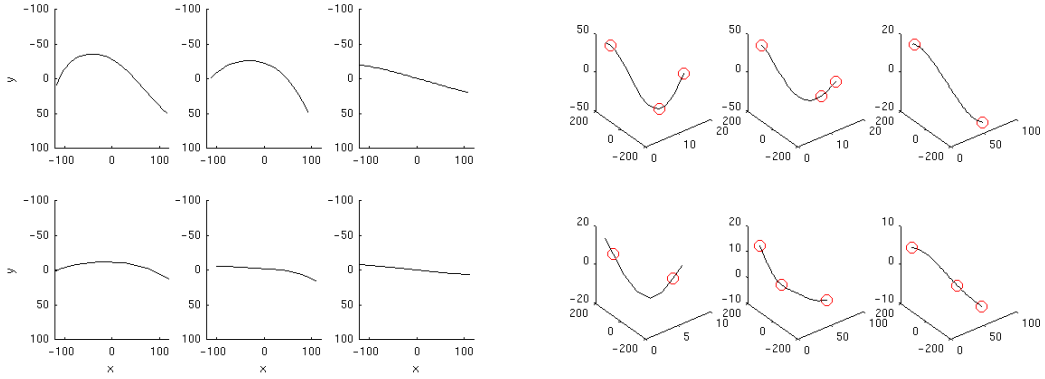


Figure 14. Cubic B-Spline matching results. Both spatial and spatio-temporal mode shown.

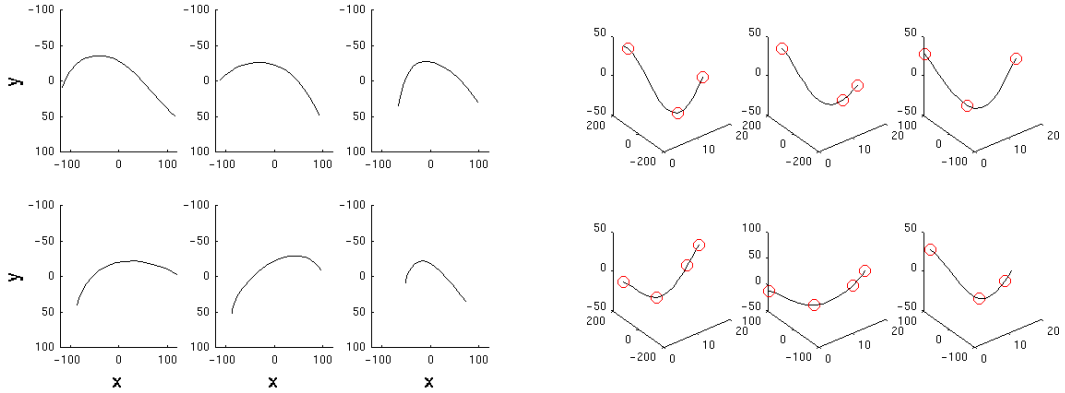


Figure 15. (b) Subtrajectory matching results. Both spatial and spatio-temporal modes shown.