The Trajectory Primal Sketch: A Multi-Scale Scheme for Representing Motion Characteristics

Kristine Gould and Mubarak Shah

Computer Science Department
University of Central Florida
Orlando, FL 32816

Abstract

Conventional approaches to dynamic scene analysis attempt to recover structure of objects using a sequence of frames, under the assumption that the structure information will be used in a recognition system. However, motion itself has generally not been used explicitly for recognition. We propose a *different* approach for the use of motion in a computer vision system which uses the motion characteristics of moving objects without actually recovering the structure. In this approach, we consider extended trajectories followed by the objects. We believe that in many cases, where an object has a fixed and predefined motion, the trajectory of several points on the object may serve to uniquely identify the object. In our approach, the trajectories are analyzed at multiple scales to identify important *events* corresponding to discontinuities in direction, speed and acceleration using scale-space. These important events are recorded in a representation called *Trajectory Primal Sketch* (TPS).

1 Introduction

The world we live in changes with time. Our visual system is capable of discovering these changes; in particular we are able to detect moving objects and recover structure from their motion. In our daily life we move our eyes, head and sometimes whole body in order to perceive the changing environment and to interact with the surroundings. We can recognize objects by looking at their shape only, however, recognition tremendously improves if the motion information is incorporated as well. Conventional approaches to dynamic scene analysis attempt to recover structure of objects using a sequence of frames, under the assumption that the structure information will be used by a recognition system. However, motion itself has generally not been used explicitly for recognition. Moreover, most approaches to the *structure from motion* problem involve a number of assumptions regarding the objects and their motion, and can only deal with a restricted set of cases with a certain minimum number of points in some minimum number of frames. It is also necessary in these approaches to solve systems of non-linear equations using approximate methods which are very sensitive to noise.

We propose a *different* approach for the use of motion, in which the motion characteristics of moving objects are used without actually recovering the structure. In this approach, we consider extended trajectories followed by the objects. We believe that in many cases, where an object has a fixed and predefined motion, the trajectories of several points on the object may serve to uniquely identify the object itself. Therefore, it should be possible to recognize certain objects based on motion characteristics obtained from trajectories of representative points.

One domain of application for a system that recognizes objects based on their trajectories is in a controlled environment such as a factory, where the trajectories of objects can be used to distinguish between stationary and various moving obstacles. This information can then be used in planning a path for a moving piece of machinery. TPS can also be applied to track the motion of stars in the solar system, the trajectory followed by a military target, and chromosome motion in the human body.

The first step in creating the TPS is to represent the changes in motion that occur in the trajectories of points. In Section 2, a method for representing these changes in motion will be discussed. In order to identify which of these changes are significant, the scale-space of the trajectory representation is calculated. This is discussed in Section 3. Section 4 deals with identifying the primitives of motion as represented by trajectories. Determining the primitive types of unknown trajectories is discussed in Section 5. Finally, section 6 presents some of the results we have obtained.

1.1 Background - Primal Sketch

The term primal sketch has previously been used in shape representation. Marr introduced the term when he defined his *raw primal sketch* [2]. The raw primal sketch contains primitives which are *edges, bars, blobs* and *terminations*. These primitives represent the information from the zero-crossings of several channels. The raw primal sketch is used to create the *full primal sketch*. This is done by grouping the primitives in the raw primal sketch into tokens and finding the boundaries among sets of tokens. The main idea is to integrate the information from several channels of zero crossings and identify primitives which correspond to significant intensity changes, and then recursively grouping these changes into boundaries.

Haralick also used the term primal sketch when he defined his *topographic primal sketch* [3]. The primitives of the topographic primal sketch are the *peak, pit, ridge, ravine, saddle, flat* and *hillside*, all of which have a precise mathematical definition. Each pixel in the image is classified as one of these primitives based on the first and second order directional derivatives. These primitives are then grouped into areas which show the reflectance, surface orientation and surface position of objects in the image.

A third example of the use of the term primal sketch is Asada and Brady's *curvature primal sketch* [1]. The curvature primal sketch is a method for representing the significant curvature changes on the boundaries of objects. The prim-

CH2752-4/89/0000/0079$01.00 © 1989 IEEE

itives are *corner* and *smooth join*, and the *crank*, *end*, and *bump* or *dent* which are combinations of corners.

We would like to define a new primal sketch, the ***trajectory primal sketch***, which is used in motion representation. We will define a set of primitives which will represent the motion characteristics of trajectories, and show how unknown trajectories can be described using these primitives.

2 Trajectory Representations

Significant changes in motion refer to a segment of the trajectory where the direction of motion of the point, the speed of the point or both are changing rapidly. In order to identify the significant changes a representation which contains all the information of the trajectory itself while simplifying the motion analysis is necessary. The advantages and disadvantages of each of several trajectory representations are described below.

2.1 Trajectory $\Psi - S$ Curve Representation

In order to easily identify the significant changes in motion, the two-dimensional trajectories are converted to a parametric one-dimensional function based on the $\Psi - S$ curve used in shape representation. In the case of trajectories, Ψ refers to the direction of motion of a point between two sequential time frames with respect to a horizontal line. S refers to the distance that a point moves between time t_i and t_{i-1}. The $\sum_i S_i$ is equal to the total distance a point covers in moving from its initial position to its final position. Ψ is plotted with respect to the distance, S, covered.

S and Ψ are calculated by applying the following equations to the trajectory points where (x_i, y_i) are the coordinates of a point in frame i.

$$\begin{aligned}\Psi &= tan^{-1}\left(\frac{y_i - y_{i-1}}{x_i - x_{i-1}}\right)\\ S &= \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}\end{aligned}$$

The $\Psi - S$ curve accurately records the shape of the trajectory but the possible changes in speed of the point are not obtainable from the $\Psi - S$ curve itself. This is a *serious* problem since the speed of a point, and therefore the time at which a significant change in motion occurs, is essential to the understanding of motion. Therefore, instead of considering Ψ as a function of S, as is done in shape representation, two graphs are used. This can be accomplished using the $\Psi - S$ information directly by plotting Ψ as a function of time and the second curve will plot S as a function of time. However a simpler solution exists.

2.2 Trajectory Velocity Representations

The trajectory velocity representations are another way to incorporate time into the direction and speed information. In this case there are again two graphs, the first one called v_x plots the velocity in the positive x direction against the time, the second graph called v_y plots the velocity in the positive y direction against time.

v_x and v_y are calculated from the following equations:

$$\begin{aligned}v_x &= \frac{x_i - x_{i-1}}{\Delta t}\\ v_y &= \frac{y_i - y_{i-1}}{\Delta t}\end{aligned}$$

If we let t be the frame number rather than actual time then Δt becomes 1, eliminating two divisions and simplifying the equations.

The trajectory velocity representation records all the information available in the trajectory. The changes in motion can be easily identified from the trajectory velocity representation. The segments of the curves which are relatively horizontal correspond to frames when the motion of the point is relatively constant. Segments of the curve where the slope is not zero, or where there is a discontinuity in the curve, correspond to frames when the motion of the point is changing. The greater the magnitude of the slope of the curve, the more rapidly the motion is changing.

3 Trajectory Primal Sketch

The TPS must be a compact representation of the significant changes in motion. We create the first level of this representation by identifying the significant changes at various scales. This results in a set of contours known as *TPS contours*, where each contour corresponds to a significant change in motion. The important information contained in the TPS contours is the position or frame number at which a contour occurs, the height or strength of the TPS contour and the shape of the contour.

3.1 Calculating TPS Contours

Once the trajectory representations have been calculated, the significant changes in motion which appear as discontinuities in these representations, need to be identified. This is done by studying the scale-space of the trajectory representation [6] which we call *TPS contours*. Scale space is used because the discontinuities may appear at various scales depending on the physical motion of the point. The discontinuities in the velocity as well as its first and second derivative have been considered. The resultant representation records the important events in the trajectories of moving points at multiple scales.

The *TPS contours* are calculated by applying a mask to the input signal. This mask is created using the Gaussian and its derivatives, where the value of σ can vary between one and fifteen. The size of the mask is dependent on σ. Since the Gaussian approaches zero after 3σ, the size of the mask should be approximately $2*(3*\sigma)$. We have used an odd sized mask which is equal to $6*\sigma+1$. The result of applying this mask is then checked for zero-crossings which correspond to discontinuities at a particular scale. TPS contours are the curves which result from plotting the position of a zero-crossing on the horizontal axis and the value of σ on the vertical axis.

7	5	4	6	8
	2	1	3	
		*		

Figure 1: Neighborhood for tracking contours.

3.2 Parsing TPS Contours

The TPS contours referred to in the previous section were represented as a simple list of coordinates, $(frame, \sigma)$, which correspond to the frame number and the value of σ at which a zero-crossing occurred. This list does not show any relation between the various coordinates, such as which zero-crossings lie on the same contour. In order to use this data to characterize motion we would like to link these unrelated zero-crossings into contours whose location, strength and size can easily be determined. Witkin refers to this process as Coarse-to-Fine tracking [6].

Tracking begins at the smallest σ value rather than at the largest σ value. We have found that in some cases it is impossible to track a contour entirely from one endpoint to the other. Therefore we will start tracking at the smallest scale since the location of the zero-crossing is most accurate at this scale, and the accurate location of the zero-crossing is extremely important.

A two dimensional matrix is created with the frame number on its horizontal axis and σ on its vertical axis. Elements of this array which correspond to a zero-crossing contain a one. Then the first zero-crossing is located. Next, we begin to track the contour which may begin with this zero-crossing by checking a neighborhood around this element for another zero-crossing. This neighborhood is shown in Figure 1. The current zero-crossing is marked by an $*$ in this figure. The numbers refer to the order in which the adjacent elements are checked for the next zero-crossing. This tracking process continues until there are no zero-crossing in the neighborhood of the current zero-crossing. Then we return to the smallest value of σ and search for the next zero-crossing which may mark the beginning of another contour.

While performing this tracking process, three values are calculated which describe the contours *location*, *strength* and *shape*. The first value, the *location* of the contour, is simply the frame number where the contour originated. The *strength* of the contour is the number of zero-crossings that lie on the contour. The value which represents the *shape* of the contour is calculated by summing the distance between each zero-crossing and the new zero-crossing it is linked to then dividing this sum by *strength* $- 1$. A contour whose next zero-crossing was always found in region one would have a *shape* value equal to one. The greater the shape value is the more curved and irregular the contour is.

Only the contours which survive over a large percentage of the range of σ should be considered to correspond to significant changes in motion. For our experiments we required that the contours survive over 60% of the σ values. The shape value could also be used to pick contours that represent significant changes in motion, since drastic changes in motion result in trajectory representation segments which are similar to step edges, and the scale-space model of a step edge is known to be a straight line [5]. The TPS contains these three values for all contours which survive.

4 Primitive Trajectories

In this section, we will present a number of *primitive* trajectories which can be written as analytical expressions. We will compute the expressions for their v_x, and v_y, and the TPS. For each primitive trajectory, we will identify important *events* in its TPS. The aim is to express an arbitrary trajectory as a composition of these primitives. We will consider four main types of motion: translation, rotation, projectile and cycloid.

4.1 Translation

The trajectory corresponding to translation can be expressed as a sum of straight lines. We know that the equation of a straight line is given by:

$$y = mx + c$$

where m, and c are the slope and y-intercept respectively. Now, incorporating the *time* in the above equation we can rewrite it as:

$$\begin{aligned} x &= at \\ y &= mat + c \end{aligned}$$

Differentiating x, y with respect to t we get the expressions for v_x and v_y:

$$\begin{aligned} v_x &= a \\ v_y &= ma. \end{aligned}$$

The sum of n such straight lines for a translation trajectory are given by:

$$\begin{aligned} x &= \sum_{i=1}^{n} a_i t \left(U(t-t_i) - U(t-t_{i+1})\right) \\ y &= \sum_{i=1}^{n} (m_i a_i t + c_i)\left(U(t-t_i) - U(t-t_{i+1})\right) \end{aligned}$$

where $U(t)$ is the unit step function, defined such that $U(t) = 1$ for $t > 0$ and $U(t) = 0$ otherwise. Now, v_x, v_y are given by:

$$\begin{aligned} v_x &= \sum_{i=1}^{n} a_i t \left(\delta(t-t_i) - \delta(t-t_{i+1})\right) \\ &+ \sum_{i=1}^{n} a_i \left(U(t-t_i) - U(t-t_{i+1})\right) \\ v_y &= \sum_{i=1}^{n} (m_i a_i t + c)\left(\delta(t-t_i) - \delta(t-t_{i+1})\right) \\ &+ \sum_{i=1}^{n} (m_i a_i)\left(U(t-t_i) - U(t-t_{i+1})\right) \end{aligned}$$

where $\delta(t)$ is the impulse function. Now, the TPS contours are the loci of zero-crossings of the following expression:

$$\begin{aligned} g_{tt}^{\sigma} * v_x &= g_{tt}^{\sigma} * \sum_{i=1}^{n} a_i t \left(\delta(t-t_i) - \delta(t-t_{i+1})\right) \\ &+ \sum_{i=1}^{n} a_i \left(U(t-t_i) - U(t-t_{i+1})\right) \\ &= \sum_{i=1}^{n} a_i \left(t_i\, g_{tt}^{\sigma}(t_i) - t_{i+1}\, g_{tt}^{\sigma}(t_{i+1})\right) \\ &+ \sum_{i=1}^{n} a_i \left(g_t^{\sigma}(t_i) - g_t^{\sigma}(t_{i+1})\right) \end{aligned}$$

where $g_t{}^\sigma$, g_{tt}^σ are the respectively the first and second derivatives of Gaussian with zero mean and standard deviation σ $(g^\sigma(t) = e^{-\frac{t^2}{2\sigma^2}})$. And the TPS contours for v_y is given by:

$$\begin{aligned} g_{tt}^\sigma * v_y &= g_{tt}^\sigma * \sum_{i=1}^{n}(m_i a_i t + c)\,(\delta(t-t_i) - \delta(t-t_{i+1})) \\ &\quad + \sum_{i=1}^{n}(m_i a_i\,(U(t-t_i) - U(t-t_{i+1})) \\ &= \sum_{i=1}^{n}(m_i a_i\, t_i + c)\, g_{tt}^\sigma(t_i)) - (m_i t_{i+1} + c)\, g_{tt}^\sigma(t_{i+1}) \\ &\quad + \sum_{i=1}^{n} m_i a_i\,(g_t^\sigma(t_i) - g_t^\sigma(t_{i+1})) \end{aligned}$$

4.2 Rotation

The trajectory corresponding to the rotation around a *fixed* axis can be expressed by an *ellipse*. The ellipse is defined as follow:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$$

where a, and b are the major and minor axes respectively. By incorporating the time and writing this equation separately for v_x and v_y we get:

$$\begin{aligned} x &= r\left(\frac{a^2+b^2}{2}\right)\cos ft \\ y &= r\left(\frac{a^2+b^2}{2}\right)\sin ft \end{aligned}$$

where r is constant, and f is frequency. Notice, that for $a = b = 1$ the above equations reduce to equations for a *circle*. The TPS contours of this trajectory can be obtained by convolving the above equations with g_{tt}^σ:

$$\begin{aligned} g_{tt}^\sigma * v_x &= -\,g_{tt}^\sigma * r\,f\left(\frac{a^2+b^2}{2}\right)\sin ft \\ &= -r\,f\left(\frac{a^2+b^2}{2}\right)\sin ft \end{aligned}$$

since there is no effect of convolving Gaussian with the sine function. Similarly, we get the following for v_y:

$$\begin{aligned} g_{tt}^\sigma * v_y &= g_{tt}^\sigma * r\,f\left(\frac{a^2+b^2}{2}\right)\cos ft \\ &= r\,f\left(\frac{a^2+b^2}{2}\right)\cos ft \end{aligned}$$

Therefore, in the interval 0— 2π the TPS contours of v_x will be three straight lines one each at $0, \pi, 2\pi$ for ft. While the TPS contours of v_y will be two straight lines one each at $\frac{\pi}{2}, \frac{3\pi}{2}$. At those points the gradient of v_x is $-ra^2\frac{a^2+b^2}{2}$, $ra^2\frac{a^2+b^2}{2}$, $-ra^2\frac{a^2+b^2}{2}$ respectively. And the gradient of v_y is $-ra^2\frac{a^2+b^2}{2}$, $ra^2\frac{a^2+b^2}{2}$ respectively.

4.3 Projectile

The trajectory followed by an object in the space due to *gravity* is called projectile, and it is defined as follows:

$$\begin{aligned} x &= v_0(\cos\alpha)\,t \\ y &= v_0\sin\alpha\, t + \mathcal{G}\frac{t^2}{2} \end{aligned}$$

where v_0 is initial velocity, $\mathcal{G}$ is the acceleration due to gravity, and α is the angle. Differentiating above equations with respect to t we get:

$$\begin{aligned} v_x &= v_0\cos\alpha \\ v_y &= v_0\sin\alpha + \mathcal{G}\,t \end{aligned}$$

4.4 Cycloid

The trajectory followed by the object which is rotating around a translating axis is defined by a cycloid. The cycloid is defined as:

$$\begin{aligned} x &= a(ft - \sin ft) \\ y &= a(1 - \cos ft) \end{aligned}$$

where a is constant, and f is the frequency. The expressions for v_x, and v_y are:

$$\begin{aligned} v_x &= a(f - f\cos ft) \\ v_y &= af\,\sin ft \end{aligned}$$

Figure 2 shows an example of each of the primitive types of trajectories. The positions at which the TPS identifies a significant change in motion are marked on the trajectory. The '+' refers to a significant change in v_x, while the 'x' refers to a significant change in v_y.

5 Determining Primitive Type

The TPS contains the location, strength, and shape of the TPS contours for the discontinuities in velocity as well as the discontinuities in the first derivative of velocity i.e. the discontinuities in acceleration. From the analysis and examples in section 4, it is clear that the first derivative discontinuities of both the rotation and cycloid have a sine and cosine relationship to each other. Therefore the first step in identifying primitive types is to separate all the trajectories into two categories. The first category, or the *rotation/cycloid* group, will contain all trajectories which exhibit a sine/cosine relationship in the first derivative. Everything else will be placed into the *translation group*.

The rotation/cycloid group can be broken down further into either rotation or cycloid motion by examining the discontinuities in velocity as well as the discontinuities in the first derivative of velocity. As predicted in the analysis of cycloid motion, the v_x graph for cycloid motion will be shifted some value above or below the zero axis. So when the velocity discontinuities for v_x are calculated, the scale space contours will not look like a cosine curve as they did with the acceleration discontinuities. Therefore v_x and v_y will not have a sine/cosine relationship. However, the scale space of v_x and v_y of a rotating trajectory will have a sine/cosine relationship. Rotation and cycloid motion can be differentiated by checking for a sine/cosine relationship in the velocity discontinuities. If it exists, then the trajectory is an example of rotation, otherwise it is a cycloid.

The trajectories placed into the translation group can also be further classified. The trajectories themselves will be broken down into segments, and each of these segments will be classified as either *straight line translation* or *curved translation*. The value of the slope of the velocity curve is the same as the value of the acceleration of the point within that segment. When a point is moving in a straight line, the acceleration of v_x and v_y must be the same. However, when a point is turning steadily, i.e., moving in a curved line, then the acceleration of v_x and v_y will be different. The slope of v_x and v_y within a segment are compared to see if they are similar to each other within a reasonable bound; if they are then the corresponding segment of the trajectory is labeled as straight line translation. If the values of the slopes are significantly different from each other, then the segment of the trajectory is labeled as curved translation.

The segmentation is accomplished by using the locations of the TPS contours of both the v_x and v_y graphs as the end points of each segment. The slope of v_x and v_y are calculated within each segment using the values of the velocity graphs at each of the end points of the segment. The slope of v_x and v_y are then compared to determine if the translation is straight line or curved.

6 Results

In this section we will present results analyzing the TPS for a number of trajectories obtained from real scenes. The trajectories of real scenes were traced from Sethi and Jain [4]. Since these trajectories were given for a small number of frames, we used interpolation to explode the original trajectories to a large number of frames. We assumed that the motion between any two consecutive frames was in a straight line. In fact, one can compute the extended trajectories with a large number of frames from a moving sequence without using any assumptions. Here, we do not consider that to be any significant factor in our results.

Figure 3 shows four trajectories from the Superman sequence which was used by Sethi and Jain [4]. The TPS for one of the trajectories is shown in Figure 3 (c). This particular trajectory is made by tracking a point on the head of the leftmost man in the scene. Note that all important events, i.e. the places where the direction or speed of the man changes, in this trajectory are captured by the TPS. The changes occur as the man moves both toward the camera and slightly to the right. The changes in v_x occur as the man makes small changes in direction and changes in speed. The changes in direction in v_y correspond to the man's head bobbing up and down as he runs toward the camera. There are also changes in speed. Figure 4 shows the v_x and v_y and corresponding scalespace contours for this trajectory.

Next, the results for the Block scene are shown in Figure 5. In this scene four different blocks are moving towards the center of the image. Trajectories of points on two of these block are shown in Figure 5 (a). These two blocks are a rectangular shaped block which travels from left to right across the image, and a triangular shaped block which moves from the upper right corner of the image to the center of the image. The TPS of a point on the rectangle is shown in Figure 5 (b). The two blocks whose trajectories are not shown are a rectangle which does not move and a triangle which moves from the bottom center of the image to the top center.

Finally, the Figure 6 shows the results for Object scene. This scene consists of three rectangular blocks, one begins at the lower left corner of the image, the second begins in the lower right, and the third begins at the top center of the image. All three of the blocks move toward the center of the image in a straight line, they meet in the middle and then continue away from each other. Trajectories of three points on the block which begins at the lower left corner of the image are shown in Figure 6 (a). The TPS of one of these trajectories is shown in Figure 6 (b).

The primitive type of all the trajectories in this section is translation, and based on the information in the TPS, they are classified as translation by our procedure for determining primitive types.

7 Conclusion

In this paper, we proposed a new approach for use of motion in a computer vision system. In our approach, we use motion characteristics of objects without actually recovering the structure. We outlined a multi-scale scheme for representing the important events in the motion trajectories. These important events correspond to the discontinuities in speed, direction and acceleration of the objects. In our scheme, 2-D trajectories are first converted into two 1-D functions called v_x, v_y. These functions are then analyzed using scalespace in order to identify important events. Primitive trajectories corresponding to translation, rotation, cycloid, and projectile motions were also studied, and it was shown that the proposed TPS scheme can be used to determine a primitive type. From the experimental results for real sequences we also found that the proposed scheme is quite promising.

Future work will focus on several aspects of this approach. First, we will work on method for isolating the trajectories which correspond to the points belonging to one object using TPSs. This is the segmentation problem. Second, the methods will be developed for integrating TPSs of an object into *composite TPS*. Finally, we will look into a use of TPS in a vision system.

References

[1] Asada, H. and Brady, M. *The curvature Primal Sketch*. Technical Report, MIT AI memo 758, 1984.

[2] Marr, David. *Vision*. Freeman, San Francisco, 1982.

[3] R. Haralick ,L. Watson ,T. Laffey. The topographical primal sketch. *Int. J. Robotics Research*, 50-72, 1983.

[4] Sethi, I.K. and Jain, R. *Finding trajectories of points in a monocular image sequence*. Technical Report RSD-TR-3-85, University of Michigan Center for Research on Integrated Manufacturing, April, 1985.

[5] M. Shah, A. Sood, and R. Jain. Pulse and staircase edge models. *Computer vision, Graphics, and Image processing*, 34:321-341, June, 1986.

[6] Witkin, A. Scale-space filtering. In *International Joint Conf. on A.I.*, pages 1019-1021, 1983.

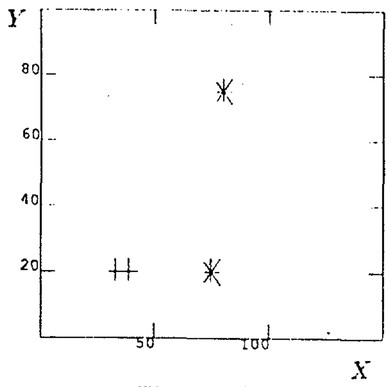

Figure 2 (a)

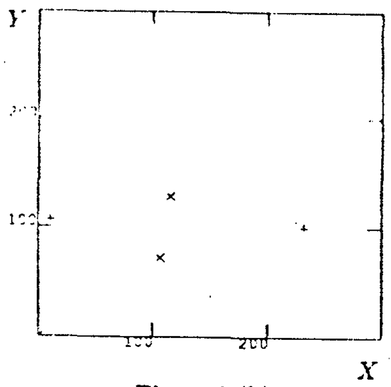

Figure 2 (b)

Figure 3 (a)

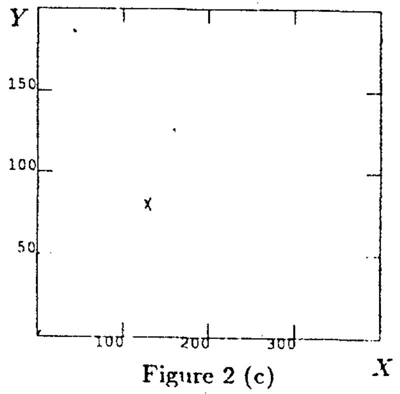

Figure 2 (c)

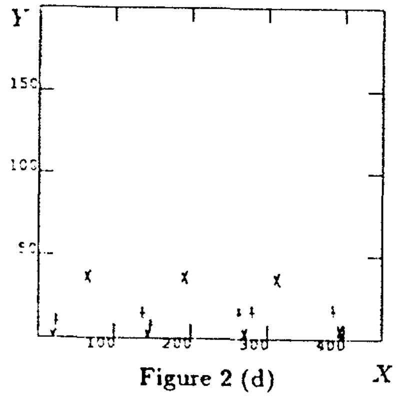

Figure 2 (d)

Figure 2: Examples of primitive types: (a) Translation, (b) Rotation, (c) Projectile, (d) Cycloid.

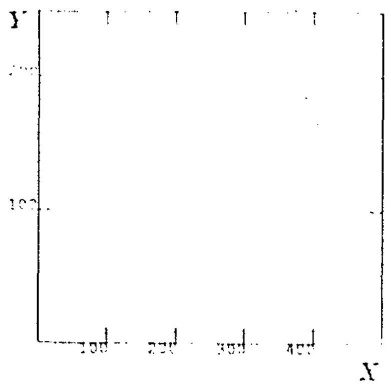

Figure 3 (b)

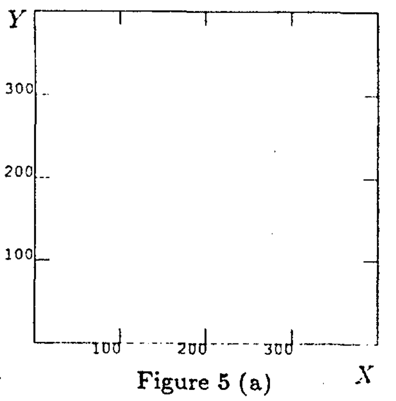

Figure 5 (a)

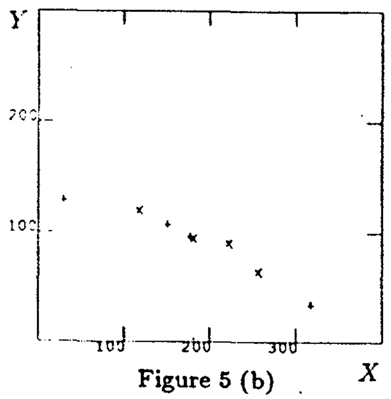

Figure 5 (b)

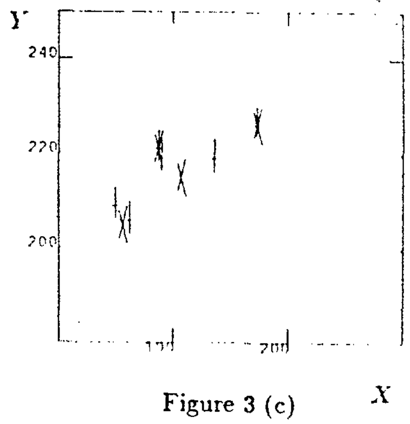

Figure 3 (c)

Figure 5: Block Scene (a) Trajectories, (b) TPS

Figure 3: Superman Sequence (a). One frame of sequence, (b) Trajectories, (c) TPS of one trajectory.

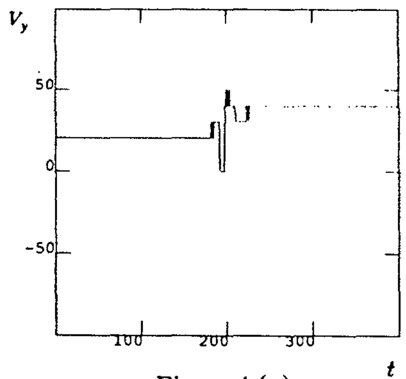

Figure 4 (a)

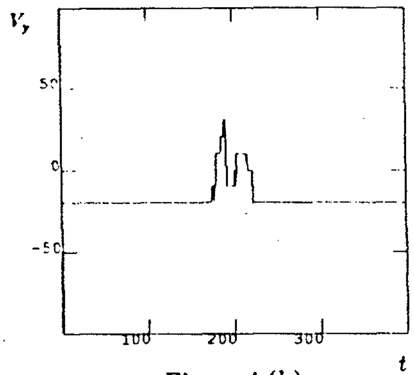

Figure 4 (b)

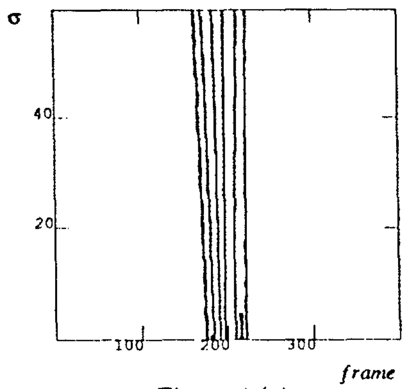

Figure 4 (c)

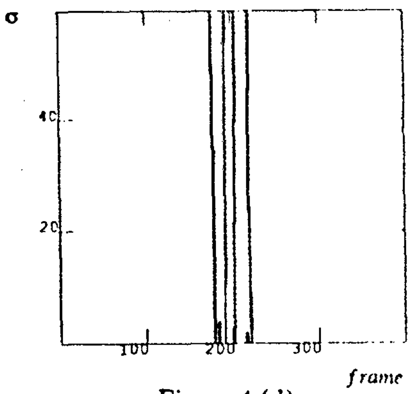

Figure 4 (d)

Figure 4: Trajectory Velocity of S1 and their scalespace (a)v_x, (b) v_y, (c) Scalespace of v_x, (d) Scalespace of v_y.

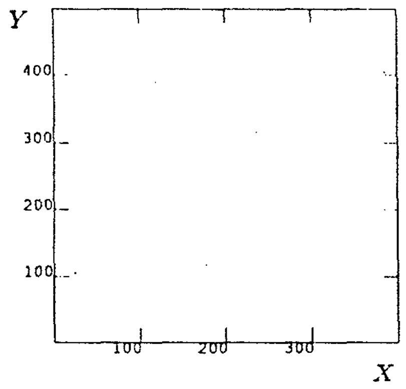

Figure 6 (a)

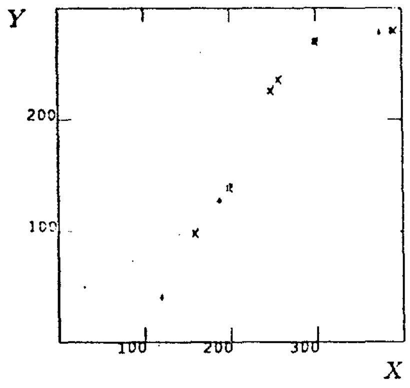

Figure 6 (b)

Figure 6: Object Scene (a) Trajectories, (b) TPS

The research reported in this paper was supported by the Center for Research in Electro Optics and Lasers (CREOL) University of Central Florida under grant number 20-52-043 and by the National Science Foundation under grant number IRI 87-13120.