RSD-TR-12-85

EVENT DETECTION AND CORRESPONDENCE

S. M. Haynes
Ramesh Jain

Department of Electrical Engineering and Computer Science

The University of Michigan

Ann Arbor, Michigan 48109

September 1985

CENTER FOR RESEARCH ON INTEGRATED MANUFACTURING

Robot Systems Division

COLLEGE OF ENGINEERING

THE UNIVERSITY OF MICHIGAN

ANN ARBOR, MICHIGAN 48109-1109

TABLE OF CONTENTS

ABSTRACT

Many problems in motion can be approached by focusing attention on events in the motion of objects. Our concept of event is the very general one of any discontinuity in consistent motion. This paper addresses the use of low-level events comprising discontinuities in the regular consistent motions of feature points. For the experiments described here, events are changes in the parameters for uniformly accelerated motion: acceleration, velocity and initial position. We then use the detected events and the concept of path coherence to achieve a correspondence which describes the motion of objects over many frames. The correspondence mechanism seeks to minimize the number of events.

1. Introduction

Dynamic scene analysis remains difficult, even after several years of hard work. Elegant mathematical formulations describing motions of patches of surface intensities have foundered on the various rocks of imaging noise, instabilities of the solution techniques, and enormous masses of data. To make matters even worse, frequently solutions are not unique. Heuristic approaches have been similarly unsatisfying. For example, using a nearest neighbor heuristic to do correspondence requires careful selection of data. In addition, there is a strong static bias among many dynamic vision researchers. Perhaps this derives from the hope of doing inductive processing as follows: find the "answer" for one frame, then find the rules for taking the answer from one frame to the successive frame. Finding the "answer" for one frame is just the problem of static scene analysis. But static scene analysis techniques also give ambiguous answers. The best known reason for this is the many-to-one mapping of the scene's projection onto the image.

Dynamic scene analysis, like static scene analysis, seeks answers to such high-level questions as where and what are the objects in the scene. It is in order to answer those high level questions that the low-level approaches, e.g., edge detection, region growing, segmentation, have been developed. Dynamic scene analysis has the additional low-level problem of motion detection and characterization.

Optical flow, that is the array of instantaneous velocities on the image, enjoyed a considerable vogue recently, because it can be shown that surface characteristics (i.e., normals) are derivable from the optical flow vectors. The problem is in obtaining the optical flow to sufficient accuracy. The correspondence problem can be viewed as a subset of the optical flow problem. Rather than a dense set of vectors, the correspondence problem is to match a fairly sparse set of points from frame to frame. The points are all identical, there is no grey-level information. There is no satisfactory solution to this problem either. If the new problems of motion characterization make the overall problem harder, then we haven't bought much. It is the strong belief of our group that the appropriate exploitation of dynamism will significantly help the computer vision task. Indeed, we believe that dynamic scene analysis will be more tractable a problem than static vision, once people get away from their static bias.

A cheerful optimism, however, is not sufficient. We need some handles on the immediate problems. Generally people approach this by simplifying the motion allowed, e.g., by requiring the motion to be on a plane only, with the axis of rotation perpendicular to the plane, or by requiring a known ego-motion and stationary world. We prefer to avoid constraining the motion in such a fashion. Instead we will consider how far along toward a solution the notion of an "event" in the trajectory of motion will take us.

1.1. The importance of events

An event is a discontinuity in the motion of an object or point. At a frame rate of 30 frames per second, a point's position will usually be predictable from its previous motions, discontinuities are rare, even for complex motions. Thus event detection will not be a particularly time-consuming process. For the purposes of this paper, we rely on the notion of *path coherence* [10] to define an event. Path coherence is the property that "the motion of an object at any time instant can not change abruptly". An event occurs when the path coherence property is violated. It is important to note that this is a filtering, not a smoothing operation. We do not have points available in advance. Rather, as each frame arrives we then have a new set of data points which must be handled immediately. Jenkin [11] uses a smoothness assumption for tracking lights moving in three dimensions. His system does a nice job obtaining correspondence; his goal is not event detection. An event is a discontinuity, but one that occurs infrequently in the context of many frames. A different definition of event is used in natural language work (see e.g., [18]). In this domain an event is an element in a script. The script describes some stereotypical scenarios. This is a much higher level definition than what we use and is more remote from the data.

The notion of event is intriguing for many reasons. One is that like segmentation, it is a way of reducing the amount of data in an apparently useful way. Another is that many motion verbs appear to incorporate a change of state in their semantics [14]. We feel it worth investigating a possible relationship between physical discontinuities and semantic notions of change to get a handle on the representations of motions.

If events are so rare in continuous motion then why use artificial intelligence techniques to obtain them? Rare events rather imply predictable motions, at least in most cases. And the mathematics of predicted motions are relatively well understood and expressed by optimal estimation theory [3]. And it is indeed true that optimal estimation is very powerful, especially where the state equations are known, and the noise is appropriately behaved. However, the point of optimal estimation is to provide an estimate of the state of the system. Events per se are not accounted for in any sense. If the event is small enough to be within the noise bounds, then the estimator will smooth over the event, giving more inaccurate state estimations, because the data around the event are considered to belong to one data set rather than two distinct sets. If the event is too large and is not within the bounds of noise, then the estimator will not work. Depending on the set-up, it may take more or less time to "recover" from the event, that is until the event is sufficiently outside the window of observations and until that time, the event is adversely affecting the estimates.

Computer vision researchers have been regularly coming up against this problem, in different applications. Consider the static case of edge detection. The problem is what is an edge point, what is not. Because of noise in the image it is standard to smooth the image more or less. If the edge is smoothed out, then the edge will simply not be detected, all the pixels in its area will have a slightly increased edginess value (bad state estimates). If the edge is large enough that smoothing does not get rid of it, then because of the smoothing, the edge cannot be positioned properly. Many researchers have suggested handling this problem by using different window sizes. A large window size is used which will smooth over noise and detect gross changes. Smaller windows are used to detect finer changes. Noise can be ignored by considering results from all sizes in parallel, or by successively applying smaller windows, removing some noise at each step. Some progress has been made in dealing with the edge detection. A second application in which smoothing over the area of interested (smoothing around the "event") causes problems is in surface-fitting to depth values. If the area of fitting contains depth discontinuities the surface fitted will not be correct. Too small an area is adversely affected by noise.

What we are urging here is that whether the discontinuity is in motion parameters for an event, in intensity greyvalues for edges, or in depth values for surfaces, any smoothing process for removing noise must not extend across the discontinuity. Of course, then the problem is how to tell the discontinuity from noise. They look alike. There are two ways (not exclusive) that spring to mind for addressing this problem. The first is to choose an algorithm which can recover from the misclassification between noise and meaningful discontinuity. The second is to choose an algorithm which delays decision making as long as possible. This property is often called *least commitment.*

1.2. Successive refinement

Another technique receiving increasing attention is called "successive refinement". The idea behind successive refinement is that the values one wishes to compute are improved in an iterative fashion as more data is considered. Thus, the first estimate of the value may be only in the ballpark. As more data is used the estimate will approach the true value. It is *not* correct to assume that successive refinement is just like a gradient descent, or any other sort of optimization problem. While there may be an error measure associated with successive refinement, it isn't always necessary; there is not necessarily any underlying function being optimized. Even more important than the distinction between gradient descent and successive refinement is the distinction between smoothing and successive refinement. Smoothing is just a kind of averaging. Successive refinement has nothing to do with averaging. It is an iterative adjustment of the estimate based on new data. In this way successive refinement has conceptual ties to estimation theory. Smoothing may be included in the refinement, but it is not an intrinsic part of it. Successive refinement is particularly attractive in the temporal domain (though it hasn't been used there much), because the data is naturally ordered. The new data used to adjust the estimates is given by the next frame.

Our algorithm for event detection makes heavy use of successive refinement. If the data given by the current frame cannot be used to refine the motion estimates, that is, if the new data violates a consistency criterion, then an event is signalled. If the data is useful, then the motion estimates can be refined in light of it. Thus, our algorithm provides motion parameters and event detection at the same time. Contrast this with smoothing which, first, cannot really handle event detection well, and second, requires the event detection first, before it can do the parameter estimation.

A first problem in dynamic scenes is to extract the instantaneous motion parameters from the digitized frame sequence. The usual approach has been to find the 2-D displacement vectors at some or all points, and then to obtain the 3-D vectors by making assumptions about object rigidity or the nature of the motions. Finding the 2-D displacement vectors has received a great deal of attention. When the data are high-interest feature points, this is correspondence. When vectors are obtained at every pixel and integrated to get velocities, we have a 2-D optical flow field. Two things make finding the 2-D vectors difficult. One problem is that over only two or three frames there are many reasonable vector assignments. The other problem is the sensitivity to noise. Many motion analysis techniques concentrate on what information is available in as few frames as possible. In a noise-free world, sometimes only two frames suffice to give 2-D displacement vectors. Because of the loss of 3-D information when the scene is projected onto the 2-D image, more than two frames are required for complete 3-D information. Frequently an algorithm will iterate over two frames until convergence or termination. There are many examples, e.g., [7], [13], [17]. Many tracking schemes require a solution or a near solution to the matching or correspondence problem over two frames. The match is then used to predict a location in the third frame, which is then used to begin again the matching process. See e.g., [8], [21]. A few researchers have considered the nature of the information in extended frame sequences [9], [20].

2. Doing the event detection and the correspondence

2.1. Event Detection

We solved this problem of event detection by modifying an on-line line fitting algorithm developed by O'Rourke [15]. O'Rourke faced the problem of fitting a line to noisy data. Usually line-fitting algorithms work by minimizing a distance error from each data point to the fitted line. However, suppose that at each time instant a new data point arrives and the desired output is a line fitted to the available data points. Each new data point is either part of the previous line, or else it begins a new line. In most cases, the line cannot be known exactly, rather it is constrained by the previous data points and the amount of noise associated with each data point. Consider Figure 1. Suppose the data point was part of the previous line. At each t_i is a measured x_i with a noise range such that the true x lies in the interval $[x_i - \alpha_i, x_i + \omega_i]$; $\alpha_i, \omega_i \geq 0$. The line must lie within the data ranges for each time instant. Notice that the data for t_1 through t_5 have several possible lines which will fit them. A few are drawn in the figure. The data range at t_6 also can fit a few of the lines drawn from t_1 through t_5, although some of the lines possible for the earlier data are filtered out with the new datum. This is a successive refinement. Now consider the datum for t_7. There is no way any of the lines fitting t_1 through t_6 can be extended through t_7. This signals that a new line is beginning with t_7. Clearly an infinite number of lines can fit the data range t_1 through t_6. These lines are all represented in the computer in a straight-forward fashion by constraining the slope and ordinal intercept. The equation of a line is $x = m\ t + \mu$. Each data point is constrained by the noise range to $[x_i - \alpha_i ,\ x_i + \omega_i]$, where x_i is the data measurement. Each data point constrains further the equation of the line to

$$x_i - \alpha_i \leq m\ t_i + \mu \leq x_i + \omega_i .$$

Note that the unknowns are m: the slope, and μ: the ordinal intercept. The above pair of inequalities can be expressed as two parallel constraint lines in the two dimensional space (m , μ) as follows:

$$\mu \leq - t_i m + (x_i + \omega i)$$

$$- t_i m + (x_i - \alpha_i) \leq \mu .$$

These constraint lines are given by their slopes: $-t_i$ and their intercepts: $(x_i - \alpha_i)$ and $(x_i + \omega i)$. Recalling that t_i is ordered, note that at each successive data point, the slope of constraint lines on the lines being fitted is increasingly negative. In Figure 2 you can see a pair of parallel lines which constrain m and μ. The dotted parallel lines which are obtained from the datum x_{i+1} further constrain m and μ. O'Rourke proves several things about these constraint polygons, especially that they are closed, and that there are two special vertices of the polygon corresponding to $(m_{\min}, c_{\max})$ and $(m_{\max}, c_{\min})$ where m

is the slope and c is the ordinal intercept of the line, and about the order of the algorithm for line-fitting.

Consider the case where we have ballistic motion. The position, x, of a particle at time, t, is given by the usual

$$x(t) = \frac{1}{2}at^2 + vt + \mu$$

where a is acceleration, v is velocity, and μ is initial position. However, x is never known exactly. For our first experiments we require the point to be present at time t_i and further, that its position is bounded by a noise factor. So at t_i, the true x lies in the range

$$x_i - \alpha_i \leq x \leq x_i + \omega_i$$

The data measurement is x_i and α_i and ω_i place bounds on the positional noise. Thus,

$$x_i - \alpha_i \leq \frac{1}{2}a\ t_i^2 + vt_i + \mu \leq x_i + \omega_i$$

Now x_i, α_i, ω_i and t_i are all known, a, v, μ are unknown. So the pair of inequalities describe two parallel constraint planes with normal: $(-\frac{1}{2}t_i^2, t_i, 1)$ and displacements from origin: $x_i - \alpha_i$ and $x_i + \omega_i$. These planes constrain the allowed values of a, v and μ. We can show three (or more) planes for different times intersect to form a closed polyhedron and that special vertices exist.[1] Suppose we have formed such a polyhedron for t_{i-2}, t_{i-1}, t_i. The values

[1]Thanks to Raymond Tse in our laboratory.

a, v and μ are constrained to lie within the polyhedron. A new x_{i+1} for time t_{i+1} provides a new pair of parallel constraint planes. These planes will either

1. Intersect with the polyhedron to form a new polyhedron or

2. Will give a null intersection with the previous polyhedron.

Case (2) is the event detection. This means that the new datum is inconsistent with the constrained motion parameters, and therefore a new piece of smooth motion is beginning. Case (1) give a successive refinement of the object's motion parameters. The polyhedron for a particular trajectory constrains the possible positions for the point in later frames only if there is no intervening event.

Each polyhedron constructed is a hypothesis describing the positional data as a smooth motion. After an event a new polyhedron must be started to account for the new incoming data. For two dimensional motion where the positions are independent of each other, that is, $x(t_i)$ is never dependent on any $y(t_j)$. We use two polyhedra for each trajectory. A null intersection in either or both polyhedra indicates an event in the particle's motion.

O'Rourke and Badler [16] use their algorithm to collect movement primitives, i.e., a time indexed list of positions which are can be described by a linear function of time. The movement primitive is then used to predict the position of the region of interest in the following frame. Then a simulation of the object model is run to resolve inconsistencies and to account for uncertainty in prediction and analysis. The images they use indicate the correspondence problem is not an issue for them. We, on the other hand, assume no *a priori* knowledge of object model. We wish to see how far the idea of events can be carried.

In this paper, an event is the occurrence of a null intersection. When an event in a particle's motion occurs it means that the motion had been smooth enough for a sufficient length of time after which there was a change in the motion parameters. When an event occurs a new intersection polyhedron must be obtained to describe the continuing motion.

For these experiments we have assumed nothing except ballistic motion with occasional discontinuities. The only thing known about the data is the position. There is no grey-level information. To prime the pump then, we combine all possible data points for the first three frames to obtain an initial constraint polyhedron, i.e., for n points, there are n^3 possible initial trajectories. This is, of course, unreasonable. Clearly, *a priori* knowledge of motions will reduce this number drastically. Also, there is information available in the frames themselves which will reduce this. Many researchers have used this; see e.g., [2], [6], [13], [17]. This information is well known. It is used to derive the motion constraint equation [7], [13]. Basically, it is that the greylevel difference over time combined with the spatial greylevel gradient, constrains the allowable velocities for the points, if the velocities are less than some bound. There are, in fact, many very reasonable ways of pruning the space of hypothesized motions. We rely primarily on having a minimum number of data points to constrain motions.

2.2. Correspondence

To show the usefulness of event detection we used it to solve the correspondence problem. The correspondence problem is the problem of matching points, frame to frame, when in a single frame all points appear identical. There have not been many approaches to solving the correspondence problem. Jain and Sethi [10] seek to optimize path coherence by minimizing a distance measure. Barnard and Thompson [1] successfully use a constraint labelling approach which requires a surface coherence: neighboring points will tend to have similar motion values. O'Rourke and Badler [16] rely on a known object model and use constraint propagation to obtain correspondence. Structure from motion researchers assume a correspondence is given as input. Then, as the points' positions on the image change over one or two frames, they provide three dimensional positional values for the points. Most approaches are exceedingly noise sensitive and require perfect correspondence. A correspondence can also be used to segment the points using motion values as the segmentation criterion. This has not received attention; the structure from motion researchers generally assume all points given lie on a single rigid, or, at most articulated, object.

Our solution to the correspondence problem relies on event detection. It is an artifical intelligence approach in that motions are *hypothesized* for points; then the space of hypothesized motions is *searched* for the correspondence which minimizes the number of events.

Suppose we have only two points in the scene. Points **a** and **b** move smoothly for a while, the event detector, as described in the previous section, builds and refines one constraint polyhedron for each smooth motion. Then, suddenly both **a** and **b** have events in their motions at t_i as detected by a **NULL** intersection with each constraint polyhedron and new data value. Before t_i the motion hypothesis for **a** was just the constraint polyhedron. How to continue the motion hypothesis for **a** at t_i when both data points give events? We make *two* motion hypotheses for **a**, one for each data point. Both hypotheses indicate an event occurring at t_i. Both begin a new constraint polyhedron. Point **b** is treated similarly. We now have more motion hypotheses than points being tracked. Some are wrong, but we don't know which until more data arrives.

If a polyhedron successively predicts a point's next location, then the polyhedron is refined using case (1) above. If a polyhedron does not successively predict a next location (case (2) above), then an event in that motion trajectory is signalled, and a new polyhedron is initiated using all the available points to extend the motion. Every time an event is signalled, there is a sudden explosion in the number of hypothesized trajectories. By requiring a minimum length of smooth motion, the hypothesis list is pruned. A single motion hypothesis is essentially a proposed correspondence of a single point over several frames; it looks at the correspondence problem from one point only. When the proposed correspondences of other points are considered as well, the first hypothesis may not be the best or not even a good possibility.

When a correspondence is needed the complete list of hypothesized trajectories is used to cover the data points, and the partition with the minimum number of events is chosen as the correspondence. That is, a search is made over the space of proposed motions to find the set which accounts for all the data and which minimizes the number of events. We chose the criterion of minimizing number of events heuristically because of our opinion that events are rare. It is a global criterion in the sense that we make no attempt to minimize the number of events in a single particle's motion description. Thus, for example, there is no bias for choosing a trajectory with no events if that means the other particles have very complicated motion.

In the event of two or more correspondences based on minimizing number of events, all are outputted. A simple event count is not sufficient to distinguish among them.

Once a constraint polyhedron has been constructed for a hypothesized trajectory, we then can make predictions about the next location of the point. If there is a point sufficiently near the predicted frame-location coordinates, we hypothesize a continuing of the motion. If there is no point sufficiently close, then either the hypothesized trajectory is false, or we have an event. There is no rule for immediately differentiating between false trajectory and event. Instead we allow further data to come in. We have a simple pruning technique of calling those trajectories false which have smooth motions lasting **min-length** number of frames or fewer. For most of our experiments here, **min-length** is 4. A motion must be smooth for at least 4 frames.

The advantages of this method are several. For one thing, the trajectories are not smoothed over events, events are localized and do not affect motion parameter assessment. For another, since all the hypotheses are kept around, a later correspondence will not be adversely affected by the choice of a wrong correspondence earlier on. For yet another advantage, the pruning method is very gentle, yet many spurious polyhedra are thrown out as having insufficient support. For another, the longer the algorithm is allowed to run, the better the results will be. There are, in fact, many very reasonable ways of pruning the space of hypothesized motions. Only two have been implemented; we are not interested in efficiency at this time. One is to disallow branching of hypothesized motions when a prediction is satisfied by the data. The other is to require some minimum number of frames of continuous motion before the occurrence of an event.

3. The Experimental Results

The algorithms were tested on a variety of generated data. We found, as we expected, that after each event there is an explosion in the number of motion hypotheses. As time progresses, the number of hypotheses is reduced.

The data is provided to the program in a standard format of time, x position, and y position. No datum for t_{i+1} is provided before all the data for t_i is read and processed.

Figure 3 shows three correspondences obtained at $t = 6, 10, 17$. The data are 5 point positions given for each $t = 0, 1, \ldots, 17$. The lines drawn between the points are an aid to the reader, they indicate the correspondence. The motion parameters are *not* given by this line. Recall, there are an infinite number of curves fitting each point correspondence as given by the constraint polyhedron. The lines drawn satisfy the appropriate polyhedron. At each time given here, there is only one correspondence which satisfies the minimum event count heuristic. At $t = 6$ the cost is 0, there are no events. At $t = 10$ the cost is 5, and at $t = 17$ the cost is 10. Humans segment these points into one object. The program does not do this, it "sees" 5 points. We propose using the output of this program to do a segmentation.

In Figure 4 we see another sequence of correspondences for 9 points at $t = 6$, 10, 15. Again, these are the only correspondences at these times. The correspondence mechanism has no trouble, even though 5 points have events when physically near the remaining 4 points. The event count at $t = 6$ is 0, at $t = 10$ is 5, and at $t = 15$ is 5.

Figure 5 indicates a sequence of correspondences of 4 points which have events at the same time ($t = 7$). The correspondences shown at $t = 5$, 13, 17 are the only ones at those times. Naturally there are many correspondences possible at other times, $t = 8$ for example.

For Figure 6 we made data consisting of 4 points each having events at various times. At $t = 6$ is shown the only correspondences objtained then at cost 2. Similarly, the correspondence at $t = 8$ is the only one at that time, its cost is 3. At $t = 16$ there are 4 correspondences at cost 7. Two are shown. (We remind you again the lines are drawn as a visual aid. In the case of the "false correspondence" they don't indicate the motion parameter constraints.) At $t = 19$ there is only one correspondence at cost 7.

To test some scenes with accleration we made the sequence of two points whose correspondences are shown in Figure 7. The two points start at the same initial position, one was given linear motion, the other parabolic motion, as shown by the correspondence at $t = 5$. At $t = 6$ the linearly moving point was given an event so that it moved tangentially to the parabolic point. The correspondences given at $t = 9$ and at $t = 10$ indicate the correct motion assignment. The correspondences shown are the only ones for those times. The cost at $t = 10$ is 1. Only the linearly moving point has an event.

The last sequence given in Figure 8 is for two "bouncing balls". The correspondences shown at $t = 3, 5, 7, 10$, and 14 are the only ones for those times. At $t = 7$ there has been one event (at $t = 5$ when the "ball hits bottom"). At $t = 10$ there have been two events. At $t = 14$ there have been three events.

4. Conclusions and Future Work

One can imagine data which will fool this correspondence mechanism because it uses only one cue, albeit a good one. In this case one clearly requires further information to correspond, e.g.,

- Other point characteristics besides motion, or
- A modelling of the collision process so that the rebound positions are in fact predicted.

As an initial test of the usefulness of events we applied the concepts to the problem of motion segmentation and the related problem of correspondence of particles. We have found it is possible to obtain a reasonable correspondence which depends on the history of a particle's motion. This technique requires an upper bound on the amount of position noise. We have not yet addressed the problem of occlusion or of more complicated motion descriptions. These problems are, however, the *reason* for doing the research presented here.

We point out that the current motion research indicates that the problems of occlusion requires good three dimensional modelling and prediction and very probably also understanding of particle motion. We hope to be able to address these issues with our use of events and of smooth trajectories. Because motion events are relatively rare, we can use some simple, gentle and effective pruning techniques to cut down on the branching of possibilities. Our currently implemented search technique for finding the best correspondences is not sophisticated; there are obvious approaches for improvement there.

The prediction of a point's next location is an important aspect of this work which, as we develop improved motion models, will become more influential in reducing computation.

Finally, we note that points on a rigid object display consistent (allowing for 3-D differences) paths. Nagel [12] points out that this is a good way to define an object. We thus look forward to investigating the interaction between an object segmentation and motion segmentation.

There are several areas in which we want to extend this work. One is to minimize the start-up computational burden ($O(n^3)$) using local grey-level information. Another is to loosen the restriction on motion dynamics by using qualitative descriptions of motion rather than numerical descriptions. But most importantly, we want to use this concept of events and event detection as a primitive, as a tool for general motion understanding.

References

[1] Barnard, S.T. and W.B. Thompson, "Disparity Analysis of Images", *IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-2,* No. 4, July 1980, 333-340.

[2] Cafforio, C. and F. Rocca, "Methods for Measuring Small Displacements of Television Images," *IEEE Transactions on Information Theory, IT-22,* No. 5, September 1976, 573-579.

[3] Chang, C-B, Tabaczynski, J., "Application of State Estimation to Target Tracking," *IEEE Transactions on Automatic Control AC-29},* No. 2, February 1984, 98-109.

[4] Grimson, W.E.L. and Theo Pavlides, "Discontinuity Detection for Visual Surface Reconstruction," *Computer Vision, Graphics, and Image Processing 30,* 1985, 316-330.

[5] Hanson, A.R. and E. M. Riseman (Editors), Computer Vision Systems, Academic Press, NY, 1978.

[6] Haynes, S.M. and R. Jain, "Detection of Moving Edges," Computer Vision, Graphics, and Image Processing, 21, 1983, pp 345-367.

[7] Horn, B.K.P. and B. Schunck, "Determining Optical Flow," Artificial Intelligence, 17, 1981, pp 185-203.

[8] Ishikawa, S.M. Yamada and S. Ozawa, "A Method of Estimating the Target Position for an Image Tracking System of Moving Targets," *International Conference on Pattern Recognition,* 1984, pp 9-12.

[9] Jain, R., W.N. Martin, and J.K. Aggarwal, "Segmentation through the Detection of Changes Due to Motion," Computer Graphics and Image Processing, 11, 1979, pp 13-34.

10] Jain, R. and I.K. Sethi, "Establishing Correspondence of Non-Rigid Objects Using Smoothness of Motion," Workshop on Computer Vision, Annapolis, April 1984.

[11] Jenkin, Michael, "Tracking Three Dimensional Moving Light Displays" *ACM/IEEE Workshop on Motion: Representation and Perception,* Toronto, 1983, 66-70.

[12] Nagel, H-H, "Formation of an Object Concept by Analysis of Systematic Time Variations in the Optically Perceptible Environment," Computer Graphics and Image Processing 7, No. 2, April 1978, pp 149-194.

[13] Nagel, H-H and W. Enkelmann, "Towards the Estimation of Displacement Vector Fields by "Oriented Smoothness" Constraints," *7th International Conference on Pattern Recognition,* 1984, pp 6-8.

[14] Novak, H-J, "On the selection of verbs for natural language description of traffic scenes," 6th German Workshop on Artificial Intelligence, Bad Honnef, September 1982, pp 22-31.

[15] O'Rourke, J. "An On-Line Algorithm for Fitting Straight Lines Between Data Ranges," Communications of the ACM 24, No. 9, September 1981, pp 574-578.

[16] O'Rourke, J and N.I. Badler, "Model-Based Image Analysis of Human Motion Using Constraint Propagation," *IEEE Trans on Pattern Analysis and Machine Intelligence PAMI-2,* No. 6, November 1980, pp 522-536.

[17] Schalkoff, R.J. and J. Labuz, "An Integrated Spatio-Temporal Model and Recursive Algorithm for Image Motion," International Conference on Pattern Recognition, 1984, pp 530-533.

[18] Schank, R. and R.P. Abelson, Scripts, Plans, Goals, and Understanding, Lawrence Erlbaum: Hillsdale, NJ, 1977.

[19] Shamos, M.I. and D. Hoey, "Geometric Intersection Problems," *IEEE Symposium on Foundations of Computer Science,* 1976, pp 208-215.

[20] Tsotsos, J.K., A Framework for Visual Motion Understanding, Technical Report CSRG-114, University of Toronto, Computer Systems Research Group, Toronto, Ontario, Canada, June 1980.

[21] Tsuji, S, M. Osada, and M. Yachida, "Three Dimensional Movement Analysis of Dynamic Line Images," International Joint Conf on Artificial Intelligence 1979, pp 896-901.

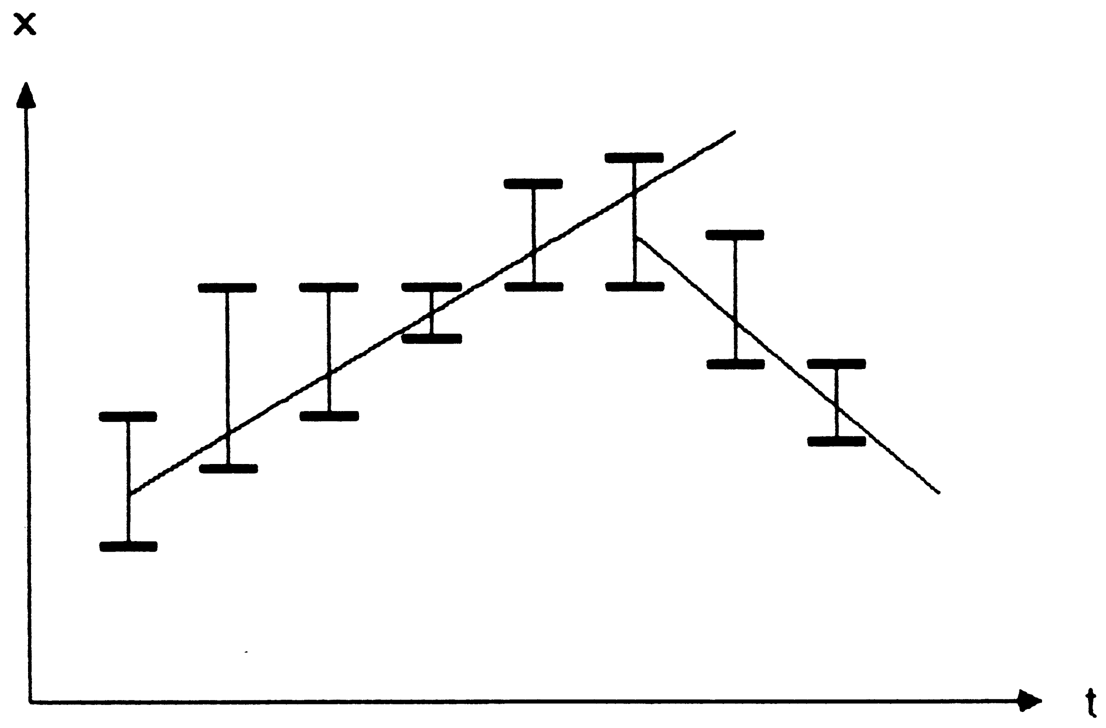


Figure 1. Fitting lines to data ranges.

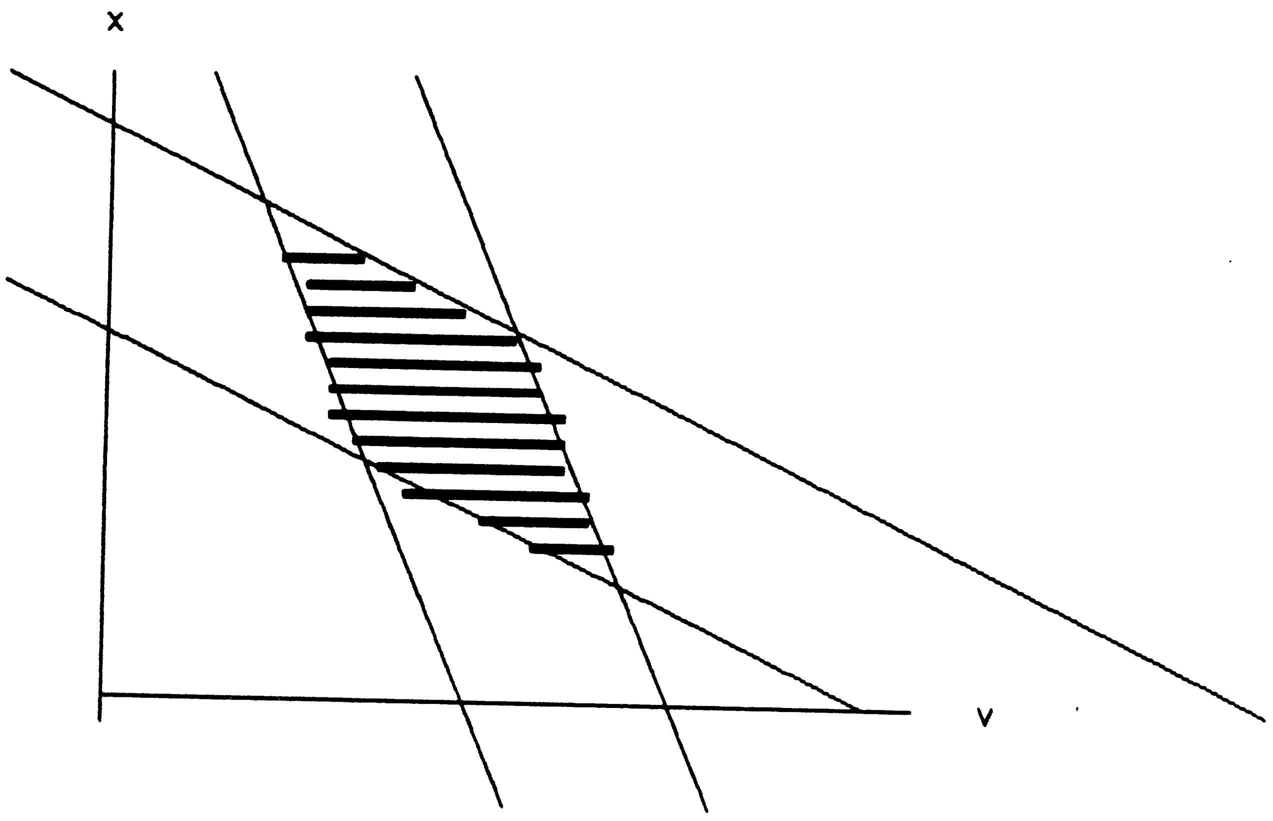


Figure 2. Constraining slope and intercept.

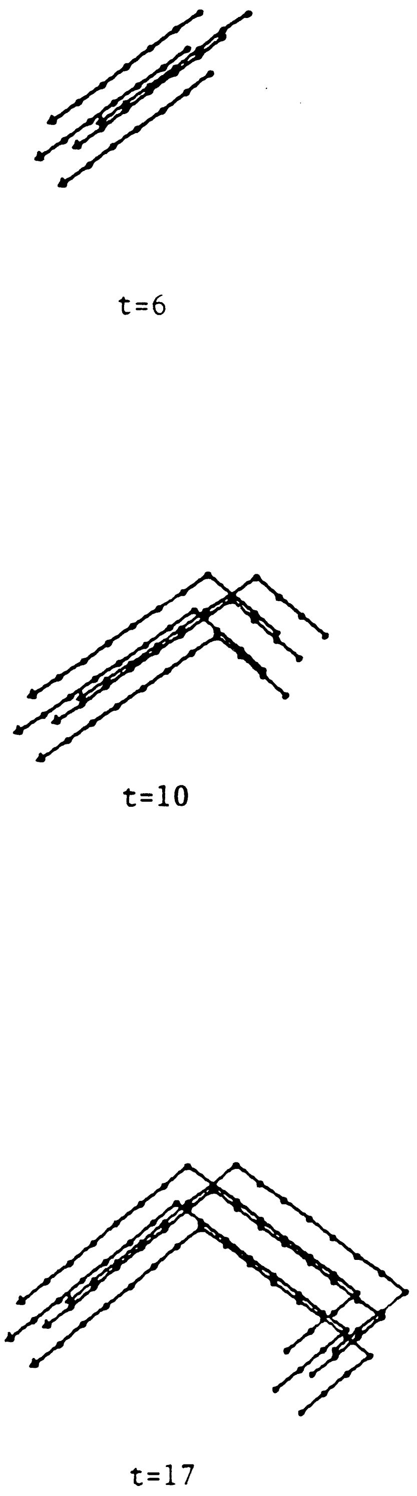
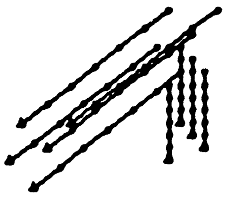
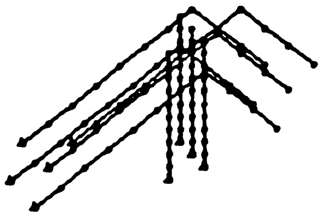


Figure 3. 5 points with similar motion parameters.


t=6


t=10

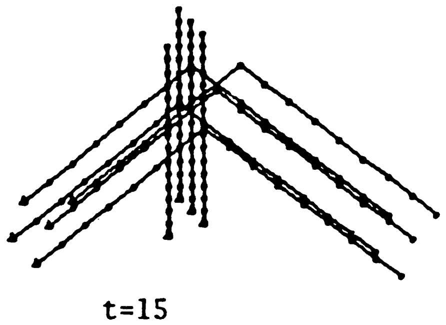

t=15

Figure 4. 9 data points: 5 have one event each, 4 have none.

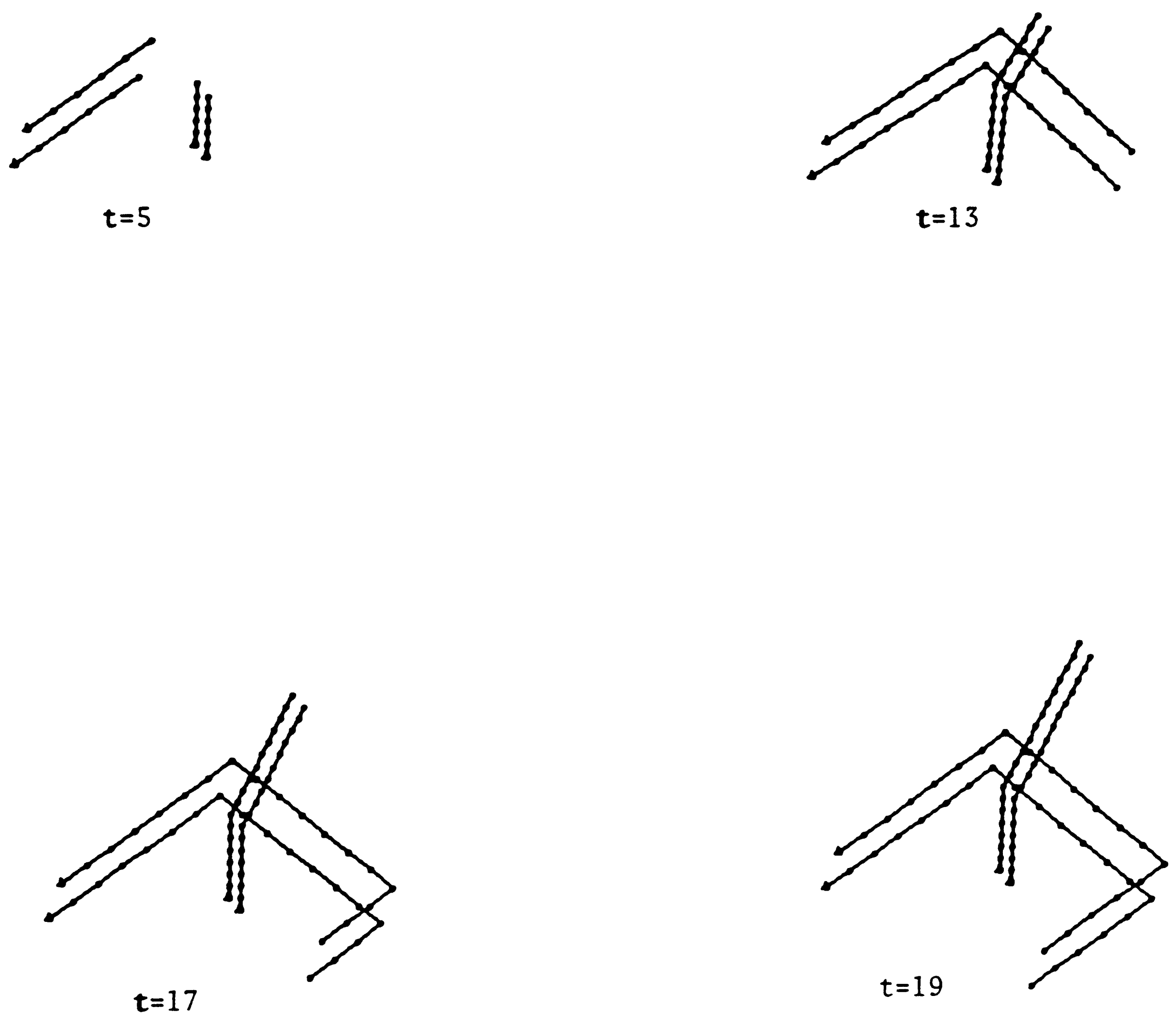


Figure 5. Two sets of two points each.

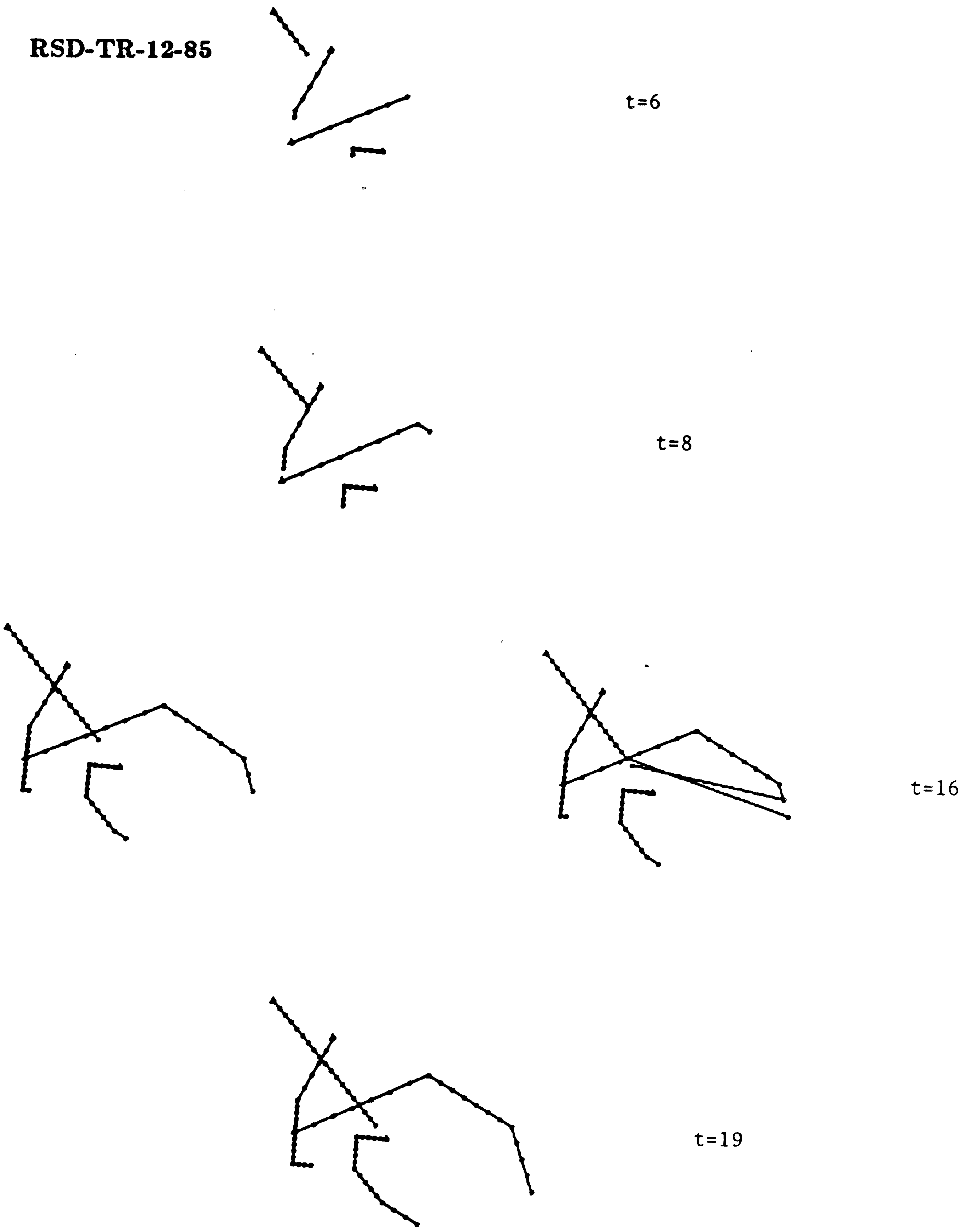
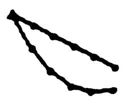
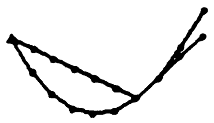


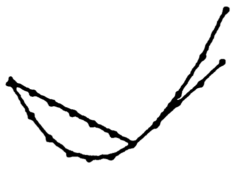
Figure 6. 4 points with differing motion parameters.



t=5



t=9



t=10

Figure 7. Parabolic and linear motions.

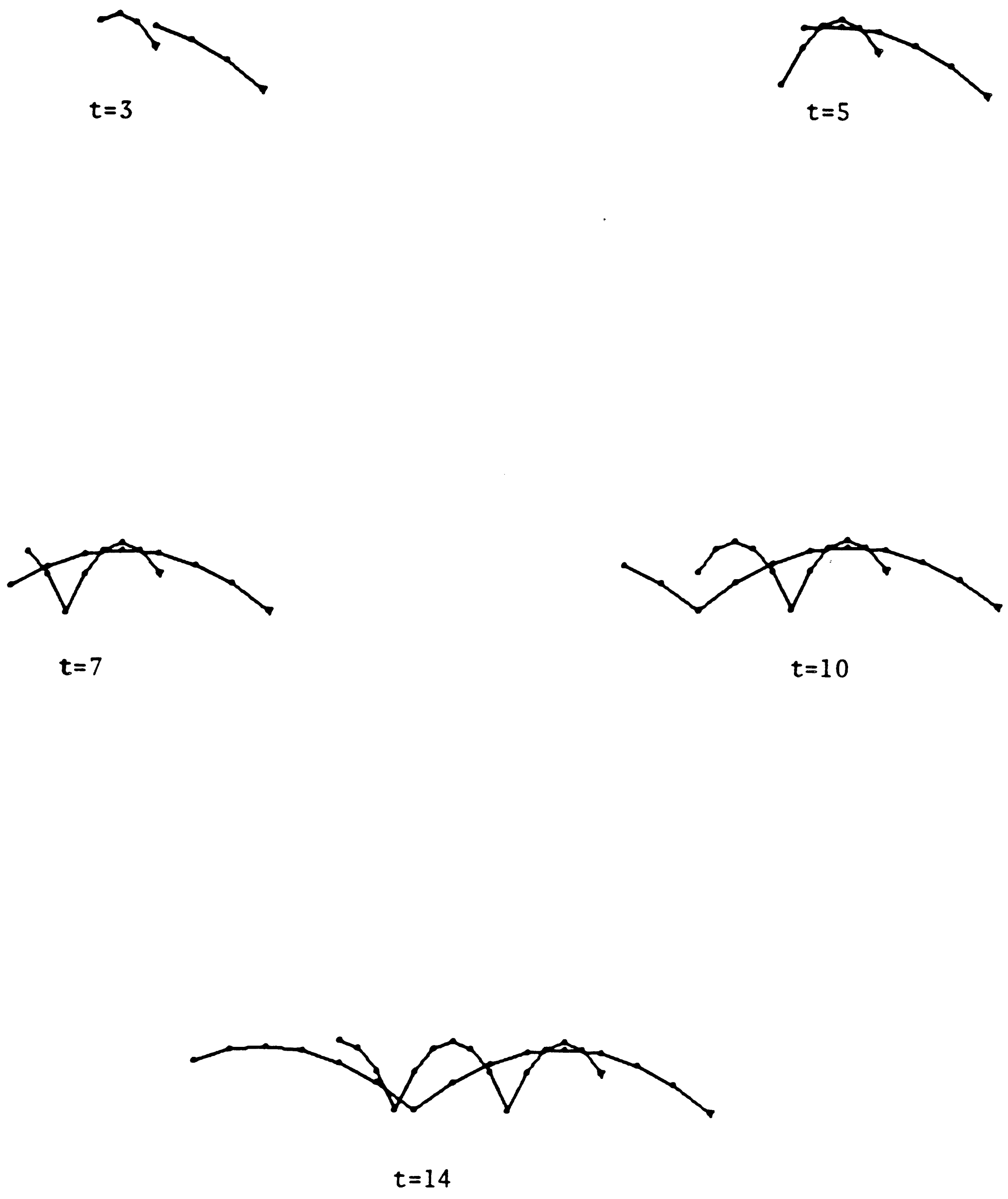


Figure 8. Two "bouncing balls".