Biol. Cybern. 56, 159–175 (1987)

Biological Cybernetics

Projected Free Fall Trajectories

I. Theory and Simulation

Bror V. H. Saxberg

M.I.T. Artificial Intelligence Laboratory, 545 Technology Square, Cambridge, MA 02139, USA

Abstract. How we manage to reconstruct the three-dimensional character of the world from the two-dimensional representations on our retinae has been a lively subject of research in the last ten or fifteen years. One principle that has emerged unifying many of these ideas is the need for constraints to allow the visual system to interpret the images it receives as three-dimensional. These constraints come from assumptions about the nature of the situation that produced the image. We have looked at how gravity can be used as a constraint in the case of a free fall trajectory projected onto an image plane by central projection. We have examined several possible methods for deriving the initial conditions of the trajectory from the two-dimensional projection, and examined their behavior under noisy and noiseless conditions, using both image simulations and videotapes of a real ball. We show that there are several ways to robustly compute the initial conditions of the parabolic trajectory from the image data in the presence of noise.

1 Introduction

The ability of a tennis player to return a 100 mph service indicates how well our visual system functions at rapidly making use of information in a visual scene to make estimates about objects' motion in three dimensions. We have examined a common visual situation, the free fall trajectory of a ball, and derived methods for using the projected images of this trajectory to make estimates about the ball's motion. Elsewhere (Saxberg 1985) we describe experiments with human subjects to determine if humans can use the kinds of information suggested by the methods in this article to determine the landing point of a thrown ball.

Structure from motion research has a psychophysical foundation reaching back to the early fifties, when Wallach and O'Connell (1953) asked subjects to interpret projections of twisted wires. When the wires were in motion, their three-dimensional structure was clearly apparent to the subjects; when the wires were stationary, subjects could discern no three-dimensional character to the projections. Using computers, even more complicated projection situations can be created, e.g. two concentric cylinders counter-rotating, their surfaces effectively covered with large spots and then projected onto the compuer screen (Ullman 1979).

One approach to understanding human performance is to first understand what is theoretically possible, and then look to see whether human performance matches this performance. This may be particularly useful when the information processing task is very complex, as it can provide a framework from which to develop hypotheses and experiments.

A number of theories for how the understanding of 3-d structure from motion information could theoretically be accomplished have been advanced. In a broad way, these theories divide along two different planes into four categories: first, either orthographic or perspective projection is employed, and second, either several discrete still views are used or a single view incorporating velocities and possibly accelerations is used. The latter distinction among theories tends to disappear in computer realizations, since the usual way to get velocity information about an image point is to use two different camera images closely spaced in time. The former distinction is more substantial: orthographic projection can be described as setting one of the three coordinates describing each feature or point in a scene to zero, while perspective projection involves setting up a central point and a projection surface; the projection of a feature point is the intersection of the projection surface with the ray from the central point to the feature point. In the limit of large distances and

small objects of interest in a single view, orthographic projection scaled by the distance to the central point is approximately equivalent to perspective projection; for very large objects at closer distances or for large movements in depth, perspective distortion and scaling are important in modelling the input to the visual system.

As Marr (1982) has pointed out, crucial to the visual system's ability to extract information from the retinal maps are the assumptions about the world it brings to bear if no conflicting information is available. The notion of assumed constraints on three-dimensional behavior is also crucial for the theoretical interpretation of visual events. All structure from motion results involve constraints on the three-dimensional structure or motion of a scene in order to derive that structure and motion from two-dimensional projections. Very complicated fluid behaviors of two objects can lead to identical projections of points, but the assumption of rigidity or other assumptions about the nature of the physical object reduces the number of possible solutions to one or a very few. Hildreth (1983) has looked at the motion of contours and curves in two dimensions and also shown how constraints play an important role in our interpretation of the movement of those curves: we tend to lose the tangential motion of a contrasty line curve or contour because there are not enough textures or features to identify unique points on the curve. Instead, Hildreth suggests we use the perpendicular components of velocity and a constraint, from which we regenerate a full velocity field we assume is appropriate. This reconstruction based on a constraint predicts certain optical misinterpretations which can be shown to correspond to human optical illusions.

In the case of a rigid object with four points identified between images, Ullman (1979) showed how three orthographically projected views could provide enough information to reconstruct the object. Hoffman and Flinchbaugh (1982) showed how three different orthographic projections of the endpoints of a rod are sufficient to determine its structure and motion when the rod is constrained to rotate in a plane around one of its endpoints; similarly, two rods joined together whose combined movement is constrained to lie in a plane also have their structure and motion defined by only two different orthographic views. Webb and Aggarwal (1981) showed that by assuming a rod rotates around a fixed axis, the structure and motion of the rod can be determined from four orthographic projections of the end points. Longuet-Higgins (1981) showed how eight points seen in two different perspective projections determine the structure and motion of those points from the solution of a set of simultaneous linear equations; Tsai and Huang (1984) have developed probably the most general technique for determining the number of points under different conditions of colinearity or other dependency required to give a unique solution for the three-dimensional structure and motion of a rigid surface from two perspective views. Prazdny (1980) and Longuet-Higgins and Prazdny (1980) both investigated using velocity flow fields (continuous areas of velocity information) from central projection of an object in motion to determine structure and motion. Waxman and Wohn (1984) have developed methods for extracting three-dimensional information about surfaces from the motion of physically generated edge contours in the image flow field. Hoffman (1980) has done work on using velocity flow fields to reconstruct three-dimensional motion in the case of orthographic projection.

Todd (1981) has looked at a particular sub-group of free-fall trajectories in some detail: those which are entirely contained in a vertical plane including the viewer. The curve drawn out by the projection of the center of mass of these trajectories on a retina is a straight line; the author does not mention that one can exactly solve for the initial conditions of such a free-fall trajectory if one knows the acceleration due to gravity and its direction. The author does show how ratios of three-dimensional variables can be calculated from the image, and shows that these can give useful information about the trajectory such as time to contact and the object's angle of approach to the projection surface. Indeed, as long as the object does not rotate as it moves or rotates only about a cylindrical axis of symmetry, given knowledge of just one actual parameter of a rigid object's motion (e.g. its real diameter), one can immediately define exactly the object's position and motion regardless of the nature of the motion.

In this article we extend the results on parabolic free fall trajectories, showing that there are a number of techniques for extracting the information available in a visual scene of a three-dimensional free fall trajectory not restricted to a single plane. These techniques can be used even in the presence of noise.

2 Noiseless Trajectories

We are interested in the simple situation of a falling or thrown ball without friction and what information and techniques one might use to predict where the ball will land. As an object moves, its projection on a retina or a camera's focal plane changes size: smaller when further away, larger when closer. The object typically travels against some kind of textured background: buildings, trees, the stands in a stadium, or clouds in the sky. The center of mass of the object also draws out an image trajectory on the focal plane or retina. It is this last cue that we have examined in some detail: how much

information the projection of the trajectory contains, how it can be used, and how the effects of noise in the image on estimates of the ball's trajectory can be mitigated.

We will use central projection; that is if (x, y, z) is a point in three-dimensional space, where x is height, y is width, and z is depth, we will project it onto a point $(\hat{x}, \hat{y}) = (x/z, y/z)$ on the projection plane, parallel to the $x-y$ plane, but located a distance 1 out along the z-axis. We will refer to the three-dimensional time-dependent trajectory as a "trajectory", while the three-dimensional curve traced out by the trajectory will be called a "path;" "image trajectory" and "image path" are defined similarly, but refer to the projections of the trajectory and path respectively. We will examine in detail the behavior of a thrown ball; in this case the center of mass and the visual center of the ball exactly coincide (assuming spherically symmetric mass distribution); in general, the visual center of an object will have a trajectory which is perturbed somewhat from that of the center of mass.

Trying to use the data contained in the image path of the trajectory without assumptions on the nature of the trajectory leads to a multiplicity of possible trajectories. There is a surface formed by lines extending from the origin through all points on the image trajectory; in principle, any trajectory which touches the line from the origin to the starting point of the image path and touches the line from the origin to the ending point on the image path, and staying on the surface between these two lines will generate the projected image (e.g. trajectory a in Fig. 1).

We can begin to reduce this ambiguity by assuming the trajectory is generated by a freely falling object under gravity without friction (trajectory b in Fig. 1);

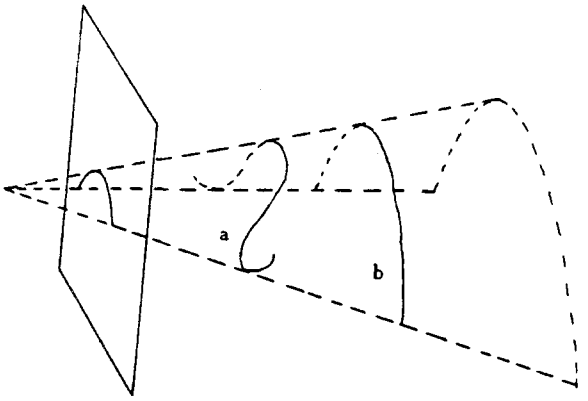

Fig. 1. The projection point is at the left, and the projection plane is shown as a rectangle. Trajectory a is a complicated curvilinear trajectory that happens to project to the same trajectory as trajectory b, which is a parabolic free fall trajectory. Any trajectory on the generalized conic surface defined by the image path and the projection point will project to a subset of the image path

we will also assume the projection plane is parallel to gravity. The trajectory becomes:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_0 + V_x t - 0.5at^2 \\ y_0 + V_y t \\ z_0 + V_z t \end{pmatrix}, \tag{2.1}$$

where x_0, y_0, z_0, V_x, V_y, and V_z are the initial conditions of the trajectory. The image trajectory is expected to be of the form

$$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{pmatrix} \dfrac{x_0 + V_x t - 0.5at^2}{z_0 + V_z t} \\ \dfrac{y_0 + V_y t}{z_0 + V_z t} \end{pmatrix}. \tag{2.2}$$

Multiplying through by the denominator and rearranging terms, we have, in matrix form

$$\begin{bmatrix} \hat{x} & \hat{x}t & -1 & -t & 0 & 0 \\ \hat{y} & \hat{y}t & 0 & 0 & -1 & -t \end{bmatrix} \begin{bmatrix} z_0 \\ V_z \\ x_0 \\ V_x \\ y_0 \\ V_y \end{bmatrix} = \begin{bmatrix} -0.5at^2 \\ 0 \end{bmatrix} \tag{2.3}$$

One approach to finding the unknown initial conditions would be to use several different times $t_1, \ldots, t_m$ to generate more equations, and try to solve for the initial conditions either by inversion or by a least squares technique. This allows us to solve for the initial conditions of x and z coordinates independently of the y coordinate, and is useful if the trajectory has no horizontal component at all [e.g. the experiments of Todd (1981) discussed earlier]. We'll return to this method when we examine robust ways of computing the initial conditions.

Another technique involves taking derivatives of (2.3)

$$\begin{aligned} &-\dot{\hat{x}} z_0 + V_x - \hat{x} V_z - \dot{\hat{x}} t V_z = at \\ &-\dot{\hat{y}} z_0 + V_y - \hat{y} V_z - \dot{\hat{y}} t V_z = 0 \\ &-\ddot{\hat{x}} z_0 - \dot{\hat{x}} V_z - \ddot{\hat{x}} t V_z - \dot{\hat{x}} V_z = a \\ &-\ddot{\hat{y}} z_0 - \dot{\hat{y}} V_z - \ddot{\hat{y}} t V_z - \dot{\hat{y}} V_z = 0. \end{aligned} \tag{2.4}$$

The last two equation in (2.4) can be rewritten in matrix form as

$$\begin{bmatrix} -\ddot{\hat{x}} - (2\dot{\hat{x}} + \ddot{\hat{x}} t) \\ -\ddot{\hat{y}} - (2\dot{\hat{y}} + \ddot{\hat{y}} t) \end{bmatrix} \begin{bmatrix} z_0 \\ V_z \end{bmatrix} = \begin{bmatrix} a \\ 0 \end{bmatrix}. \tag{2.5}$$

Solving for z_0 and V_z using Cramer's rule, we have

$$\begin{aligned} z_0 &= \frac{-a(2\dot{\hat{y}} + \ddot{\hat{y}} t)}{2(\ddot{\hat{x}}\dot{\hat{y}} - \ddot{\hat{y}}\dot{\hat{x}})} \\ V_z &= \frac{\ddot{\hat{y}} a}{2(\ddot{\hat{x}}\dot{\hat{y}} - \ddot{\hat{y}}\dot{\hat{x}})}. \end{aligned} \tag{2.6}$$

This gives us a direct, local way of computing the initial depth location z_0 and initial depth velocity V_z from time derivatives of the projected trajctory when the horizontal component of the image trajectory is changing.[1] From Eqs. (2.2) and (2.4), we have

$$\begin{aligned} V_x &= at + \dot{\hat{x}} z_0 + \hat{x} V_z + \dot{\hat{x}} t V_z \\ V_y &= \dot{\hat{y}} z_0 + \hat{y} V_z + \dot{\hat{y}} t V_z \\ x_0 &= 0.5at^2 + \hat{x} z_0 - t V_x + \hat{x} t V_z \\ y_0 &= \hat{y} z_0 - t V_y + \hat{y} t V_z \end{aligned} \tag{2.7}$$

so we can get full information about the initial velocity and initial position of the parabolic trajectory at $t=0$.

If we had used the orthographic projection of the parabolic trajectory, we could not have uniquely determined the initial conditions. The projection of the parabolic trajectory

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_0 + V_x t - 0.5at^2 \\ y_0 + V_y t \\ z_0 + V_z t \end{pmatrix}, \tag{2.1}$$

where x_0, y_0, z_0, V_x, V_y, and V_z are as before initial conditions of the trajectory, onto the $x-y$ plane would be:

$$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{pmatrix} x_0 + V_x t - 0.5at^2 \\ y_0 + V_y t \end{pmatrix}. \tag{2.8}$$

We would still have to scale the orthographically projected image to fit it into a retaina or camera; this scaling factor can be recovered from the acceleration in the x direction. One has completely lost track of any movement in depth, however, and there is a continuum of possible trajectories that would project to this same image trajectory.

The above calculations were done under the assumption that gravity was acting in a direction parallel to the projection plane. In fact, this assumption is not necessary, although it does make calculations easier. We can see this by rewriting the projection equations in vector form:

$$\hat{\mathbf{P}} = \|\mathbf{m}\|^2 \frac{\mathbf{P}}{\mathbf{P}\cdot\mathbf{m}} - \mathbf{m}, \tag{2.9}$$

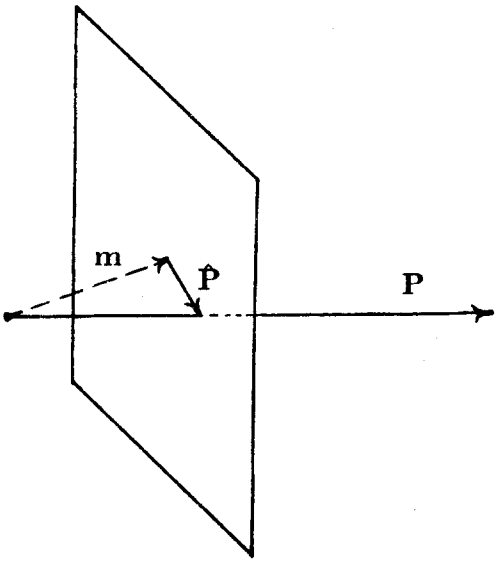

Fig. 2. In this coordinate free representation, the projection point is off to the left, and the projection plane is represented as a rectangle with vector **m** perpendicular to the plane from the projection point. **P** is the vector out to a point in the world; $\hat{\mathbf{P}}$ is the vector from the tip of **m** to the intersection of **P** with the projection plane

where $\hat{\mathbf{P}}$ is the projected image vector lying on the image plane, **P** is the three dimensional vector out to the point of interest, and **m** is the projection vector running from the origin to the projection plane and perpendicular to it (we essentially assumed **m** to be a unit vector in what we did above, although this is not necessary). The vector $\hat{\mathbf{P}}$ runs from the origin of the projection plane (defined as the intersection of **m** with the projection plane) to the intersection of the ray **P** with the projection plane, and it is this vector and its changes in the image plane as the trajectory proceeds that we are concerned with (Fig. 2). If we take derivatives with respect to time we see

$$\begin{aligned} \|\mathbf{m}\|^2 \mathbf{P} &= (\mathbf{P}\cdot\mathbf{m})\,[\hat{\mathbf{P}} + \mathbf{m}] \\ \|\mathbf{m}\|^2 \dot{\mathbf{P}} &= (\dot{\mathbf{P}}\cdot\mathbf{m})\,[\hat{\mathbf{P}} + \mathbf{m}] + (\mathbf{P}\cdot\mathbf{m})\dot{\hat{\mathbf{P}}} \\ \|\mathbf{m}\|^2 \ddot{\mathbf{P}} &= (\ddot{\mathbf{P}}\cdot\mathbf{m})[\hat{\mathbf{P}} + \mathbf{m}] + 2(\dot{\mathbf{P}}\cdot\mathbf{m})\dot{\hat{\mathbf{P}}} + (\mathbf{P}\cdot\mathbf{m})\ddot{\hat{\mathbf{P}}}. \end{aligned} \tag{2.10}$$

Assuming we know the gravity acceleration vector, **a**, we have

$$\begin{aligned} \mathbf{P} &= \mathbf{P}_0 + \mathbf{V}_0 t + 0.5\mathbf{a}t^2 \\ \dot{\mathbf{P}} &= \mathbf{V}_0 + \mathbf{a}t \\ \ddot{\mathbf{P}} &= +\mathbf{a}. \end{aligned} \tag{2.11}$$

[1] The denominator of (2.6) can be expressed in a different way:

$\dot{\hat{x}}\ddot{\hat{y}} - \ddot{\hat{y}}\dot{\hat{x}} = \dot{s}^3(\hat{x}''\hat{y}' - \hat{y}''\hat{x}')$,

where s represents arc-length, s is the speed of traversal of the trajectory, and the $'$ denotes derivative with respect to arc-length; thus,

$\dot{\hat{x}}\ddot{\hat{y}} - \ddot{\hat{y}}\dot{\hat{x}} = \dot{s}^3\kappa$,

where κ is the curvature of the projected trajectory path

If we write the second derivative of the projection equation as

$$\|\mathbf{m}\|^2\ddot{\mathbf{P}}-(\ddot{\mathbf{P}}\cdot\mathbf{m})[\hat{\mathbf{P}}+\mathbf{m}]=2(\dot{\mathbf{P}}\cdot\mathbf{m})\dot{\hat{\mathbf{P}}}+(\mathbf{P}\cdot\mathbf{m})\ddot{\hat{\mathbf{P}}}, \quad (2.12)$$

then we know or can measure all the quantities on the left hand side, and also know or can calculate the coefficients for the unknowns $(\dot{\mathbf{P}}\cdot\mathbf{m})$ and $(\mathbf{P}\cdot\mathbf{m})$ on the right hand side. This can be rewritten as a matrix equation

$$\|\mathbf{m}\|^2\ddot{\mathbf{P}}-(\ddot{\mathbf{P}}\cdot\mathbf{m})[\hat{\mathbf{P}}+\mathbf{m}]=[2\dot{\hat{\mathbf{P}}},\ddot{\hat{\mathbf{P}}}]\begin{bmatrix}(\dot{\mathbf{P}}\cdot\mathbf{m})\\(\mathbf{P}\cdot\mathbf{m})\end{bmatrix}; \quad (2.13)$$

as long as the image velocity and image acceleration vectors are not colinear, this becomes a problem of solving three linear equations, two of them independent, for two unknowns. Note that the form of this solution is similar to the solution when the projection direction **m** is assumed perpendicular to gravity: we solve first for the current velocity and position in depth as measured in the direction of the projection vector, **m**. This does not directly give us our velocity and position initial conditions for depth because **m** and gravity (i.e. $\ddot{\mathbf{P}}$) are not necessarily perpendicular here, and so $(\mathbf{P}\cdot\mathbf{m})$ and $(\dot{\mathbf{P}}\cdot\mathbf{m})$ are possibly time varying. However, by rewriting the projection equation and its first derivative with the initial conditions $\mathbf{V}_0$ and $\mathbf{P}_0$, we see that

$$\mathbf{V}_0=\frac{(\dot{\mathbf{P}}\cdot\mathbf{m})[\hat{\mathbf{P}}+\mathbf{m}]+(\mathbf{P}\cdot\mathbf{m})\dot{\hat{\mathbf{P}}}}{\|\mathbf{m}\|^2}-\mathbf{a}t, \quad (2.14)$$

allowing us to solve for the initial velocities, $\mathbf{V}_0$, and

$$\mathbf{P}_0=\frac{(\mathbf{P}\cdot\mathbf{m})[\hat{\mathbf{P}}+\mathbf{m}]}{\|\mathbf{m}\|^2}-\mathbf{V}_0t-\mathbf{0.5}\mathbf{a}t^2, \quad (2.15)$$

allowing us to solve for the initial positions $\mathbf{P}_0$.

As long as we know which direction gravity is pointing in, it doesn't matter if we move our image plane to an orientation not parallel to the gravity vector; we can still directly use local computations of image velocity and accelerations to give us complete information about the three-dimensional trajectory. The condition that $\dot{\hat{\mathbf{P}}}$ and $\ddot{\hat{\mathbf{P}}}$ be non-colinear is exactly the condition that the image of the parabolic trajectory not be a line segment, analogous to the condition that there be some horizontal motion in the image when the projection plane is parallel to gravity.

Before looking at noisy data, we can apply the theoretical technique described above to a "noise-free" situation and see how accurate the calculations can be. The direct, local computation requires us to calculate derivatives of the image trajectory arrays. In the case with no significant noise, we can rely on the inherent smoothness of the data and precision of the computer to give us good estimates using a very simple difference operator to approximate the derivative. We show the calculated estimates for z_0 and V_z using the direct local method in Fig. 3. At each instant of time, an estimate is made of the initial depth and initial velocity of the trajectory: these are the estimate curves. (Note the expanded distance and velocity scales.) As one can see, the calculation yields quite good results with negligible

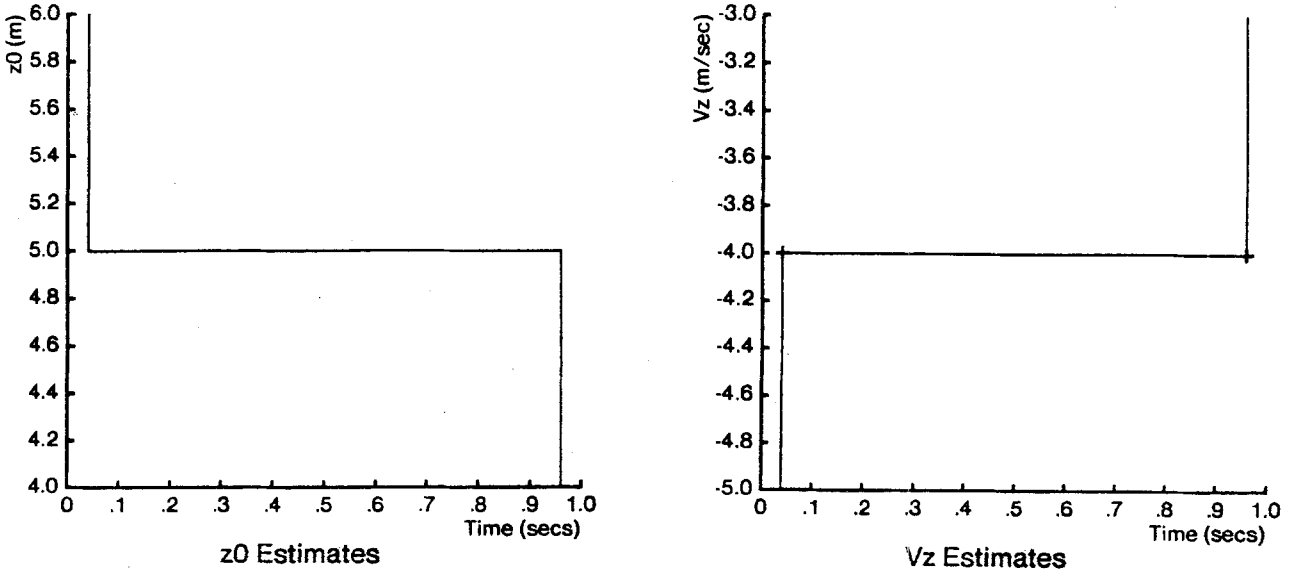

Fig. 3. Graphs of estimated z_0 (*on the left*) and V_z (*on the right*) versus time for noiseless data from the forward trajectory (Fig. 5) using the local derivative calculation. The initial conditions are calculated at each moment in time, and the estimates are plotted versus time

error everywhere that the derivatives make sense even using a simple method for estimating the derivatives of the digitized signal.

Real images, unfortunately, are noisy. How can we handle the introduction of noise into the projected data? The methods used above are no longer immediately useful, since simple derivatives of noisy data bury important information in extreme variation.

3 Noisy Trajectories

In this section we will discuss several different ways to overcome the effects of noise on our calculations. The first method we will look at allows us to extend the calculations done in the previous chapter to the noisy case: we filter the signal with a linear filter in such a way as to remove as much of the noise energy as possible without distorting the signal. The second method attempts to explicitly make use of global information about the image trajectory by using a least squares technique to calculate the initial conditions. The third method is based on optimal signal processing theory, and attempts to pick initial conditions to minimize some measure of the expected error between the actual image trajectory data and the estimated image trajectory.

We simulate trajectories on a Symbolics 3600 Lisp Machine as shown in Fig. 4. The distance of the projection point from the projection plane is taken to be 0.85 cm; different values for this distance correspond to different focal lengths of the projection lens. The object departs from an initial point on the ground 2 m below the projection center with an initial velocity in some direction; the trajectory is simulated by computing evenly spaced points in time (every 0.01 s) until the point contacts the ground again. These points are then projected onto the image plane assuming the projection plane is oriented perpendicular to the ground. The resulting image trajectory arrays $\hat{x}$ (vertical position on the image plane) and $\hat{y}$ (horizontal position) are then available for analysis by the implementations of different methods. We can examine the performance of these methods by looking at the estimates for the initial conditions created by each method using all the information available up to a particular time t; we can then plot these estimates versus t to show how these estimates change during the course of the trajectory. Since depth information is the most difficult to recover from the trajectory, we will tend to focus our attention on the calculation of z_0 and V_z. Typically, we will show the results of calculations on two representative trajectories, shown in Fig. 5, known as the forward and backward trajectories respectively.

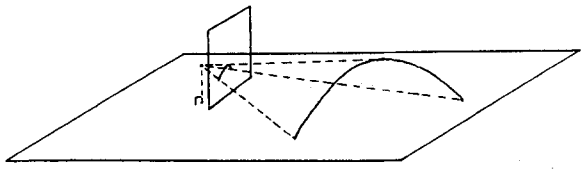

Fig. 4. The viewing point is taken to be 2 m above the ground. The ball begins its trajectory from ground level 5 m directly ahead of the projection plane. The distance between projection point and projection plane is taken to be 0.85 cm. The image of the object is found by intersecting a line from the projection point to the ball's visual center with the projection plane

If we add noise to the image data, we expect our performance to degrade depending on the amount of noise. Noise in all our simulations here is the addition of independent, uniformly distributed random numbers (between -0.003 and 0.003 perspective units, where the perspective units are the units of measure on the projection plane: see Fig. 5) to the $\hat{x}$ and $\hat{y}$ data arrays. The added noise completely disrupts the calculation when derivatives of the noisy waveforms are taken directly.

To study the effects of noise on estimates of the parameters of a trajectory, we can generate an ensemble of estimate curves by repeating the same trajectory with newly generated noise for many trials (typically 50 in our case). We can then qualitatively examine the ensemble of estimate curves generated this way and can compute the mean estimate curve from the ensemble as well as variance curves by computing the mean and variance at each point in time over the ensemble of curves. This gives us an idea of how well the estimate curve will approximate the correct parameter value as time progresses. If the mean of the ensemble is wrong, the method is biased; if the variances are very large, the method is not reliable: the method is very sensitive to the noise.

3.1 Filtering

Filtering a noisy signal is one way of taming the derivatives. Filtering with low-pass filters is aided by the near parabolic nature of the image signal: one can show that if the signal begins from ground level, the first zero crossing of the spectrum of the signal has to occur before $1/T_f$ in the frequency domain, where T_f is the time to return to the ground; this suggests that most of the signal energy will be within a few multiples of $1/T_f$ in the frequency domain. If we assume the noise spectrum is much wider than the signal spectrum, we expect to get a significant reduction in the contribution of noise to the signal and its derivatives by preprocessing with a low-pass filter just wide enough to allow the significant components of the signal to get through. Derivatives of the filtered signal may be taken

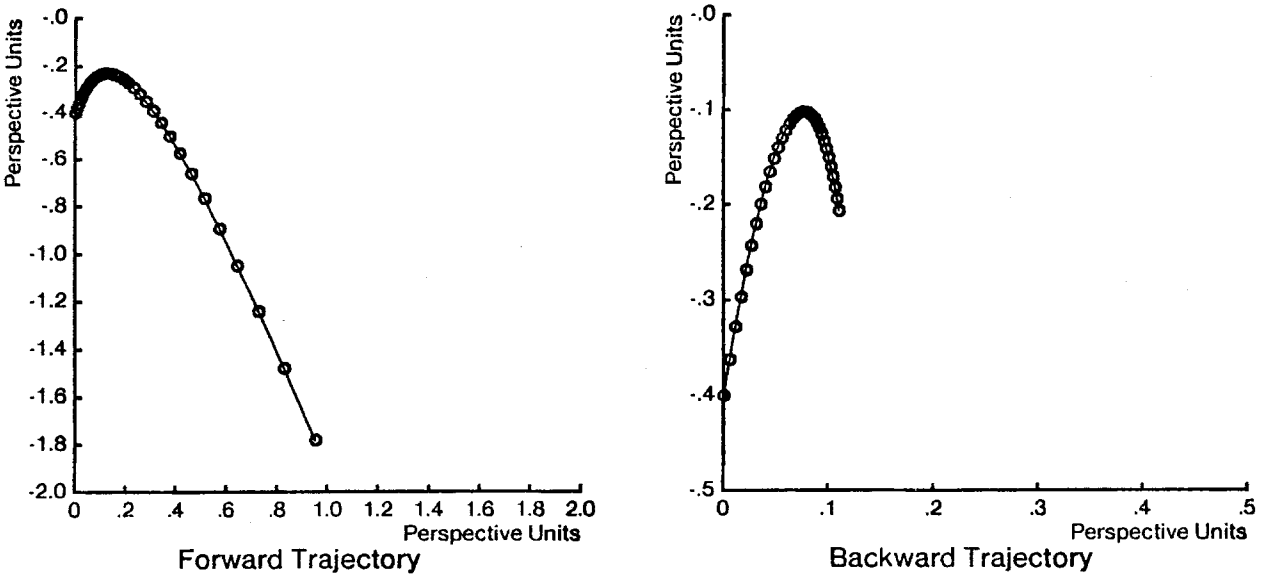

Fig. 5. The curve *on the left* is the image path generated by the forward trajectory, which begins 5 m in front of the projection plane and 2 m below the origin of the projection plane with a horizontal velocity of 1 m/s, a vertical velocity of 5 m/s, and a velocity in depth towards the image plane of 4 m/s. The curve *on the right* is the image path generated by the backward trajectory, which begins 5 m in front of the projection plane with a horizontal velocity of 1 m/s, a vertical velocity of 5 m/s, and a velocity in depth away from the image plane of 4 m/s. Note the fourfold difference in scales. Units here are dimensionless projection units

by convolving the signal with derivatives of the filter; this avoids the problem of compounding numerical errors by numerical calculation of derivatives of the filtered signal. There are a number of low-pass filters we could try. It seems that most of the observations are connected to the bandwidth of the filter and are independent of the exact shape, so we shall illustrate the filtering technique with a low-pass filter based on a truncated sinc pulse in the time domain.

In Fig. 6 we show the form of the low-pass filter used on the trajectories. In Fig. 7 we have plotted four z_0 estimate analyses for the two different trajectories with two different sizes of filters: the results on the left are for the wider filter, while the top results are for the backward trajectory. As mentioned above, the graphs analyze the results of repeating the addition of noise to the image trajectory a number of times, fifty in this case. At each point in time, the mean and standard deviation for the fifty estimates are computed. The mean, the standard devitation added and subtracted from the mean, and a single sample estimate curve are then plotted with respect to time. These give information about the ensemble statistical behavior of the estimate method in the presence of the uniformly distributed independent noise. Several effects can be seen: an effect due to the width of the filter alone, an effect due to reduction of noise by the filter, and an effect due to distortion in the signal caused by the filter.

The effect of the filter width can be seen towards the start and end of the trajectories when variances jump considerably. The transition between acceptable estimates in the middle of the time range and unacceptable performance at the edges is fairly sharp. The poor performance regions seem to be due to the velocity discontinuity in the smooth trajectory at the start and end of the trajectory. When the truncated filter touches either discontinuity, errors in the filtered signal and derivatives increase rapidly. These "bad" regions occur

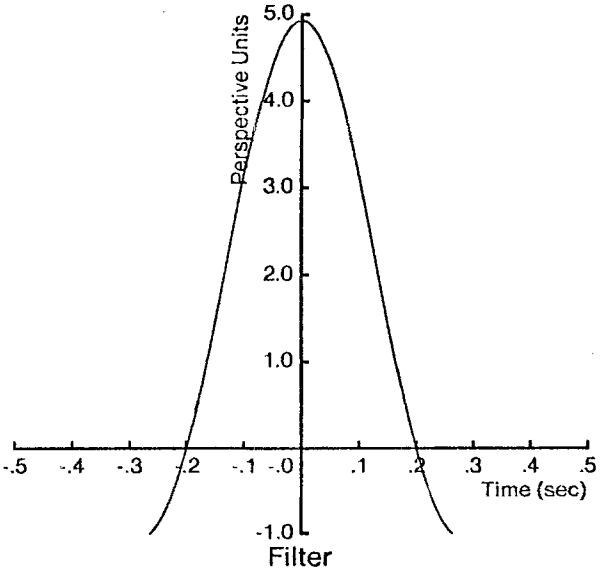

Fig. 6. The low pass filter used to smooth the noisy data. The filter is scaled to a desired width, and the derivative of the filter is convolved with the data to get first derivatives

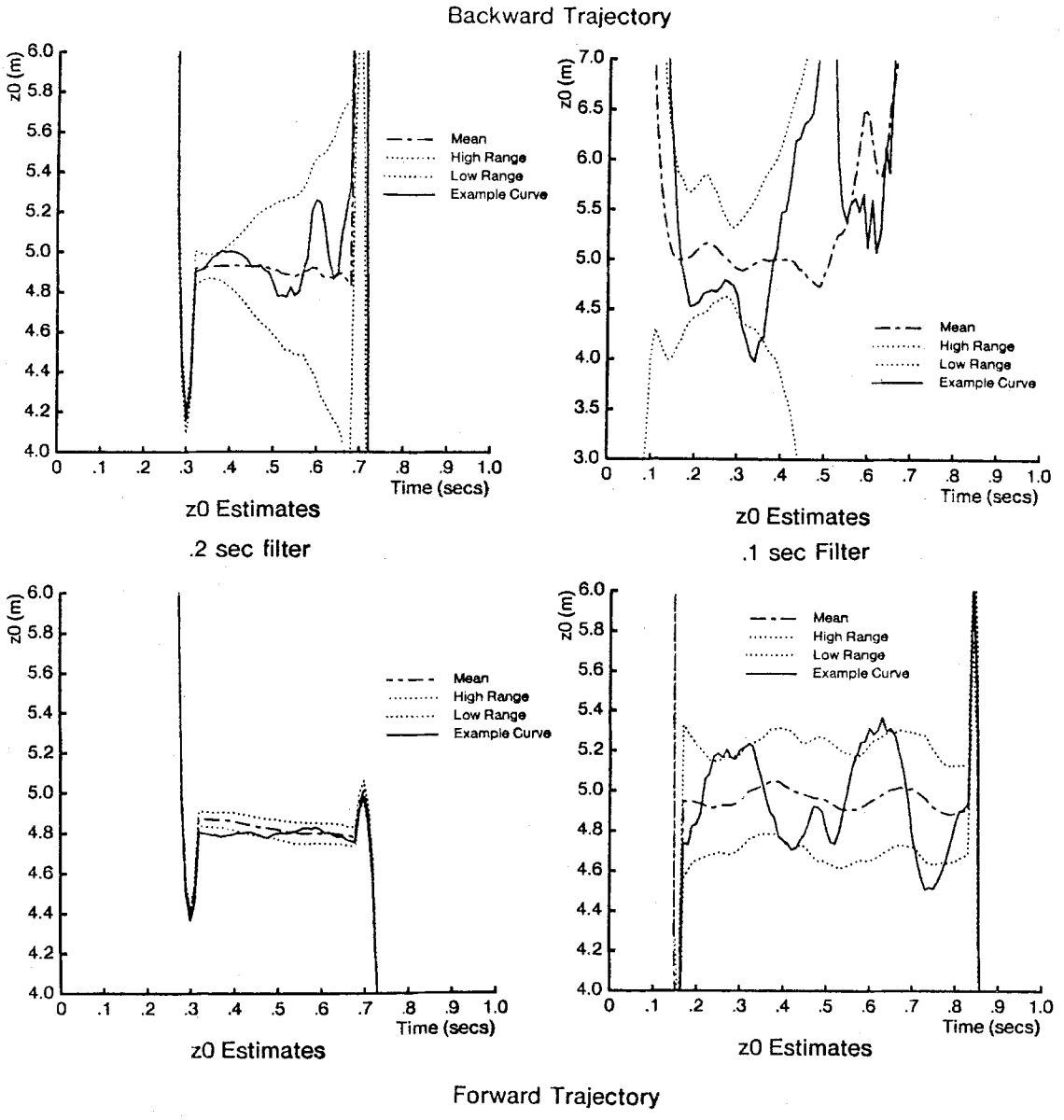

Fig. 7. Graphs of estimated z_0 versus time for filtered noisy data, repeated 50 times. The lower graphs are from the forward trajectory, the upper graphs from backward trajectory. The graphs on the left are from noisy data filtered by a 0.2 s wide filter, while the graphs on the right are from noisy data filtered by a 0.1 s wide filter. In each graph, the variance and mean over all 50 curves have been computed at each point in time. The *high range* curve is a graph of the mean plus the standard deviation; the *low range* curve is a graph of the mean minus the standard deviation. An example of an estimate curve has also been plotted. A wider filter decreases the noise, and the smaller amplitude backward trajectory, being more corrupted by the added noise, seems to have more variability, particularly as it slows down towards the end of the image trajectory

in the derivatives and the estimates of the initial conditions derived from the derivatives at about a filter's width from the start and end of the trajectory.

Reduction in noise is connected to the filter bandwidth. Wider filters (in time) remove more of the random variation in the derivatives and the estimates. With a filter width of 0.1 s, the backward trajectory estimate variance is at best 10% of the mean; the forward trajectory variance is 5%. Increasing the filter width to 0.2 s makes the variances in the backward trajectory at worst 10% and in the forward trajectory the variance is less than 2% of the mean. This indicates as expected that the more low-pass the filter, the more noise is removed from the signal. The increasing variance in the backward trajectory is probably due to the slowing of the image trajectory, making the noise relatively more important as time increases.

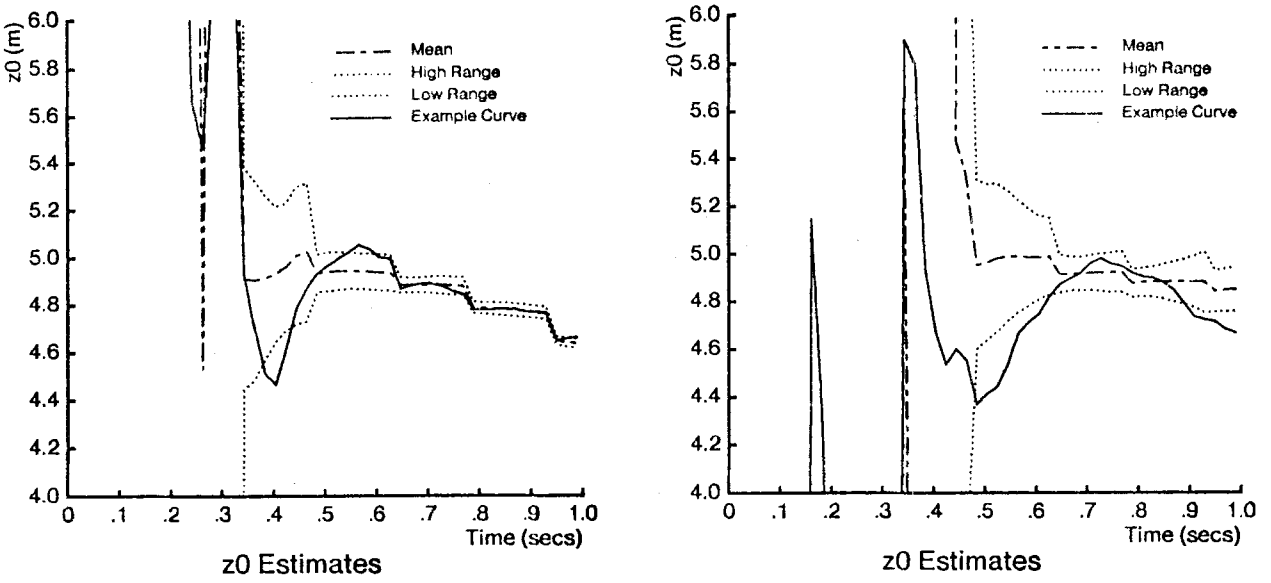

Fig. 8. Graphs of estimated z_0 versus time for filtered noisy data using the variable filter technique, repeated 50 times. The *left graph* is for the forward trajectory, the *right graph* for the backward trajectory. See Fig. 7 for explanation of the curves. Here, the filter width is increased as the trajectory develops. The variance decreases with time as the filter width increases (one can make out the individual steps between filters), and the bias on the forward trajectory increases more than that of the backward trajectory as the filter width increases. The backward trajectory estimates have a higher variance at each point in time than the forward trajectory estimates

Distortion in the filtered signal, reflected by differences between the ensemble mean and the correct value as time proceeds, is due to the bandwidth of the signal. The forward trajectory has a very quick finish, contributing significantly more high frequency energy to its image trajectory than the backwards trajectory does; thus when filtering with a wide (in time) filter to reduce the effects of noise, one is more likely to clip off significant parts of the spectrum of the signal as well, resulting in distortion of the signal and its derivatives, and hence in calculations of the initial estimates. The distortion is not extreme, but for the 0.2 s filter the bias in the estimate for the forward trajectory is of the same magnitude as the variance.

These effects create a problem: how to decide how wide to make the filter? Too wide, and the noise will be reduced, but the signal might be distorted; too narrow and the noise will dominate and ruin the estimates. This is related to a question in Torre and Poggio's regularization view of early vision problems (Poggio and Torre 1984): in the most general formulation of a regularization technique, one is left with a free parameter λ, which effectively balances between reduction in smoothness and reduction in fidelity to the data. Choosing λ becomes a significant task: how does one optimize the balance between variability in the estimates and fidelity to the original data?

We can see this problem more clearly by making use of a variable width filter to get estimates of the initial condition. At a particular time t we use the widest filter from a collection of filters that will fit inside the time span of our data, and compute estimates from the derivatives that are within the "acceptable" zone of that filter's action. As shown in Fig. 8, as time proceeds, the filters get wider, the variability in the estimates due to noise decreases, and the bias (i.e. error in the ensemble mean of the estimate) in the forward trajectory increases; bias in the backwards trajectory is much smaller, due to the trajector's narrower spectral dispersion. Note also the importance of the midpoint of the trajectory: it is just after the midpoint that the accuracy of the projections become reasonable.

Other filters we have tried include the spline filter of Poggio et al. (1985) which effectively fits a new cubic polynomial to each data point and its surrounding data points before taking derivatives, and a "polynomial" filter (Korn and Korn 1968) which provides another kind of polynomial smoothing to the data. These filters essentially behave the same way as the low pass filter discussed above.

3.2 Least Squares

In the previous section we discussed a way to use the local derivative calculation in the presence of noise. It might be argued that we are not making the best use of all the data available to us in mitigating the effects of noise by concentrating on an ultimately local calculation. In this section, we discuss an alternative method of calculating the initial conditions that can make explicit use of all the data gathered.

To do this, let us return to the idea mentioned above of taking several time points to generate equations which can be solved for the unknown initial conditions. We can try to generalize equations (2.1) by using a general power series in time to describe each coordinate of the position in three dimensions:

$$\begin{bmatrix} x(t) \\ y(t) \\ z(t) \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} a_{xi}t^i + \sum_{i=1}^{n} b_{xi}t^i \\ \sum_{i=1}^{n} a_{yi}t^i + \sum_{i=1}^{n} b_{yi}t^i \\ \sum_{i=1}^{n} a_{zi}t^i + \sum_{i=1}^{n} b_{zi}t^i \end{bmatrix}. \tag{3.1}$$

Note that we have taken a coordinate system whose z component is the projection direction; gravity can have one or more non-zero components in this system since it is not necessarily lined up with one coordinate direction. We assume we know the coefficients b_{xi}, b_{yi}, b_{zi}, the known influences, and wish to solve for the unknown coefficients a_{xi}, a_{yi}, a_{zi}. The matrix equation for the $\hat{x}$ component of (2.3) is derived as above but with more terms and becomes

$$\begin{bmatrix} \hat{x}_1 t_1^0 & \hat{x}_1 t_1^1 & \dots & \hat{x}_1 t_1^n & -t_1^0 & -t_1^1 & \dots & -t_1^n \\ \hat{x}_2 t_2^0 & \hat{x}_2 t_2^1 & \dots & \hat{x}_2 t_2^n & -t_2^0 & -t_2^1 & \dots & -t_2^n \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ \hat{x}_n t_n^0 & \hat{x}_n t_n^1 & \dots & \hat{x}_n t_n^n & -t_n^0 & -t_n^1 & \dots & -t_n^n \end{bmatrix} \begin{bmatrix} a_{z0} \\ a_{z1} \\ \vdots \\ a_{zn} \\ a_{x0} \\ a_{z1} \\ \vdots \\ a_{xn} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} (b_{xi} - \hat{x}_1 b_{zi}) t_1^i \\ \sum_{i=1}^{n} (b_{xi} - \hat{x}_2 b_{zi}) t_2^i \\ \vdots \\ \sum_{i=1}^{n} (b_{xi} - \hat{x}_n b_{zi}) t_n^i \end{bmatrix}, \tag{3.2}$$

or, more concisely,

$$\mathbf{Ta} = \mathbf{b}, \tag{3.3}$$

and we might think of inverting the large matrix on the left in order to find the coefficients of the polynomials. Unfortunately, this set of equations is not independent: we can subtract the right hand side of (3.2) from the left hand side of the equation to give:

$$\begin{bmatrix} \hat{x}_1 t_1^0 & \hat{x}_1 t_1^1 & \dots & \hat{x}_1 t_1^n & -t_1^0 & -t_1^1 & \dots & -t_1^n \\ \hat{x}_2 t_2^0 & \hat{x}_2 t_2^1 & \dots & \hat{x}_2 t_2^n & -t_2^0 & -t_2^1 & \dots & -t_2^n \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \hat{x}_n t_n^0 & \hat{x}_n t_n^1 & \dots & \hat{x}_n t_n^n & -t_n^0 & -t_n^1 & \dots & -t_n^n \end{bmatrix} \begin{bmatrix} a_{z0} + b_{z0} \\ a_{z1} + b_{z1} \\ \vdots \\ a_{zn} + b_{zn} \\ a_{x0} + b_{x0} \\ a_{x1} + b_{x1} \\ \vdots \\ a_{xn} + b_{xn} \end{bmatrix} = 0.$$

More compactly, we have effectively found an $\hat{\mathbf{a}}$ such that

$$\mathbf{T}\hat{\mathbf{a}} = \mathbf{0}, \tag{3.5}$$

and we know that $\mathbf{T}$ must then be singular if $\hat{\mathbf{a}} \neq 0$. We have gotten into this trouble because we have allowed too much flexibility in the a_{xi} and a_{zi}. If we insist that one of the a_{xi} (or one of the a_{zi}) be equal to zero, remove its corresponding column from $\mathbf{T}$, and reduce the number of rows by one to keep $\mathbf{T}$ square, then we cannot get the above simple solution to the homogeneous equation, and the new $\mathbf{T}$ matrix seems much more likely to be non-singular. In our situation, we assume we know the components of gravity in the different directions; if we ignore second order frictional effects, it makes sense to insist that gravity be the most important second order component of either the x or the z behavior in time, which suggests setting at least one of a_{x2} or a_{z2} to 0. (If gravity does not have a component in either the z or the x directions, it must have a non-zero component in the y direction, and we can rewrite the above equations using y instead of x.) In principle, we can then invert the matrix $\mathbf{T}$, and solve for the remaining parameters. In practice, we can take far more data points than is theoretically required, and use a Moore-Penrose inverse scheme to obtain a least squares solution $\hat{\mathbf{a}}$ for the unknown parameters in order to reduce the effects of noise on the data:

$$\hat{\mathbf{a}} = -(\mathbf{T}^t\mathbf{T})^{-1}\mathbf{T}^t\mathbf{b}, \tag{3.6}$$

where, if we assume we set $a_{x2} = 0$, (as in the case of a projection plane nearly parallel to gravity)

$$\mathbf{T} = \begin{bmatrix} \hat{x}_1 t_1^0 & \hat{x}_1 t_1^1 & \dots & \hat{x}_1 t_1^n & -t_1^0 & -t_1^1 & -t_1^3 & \dots & -t_1^n \\ \hat{x}_2 t_2^0 & \hat{x}_2 t_2^1 & \dots & \hat{x}_2 t_2^n & -t_2^0 & -t_2^1 & -t_2^3 & \dots & -t_2^n \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \hat{x}_m t_m^0 & \hat{x}_m t_m^1 & \dots & \hat{x}_m t_m^n & -t_m^0 & t_m^1 & -t_m^3 & \dots & -t_m^n \end{bmatrix}$$

$$\mathbf{a} = \begin{bmatrix} a_{z0} \\ a_{z1} \\ \vdots \\ a_{zn} \\ a_{x0} \\ a_{x1} \\ a_{x3} \\ \vdots \\ a_{xn} \end{bmatrix} \tag{3.7}$$

$$\mathbf{b} = \begin{bmatrix} \sum_{i=1}^{n} b_{xi}t_1^i \\ \sum_{i=1}^{n} b_{xi}t_2^i \\ \vdots \\ \sum_{i=1}^{n} b_{xi}t_m^i \end{bmatrix},$$

where m is the number of points used.

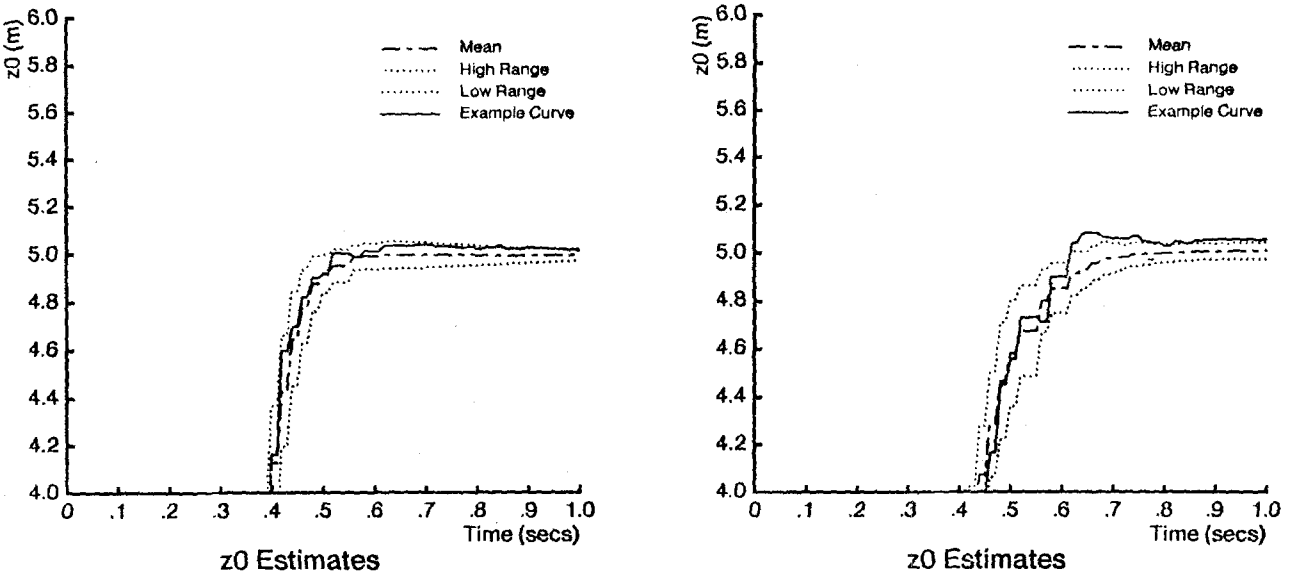

Fig. 9. Graphs of estimated z_0 versus time for least squares technique applied to noisy data, repeated 50 times. The *left graph* is for the forward trajectory, the *right graph* for the backward trajectory. The curves are described in Fig. 7. Each curve is calculated using the least squares technique applied to the noisy trajectory seen up to that time

This method appears to be expandable to accomodate varying numbers of parameters in the trajectory model (e.g. to account for effects of friction, etc.). The local derivative techniques discussed earlier can also be expanded to cover more unknown parameters by taking higher time derivatives of the power series, eventually knocking out all the a_{xi} coefficients, and allowing a matrix solution for the unknown a_{zi}. With the derivative methods, however, we have to assume a non-zero value for b_{xk}, where k is the highest derivative we take; otherwise our equation again becomes of the form $\mathbf{Ta}=\mathbf{0}$, and again we will have a singular $\mathbf{T}$; using power series without derivatives does not require us to provide a non-zero b_{xk}, $k>2$, only a non-zero b_{x2}.

We can apply the direct least squares technique to the forward and backward trajectories. In our particular situation, we assume we are dealing only with quadratic polynomials; we also know gravity has no components in the viewing direction, so the viewing direction polynomial is linear. We can apply the direct least squares technique to partially complete trajectories: using every other point until a particular time t, we can generate estimates of the initial conditions for that time. The results of applying this technique to 50 noisy copies of the trajectory data are shown in Fig. 9. The estimates improve with increasing time as expected and as did the variable filter technique discussed earlier, with the biggest improvement coming after the peak in the image trajectory. In addition, there is no noticeable bias in the forward trajectory estimates. In Fig. 10 we show examples of the convergence of the calculation of V_z for different values of V_z. We also show the result of applying the direct least squares technique to the synthesized trajectories without noise; as one can see, only the first few tenths of a second are corrupted by numerical errors; the bulk of the errors encountered in the noisy situation are due to the noise itself, not to round-off errors.

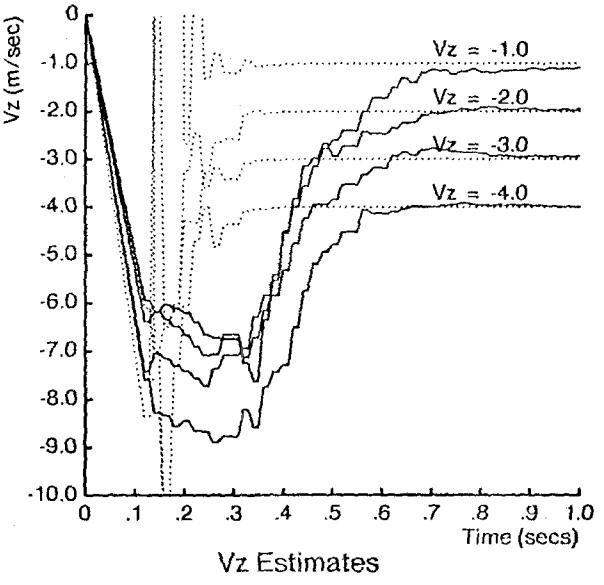

Fig. 10. Graphs of estimated V_z versus time for the least squares technique applied to noisy data. Here the forward trajectory is used, but the velocity in depth is modified. For each velocity in depth (-1.0, -2.0, -3.0, -4.0 m/s) estimates of V_z versus time are generated both for the noiseless version (*dotted curve*) and the noisy version (*solid curve*) of the trajectory using the least squares technique. By looking at the noiseless convergence, one sees that numerical problems account for only a very small part of the convergence time in the noisy situation

It is interesting to speculate on the "meaning" of the least squares calculation: why does it work so well? One observation is that to derive the method, we multiplied the image coordinates, which are corrupted by the noise, by the actual depth. In a sense, we are biasing the noise on the image by the estimated depth of the object in the image: if the object is far away, noise is taken quite seriously into account; if the object is very close, the noise is discounted. This fits well with our imaging scheme: when the object is far away, noise on the image corresponds to large variations in absolute position in three-dimensions around that original depth; similarly, when an object is close at hand, noise in the image corresponds to relatively small absolute displacements of the object in three dimensions.

This least-squares method seems to work reliably and robustly in the examples we have used. It is not perfect: if we attempt to use more than every other sample of data on the forward and backward trajectories, we end up with poor V_z estimates – due undoubtably to numerical problems in multiplying out two nearly redundant matrices. We have not explored modifications of the least squares method to avoid this – the performance as is very good. We will compare the tracking filter method with the continuous least squares method below when we look at synthesized and real video images of a moving ball as opposed to the simulated trajectories and noise we have used so far. There is one other technique we will discuss first for minimizing the effect of noise on the calculation of the initial conditions: optimal signal processing.

3.3 Optimal Parameter Estimation

Since we are trying to get an estimate of a parameter which controls a signal in a known way from a perhaps corrupted time image of that signal, another natural method of solution makes use of optimal signal processing theory (e.g. Van Trees 1968). We assume our signal is a discrete signal $s(k, \mathbf{A})$, where k is the "frame number" of the image projection in the sequence of image projections making up the trajectory data and $\mathbf{A}$ is a vector of initial conditions. This signal is then corrupted by independently distributed noise $w(k)$ such that $E[w(k)w(j)] = \sigma^2 \delta_{kj}$, to give a received signal

$$r(k) = s(k, \mathbf{A}) + w(k). \tag{3.8}$$

We can consider the discrete function $r(k)$ as a random vector $\mathbf{r}$ of n components, where n is the number of samples of the functions we are looking at. If we also assume that $w(k)$ is a normal process, then

$$\mathbf{r} \sim N(\mathbf{s}(\mathbf{A}), \sigma^2 \mathbf{I}), \tag{3.9}$$

that is, $\mathbf{r}$ has a multivariate normal distribution with mean vector $\mathbf{s}(\mathbf{A})$ and variance matrix $\sigma^2\mathbf{I}$, where $\mathbf{s}(\mathbf{A})$ is the vector of samples $\mathbf{s}(k, \mathbf{A})$, and $\mathbf{I}$ is the identity matrix. The maximum likelihood estimator of $\mathbf{A}$ is given by looking at:

$$\min_{A} [\|\mathbf{r} - \mathbf{s}(\mathbf{A})\|^2] = \min_{A} \left[\sum_{i=1}^{m} (r(k) - s(k, \mathbf{A}))^2 \right]; \tag{3.10}$$

the minimizing $\hat{\mathbf{A}}$ is the estimate of $\mathbf{A}$. By an application of the Cramer-Rao inequality, we can get an inexact lower bound on the variance of this estimate (inexact because our maximum likelihood estimator here is likely to be biased; as the number of points increases, however, the bound will be approached):

$$\mathrm{Var}[\hat{\mathbf{A}} - \mathbf{A}] \gtrsim \frac{\sigma^2}{\left\| \frac{d}{d\mathbf{A}} s(\mathbf{A}) \right\|^2}. \tag{3.11}$$

This lower bound is not likely to be achieved except in the case that $s(k, \mathbf{A})$ depends linearly on $\mathbf{A}$; this is clearly not the case in our situation. It does suggest, however, that the variance of the estimate can be quite large if $s(\mathbf{A})$ does not change magnitude much with changes in $\mathbf{A}$. We might expect this problem to arise in our case because of the following near duplication of trajectories: given one trajectory beginning at $\mathbf{x}_0$ on the ground and ending at $\mathbf{x}_f$, we can move the starting point towards the observer so that the new $\mathbf{x}_0$ projects to the same point on the image plane as the old one did; we can arrange for the trajectory to pass through the same highest point (and hence take the same amount of time) as the old trajectory, and we can arrange to have it finish its trajectory at a point which projects to the same image point as the old final point. The image trajectory will still be a convex function without ripples, will have the same starting and ending points, nearly the same high points, and will take the same amount of time as the original trajectory. The only possible differences will be subtle ones in the curvature and a slight shift in the projected peak, so we might expect the variance of estimates to be quite high. In the real world, however, other constraints might be brought to bear to limit this problem: we are typically some distance above the ground, for example, so that a trajectory that is assumed to start from near ground level might not have a start or finish that is either below ground level or much above it; thus, the matches for the image points described above might be excluded on "physical grounds."

To implement the optimal signal processing technique we are interested in a minimization over a six dimensional space: we have a six dimensional surface defined by the function to be minimized (the "index") and the six different initial condition parameters which are arguments of the index. With the projection plane

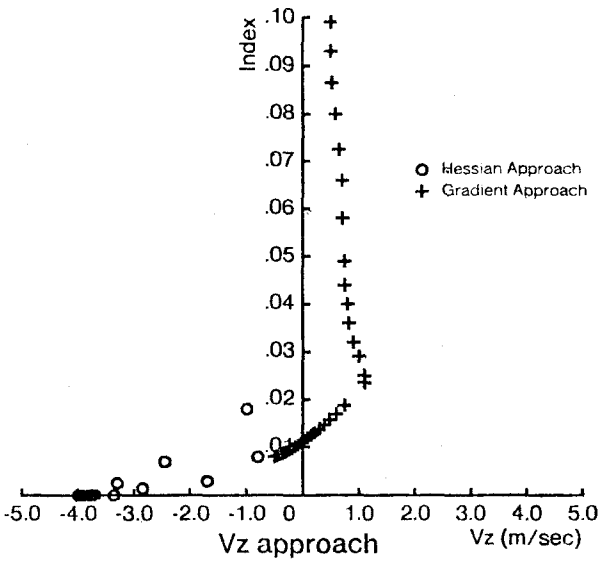

Fig. 11. This shows how a combination of techniques for finding an optimal solution converges on an estimate for V_z on the backward trajectory corrupted by noise. The horizontal axis gives V_z values, while the vertical axis gives index values. The algorithm's estimates step in from the top of the graph. *The plus signs* indicate the region where a pure gradient approach is being used: it converges quite rapidly initially, but at about index level 0.02, the approach slows down drastically. At this point an approximate Hessian technique is called into play (*the circles*), under the assumption that slowing down means the solution is nearby. The Hessian approximation technique constructs an approximation to the second derivative matrix, and locates the minimum of the surface as if it were a quadratic surface. As one can see, the final approach is relatively rapid. Unfortunately, this combination of techniques frequently diverges in other examples

parallel to gravity, we can reduce the difficulty of the search somewhat by separating the image coordinates: we can use the $\hat{x}$ coordinates to find the initial conditions x_0, V_x, z_0, V_z, reducing the problem to looking for the lowest point on a four-dimensional surface, which turns out to be very flat in one direction and very curved in another at the correct point on the surface.

There are a large number of methods available for finding the minima of a function over a surface; however, all of them have troubles with complicated or extreme surfaces: they either get stuck in local minima, an endemic problem with local calculation techniques; or they have accuracy problems in actually doing the computations necessary; or they bounce around a cycle of points forever; or they take forever. In Fig. 11 the graph points labeled as "gradient approach" show a typical sequence of iterations of a gradient search technique, which starts from an initial estimate and moves along the gradient to find a minimum. As can be seen, an initial fast approach to an optimal solution (moving downwards in this figure) is followed by an extremely slow approach due to the flatness of the bottom of the surface.

One can augment the gradient search technique with an approximate Hessian technique which is related to Newton-Rapheson approaches. Newton-Rapheson techniques create quadratic models of the surface being searched using a second-derivative matrix (the Hessian) to approximate the quadratic; the approximate Hessian technique attempted here creates estimated of the Hessian as the algorithm proceeds rather than explicitly calculating the second derivative matrix (which seems very error prone in this situation). This class of techniques works best with a good initial estimate of the minimum point. One starts with a gradient approach, and continues until the changes in the index become too small; one then shifts to the approximate Hessian technique, on the assumption that one is close to the minimum point and hence the surface can be approximated by a quadratic surface. The difficulty is that the surface is very much like a flat valley with steep sides, hence the Hessian has both small and large components, and the inverse of the Hessian used in Newton-Rapheson style algorithms becomes unstable, while the updated estimates run the risk of being nonsensical. In Fig. 11 we show what a typical successful use of this combination might look like: one can identify the region of fast gradient approach followed by a region of slow gradient approach until a certain threshold of lack of progress is reached; finally, a fast convergence phase to a final solution quite close to the actual value. Unsuccessful searches typically explode out from the transition to the approximate Hessian technique, never to be seen again. It is conceivable that by adding constraints to the initial condition estimates (e.g. don't allow estimates for z_0 of over 10^6 m), and restarting searches that go awry from a new, randomly determined location, one could eventually converge on a nearly correct solution every time; we have not proceeded with this.

3.4 Discussion

The calculations using a fixed width filter led either to large variance if the filter was too small to remove significant noise, or to large distortion if the filter was too wide to spare the signal. The optimal signal processing technique, although in conception the tidiest method, in execution seems to be quite difficult to realize stably; it is conceivable that there is a search algorithm that can be tailored to work best on this kind of problem, but we did not explore this further.

It would appear that the least squares technique is the best of the techniques we have looked at so far. Both it and the variable filter technique are relatively stable and accurate in the presence of noise; in addition, the least squares techniques does not suffer

from bias on trajectories directed towards the observer. It is conceivable that statistical techniques applied to the individual estimates from the variable filter technique could reduce the bias by identifying a shift in the mean of the estimates using some sort of moving average. The bias is actually quite small, and we shall see in the next section that in "real" situations with fewer images, the variance is large enough to swamp the bias, and these two techniques are nearly equivalent.

4 Experiments with Images

We looked at how the two more successful techniques for estimating the initial conditions of a free fall trajectory fare when confronted with image data, as opposed to simulated trajectories. We looked at both simulated images and digitizations of a videotape of a high contrast ball in a parabolic trajectory. The center of the ball was located by finding several edges of the ball and drawing perpendicular bisectors to the chords connecting these edge points. The bisectors intersect at the center of the circle. When we used a computer simulated ball image (a black disk on a white background), the main source of error in locating the ball's center was the quantization error for the location of the ball's edges. When we used a real videotape of a ball's flight, we had additional errors introduced by the non-spherical nature of the ball, distortions from the optics, time-smearing, and problems with defining the actual edge of the projected ball.

4.1 Simulated Images

We first generated a series of images at 0.03 s intervals consisting of a black disk of appropriate radius against a lighter background. The radius and position of the disk were determined by the trajectory and the known projection parameters. An array of these images (a "film") was then fed into another routine which estimated the location of the center of each disk in each image frame in the way suggested above: a chord was drawn through two edges of the disk, either horizontally or vertically. This chord was bisected, and another chord was drawn as the perpendicular bisector; finally, this last chord was bisected, and the perpendicular bisector drawn. The intersection of these last two chords was taken as the center of the circle. In principle, if the image were exactly a disk only two cords needed to be drawn; however, if one started close to a horizontal or vertical edge of the (possibly distorted) disk, the first chord one drew might suffer badly from the noise in a real video image at the edges of the disk. After the center of the ball was located, each center position in pixels was converted to projection coordinates and placed in an array. Finally, this array was processed to remove any initial images showing the ball as stationary, and an array consisting of the projected moving centers of the disks was given to the initial condition estimation routines.

Estimates for z_0 and V_z derived from the variable filter method and the least squares method are shown in Fig. 12. The backward trajectory estimates do follow the simulation results shown in Figs. 8 and 9 for the variable filter technique and the least squares technique respectively: both techniques converge to nearly the correct initial condition for both z_0 and V_z. Convergence for the least squares solution appears to be slightly faster than for the variable filter solution, particularly for estimates of V_z.

The forward trajectory results using the variable filter technique also agree with the simulation results. In this case, the trajectory cuts off at 0.8 s, so we are not able to see the bias develop as strongly as in Fig. 8, but there is a hint of a downward trend to the z_0 estimate curve. There is good convergence to estimates near the correct values.

The forward trajectory results with the least squares technique are more at odds with the simulation results. Instead of performance similar to that for the backward trajectory, we have a considerable bias demonstrated in the final estimate for z_0, although the bias is decreasing as time increases. The estimate for V_z has clearly not yet converged by the 0.8 s cut-off but is heading in the correct direction. Since z_0 and V_z are estimated simultaneously, both the z_0 estimate and the V_z estimate are most likely aflicted with the same problem: not enough independent data has been accumulated by the time 0.8 s is reached, given the noise created by the pixelization and the center location algorithm used. The estimates are off by less than 10%, so useful information is still produced.

Well before the ball actually lands, one has good information about the initial conditions of the trajectory from both techniques and can generate information about the general location of the landing site. As more data is received and the ball gets closer to the actual landing site, the estimates improve. With simulated images, the variable filter technique gets within 5% of the actual value for z_0 0.4 s before the end of the data: the least squares technique performs about the same on the backward trajectory and is still within 10% of the correct z_0 0.4 s before the end of the data. In all cases, it appears to be more difficult to get accurate estimates for V_z than z_0, consistent with the general difficulty of getting accurate derivatives from data.

4.2 Videotaped Images

We took videotapes of a thrown ball with a Hitachi solid state video camera and a JVC three-quarter inch

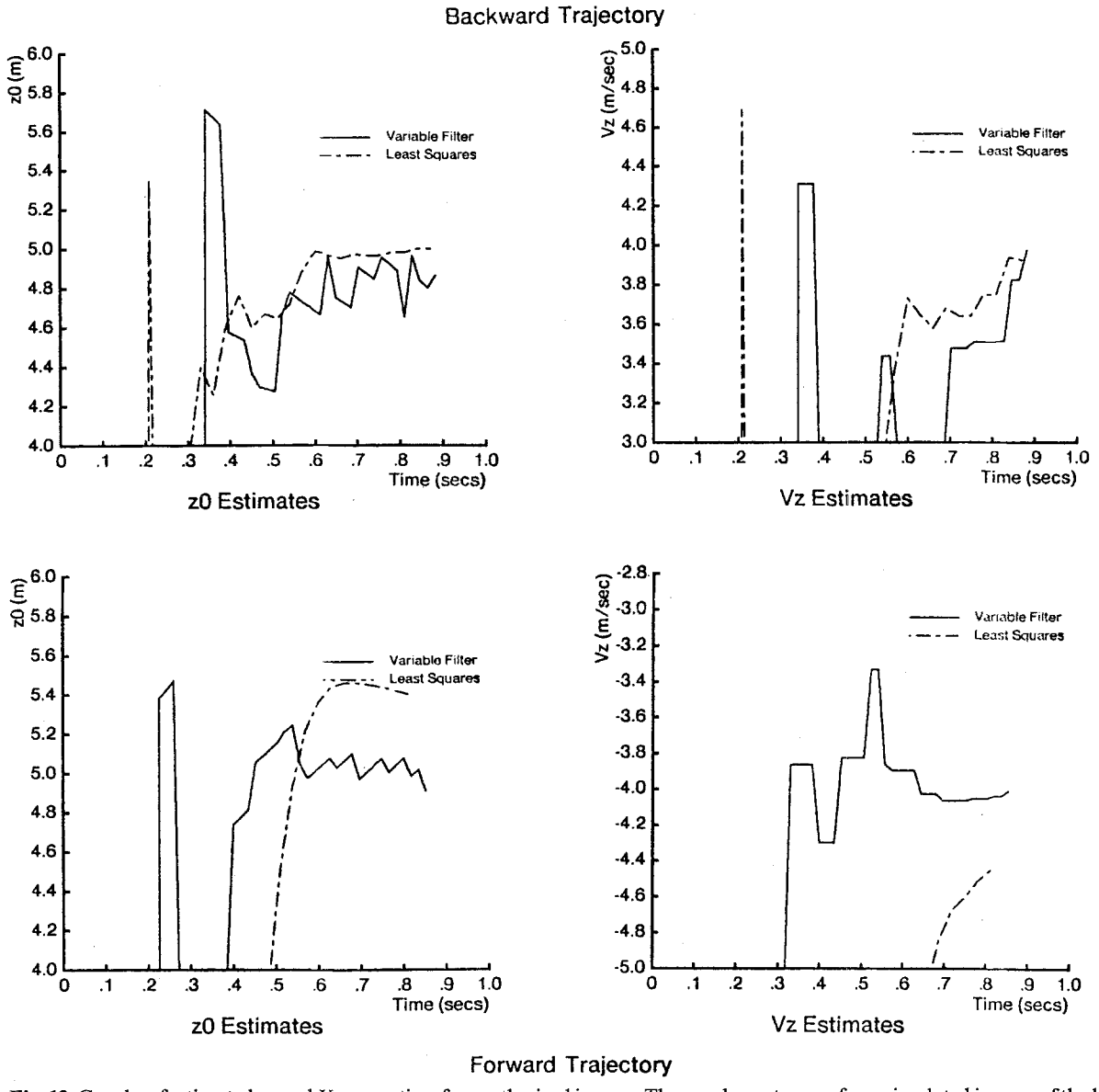

Fig. 12. Graphs of estimated z_0 and V_z versus time for synthesized images. The graphs on top are from simulated images of the backward trajectory; on the bottom from simulated images of the forward trajectory. Each graph shows both the variable filter technique estimates and the least squares technique estimates

videotape machine. The experimenter stood about 1.5 m from the video camera, which was equipped with a 28 mm lens, and threw the ball backward with a velocity of around 3 m/s away from the camera and 1 m/s horizontal to the camera. Care was taken to try to maximize contrast between the black ball and the light surroundings. A flash was used to signal the start of a toss on the videotape. The videotape was played back into a frame grabber which digitized the video signal and provided an MIT Lisp Machine with a continuously updated memory image of the video signal. The video image appears to the Lisp Machine as an array of 8 bit numbers with dimensions 576 by 454; each number is the grey-level for that pixel. To analyze the videotape, we needed to store the entire sequence of image frames in the Lisp Machine, since the routines used in previous sections were not quite fast enough to analyze the incoming video information in real time. Since the Lisp Machine could not copy the video information into other parts of its memory as fast as it came in, the video tape was played through once for each frame – the flash on the videotape provided a constant reference point in time. Once 15 or 20 images were stored in the machine, they were transferred via a central file server to a Symbolics 3600 Lisp Machine which actually analyzed the frames in almost the same

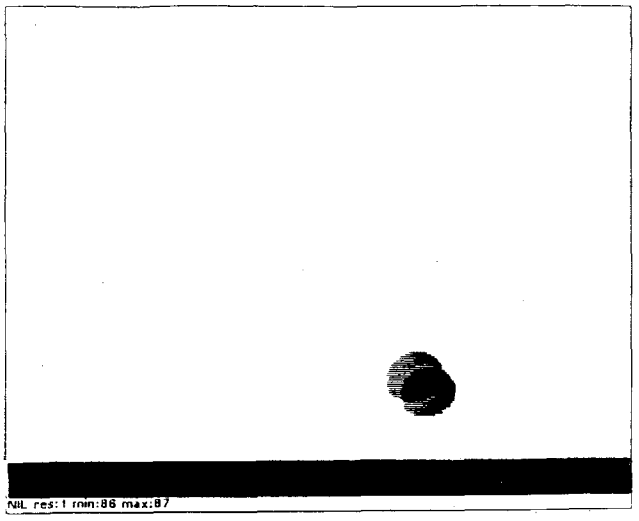

Fig. 13. Thresholded frame from a video tape of a free-falling ball. Interlacing of the image effectively creates two overlaid images of the ball – one on every other line of the image

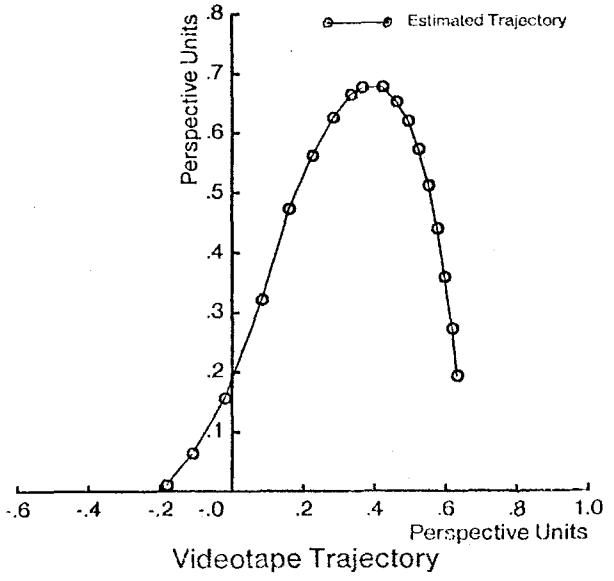

Fig. 14. Graph of image trajectory on the image plane from the videotape. The points in this trajectory were derived from the videotape used for Fig. 13

way that the simulated images were analyzed above. To eliminate interlacing, we used every other pixel instead of every pixel in the image.

Figure 13 shows a threshold version of the videotype frame: almost all the extraneous information is removed. Note how interlacing creates two overlaid images of the ball. After analyzing the frames, the resultant trajectory is shown in Fig. 14. Finally, in Fig. 15, we show the estimates given by the two different techniques for initial condition parameters applied to the convex portion of the trajectory (the free-fall portion of the trajectory). As one can see, both techniques approach the same z_0 and V_z values, although the last squares technique lags behind the variable filter technique in the estimate of V_z by about 0.2 s.

There are a number of differences between the simulated images sequences and the videotaped images. We were not able to videotape large scale or fast trajectories: the video cameras we used had a relatively limited field of view, and space constraints came into play. Noise on the videotape images appears quite comparable to noise in the synthesized images: by thresholding the image, we can essentially generate the same kind of black disk on white background we used in the simulations. With the very small number of points here (18), it seems the least-squares technique requires more time to effectively estimate the depth parameters; on the other hand, one could still see convergence of both techniques to reasonable answers with only a few points.

5 Conclusions

It seems to be possible to take a sequence of noisy images of the trajectory of a ball in free fall and calculate from it an estimate of the initial conditions based solely on the trajectory of the center of the ball. From the simulations, it seems that backward trajectories may be more sensitive to extraneous frames than forward trajectories because of the digitization problem which tends to correlate errors in the points on the image plane, but both techniques can effectively estimate initial conditions. The limited videotape testing we tried also suggested that it was possible to get useful information about the initial conditions of a ball's flight from the projection of the center of the ball.

Having seen that a computer can perform the task of figuring out the parameters of a free fall trajectory from the two-dimensional projection of the trajectory on an image plane, we wondered how humans might fare in a similar task – do we also make use of the information contained in the projection of the center of the ball? In an experiment reported elsewhere (Saxberg 1985) we built a videogame to find out what information is important in solving this kind of task for a human being. These results suggest strongly that humans do not rely on the center of mass trajectory to make estimates about the trajectory of a ball; rather, the changing size of the image of the ball seems to be the primary clue for determining object trajectory. This makes some sense: a human being has to cope with a moving visual system and with imperfections in trajectories due to air resistance, self-propulsuion, etc. The changing size of an object on the retina is likely to be a more stable cue than attempting to reconstruct the

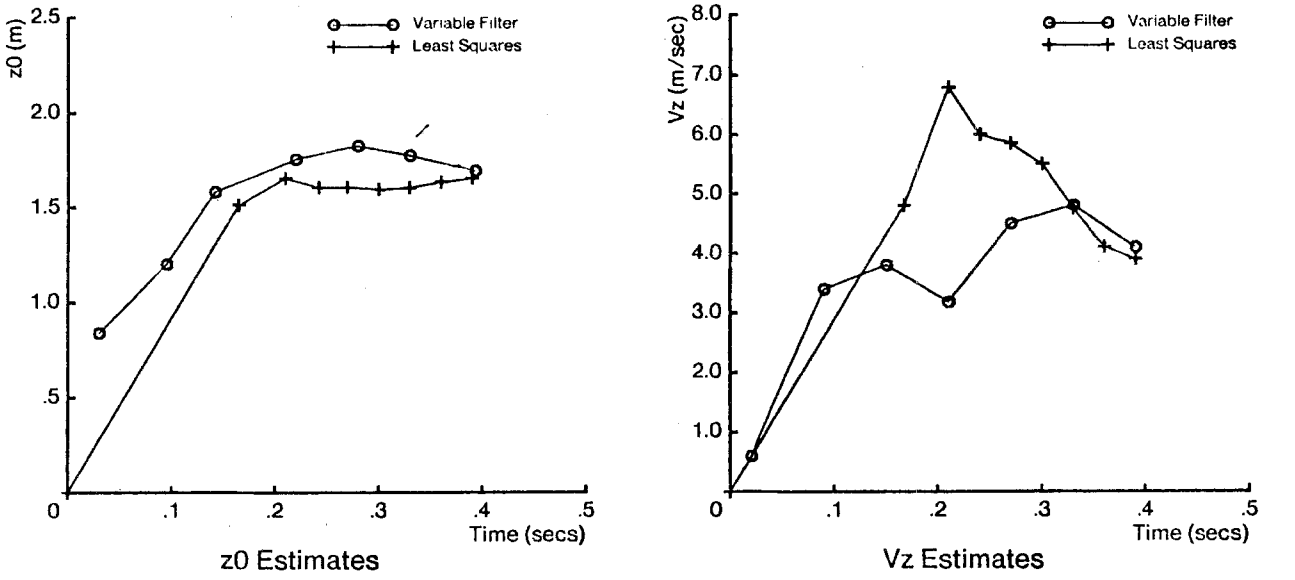

Fig. 15. Graphs of estimated z_0 and V_z versus time using videotape data (Fig. 14). The graph *on the left* is for z_0: *on the right* for V_z. Each graph shows both the variable filter technique estimates and the least squares technique estimates

projected trajectory on a theoretically stationary projection surface.

Acknowledgements. My thanks to Dr. Tomaso Poggio for his suggestions and support. This work has been supported in part by a grant to T. Poggio from the Office of Naval Research, Engineering Division; by the Advanced Research Project Agency of the Department of Defence under Office of Naval Research contract N00014-80-C-0505; by Public Health Service National Research Service Award 5 T 32 GM 07753-06; and by a fellowship from IBM.

References

Hildreth EC (1983) The computation of the velocity field. MIT AI Memo 734

Hoffman DD (1980) Inferring shape from motion fields. MIT AI Memo 592

Hoffman DD, Flinchbaugh BE (1982) The interpretation of biological motion. Biol Cybern 42:195–204

Korn GA, Korn TM (1968) Mathematical handbook for scientists and engineers. McGraw-Hill, New York, p 771

Longuet-Higgins HC (1981) A computer algorithm for reconstructing a scene from two projections. Nature 293:133–135

Longuet-Higgins HC, Prazdny K (1980) The interpretation of a moving retinal image. Proc R Soc Lond (Biol) 208:385–397

Marr D (1982) Vision. Freeman, San Francisco

Poggio T, Torre V (1984) Ill-posed problems and regularization analysis in early vision. MIT AI Memo 773

Poggio T, Voorhees H, Yuille A (1985) A regularized solution to edge detection. MIT AI Memo 833

Prazdny K (1980) Egomotion and relative depth map from optical flow. Biol Cybern 36:87–102

Saxberg BVH (1987) Projected free fall trajectories. II Human Experiments Biol Cybern 56:177–184

Todd JT (1981) Visual information about moving objects. J Exp Psych 7:795–810

Tsai RY, Huang TS (1984) Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces. IEEE Trans Patt Anal Mach Intel 6:13–27

Ullman S (1979) The interpretation of visual motion. MIT Press, Cambridge, MA

Van Trees HL (1968) Detection, estimation, and modulation theory. Wiley, New York

Wallach H, O'Connell D (1953) The kinetic depth effect. J Exp Psych 45:205–217

Waxman AM, Wohn K (1984) Contour evolution, neighborhood deformation and global image flow: planar surfaces in motion. Univ M Cent Autom Res TR 58

Webb J, Aggarwaal JK (1981) visually interpreting the motion of objects in space. Computer 14:40–46

Received: September 11, 1986

Bror V. H. Saxberg
M.I.T. Artificial Intelligence Laboratory
545 Technology Square
Cambridge, MA 02139
USA