Applying Temporal Constraints to the Dynamic Stereo Problem

MICHAEL JENKIN and JOHN K. TSOTSOS*

Department of Computer Science, University of Toronto, Toronto, Ontario M5S 1A4, Canada

Received December 15, 1984; revised April 16, 1985; accepted June 14, 1985

An algorithm is presented for the problem of the stereopsis of time-varying images (the dynamic stereo problem). Dynamic stereopsis is the integration of two problems: static stereopsis and temporal correspondence. Rather than finding the intersection of these problems to be more difficult, it was found that by solving the two problem simultaneously, and thus incorporating the spatio-temporal context within which a scene exists, some of the hard subproblems belonging to stereopsis and temporal correspondence could be avoided. The algorithm relies on a general smoothness assumption to assign both disparity and temporal matches. A simple model of the motion of three-dimensional features is used to guide the matching process and to identify conditional matches which violate a general smoothness assumption. A spatial proximity rule is used to further restrict possible matches. The algorithm has been tested on both synthetic and real input sequences. Input sequences were chosen from three-dimensional moving light displays and from "real" grey-level digitized images.

INTRODUCTION

A basic problem in any motion understanding vision system is that of determining a temporal correspondence between objects. Independent of the domain of application, be it human body motion [14], Hamburg street scenes [2], or human hearts [17], in order to analyze the motion of the objects it is first necessary to determine the position of objects in the scene as a function of time. In a perspective projection of the world such a correspondence can be difficult to obtain unless some assumption is made as to either the type of object being observed (such as being rigid or being composed of jointed rigid parts [19]) or the type of motion the object can exhibit [18]. The problem arises because in a perspective view true object location is not readily available. The human visual system overcomes this problem in part by using two images of the scene rather than just one. By combining the two views depth information can be obtained and thus accurate positional information is available. Such information would simplify the problem of determining temporal correspondence.

Current approaches to the design of stereopsis algorithms have had to face two critical design choices: the choice of monocular point descriptions that are to be used, and the choice of mechanisms that are to be used to resolve any ambiguity within the population of possible local matches. When the stereopsis problem is expanded to include time-varying images, the algorithm must also deal with the problem of tracking the monocular point descriptions, or the three-dimensional descriptions which they represent through time. In addition a choice must be made as to the order of processing. Temporal matching could be performed before, after, or simultaneously with stereopsis.

If stereopsis is performed after temporal matching, then temporal matching must take place in a two-dimensional perspective view of the world. Without some motion

*Fellow, Canadian Institute for Advanced Research.

0734-189X/86 $3.00
Copyright © 1986 by Academic Press, Inc.
All rights of reproduction in any form reserved.
or object assumptions, rotation in depth and plastic deformation would pose serious
difficulties to such a scheme. Alternatively, if stereopsis takes place before temporal
matching the temporal matching is easier as it can be performed in real world
coordinates. Unfortunately the stereopsis problem would then reduce to that of
solving static stereoscopic images. From the work of Marr and Poggio [11] and
Mayhew and Frisby [12] we know that even when sophisticated monocular primit-
tives such as zero-crossings or zero-crossings and peaks are available, the problem of
static stereopsis is very difficult, and usually cannot be solved without some global
mechanism to assign final disparities. Performing stereopsis simultaneously with
temporal matching seems potentially best as the temporal matching can be per-
formed in real world coordinates (thus avoiding the problem of perspective match-
ing) and motion information can be used to further limit the possible binocular
matchings.

An algorithm using this approach has been previously described [8]. This ap-
proach made certain simplifying assumptions about the nature of the input, which
would be difficult to overcome in order to apply the algorithm to real image data.
The algorithm assumed that there was no occlusion, that all points were visible in all
frames, and that all real world positions were known at some initial time. We base
our algorithm on the same underlying principles as Jenkin [8] but present one more
suitable to the analysis of real world images. In particular we deal with the problem
of occlusion; a feature visible by one eye may not be visible by the other, or points
may not be visible to either eye at all.

PSYCHOLOGICAL BASIS

We based our algorithm on the assumption that the motion of an object being
viewed would obey a general smoothness assumption.

The smoothness assumption was based upon certain results from psychology. The
principle of least action [16] suggests that when we perceive an object as moving we
tend to perceive it as moving along a path that in some sense is the shortest,
simplest, or most direct. Ramachandran and Anstis' principle of "visual inertia" [15]
suggests that when any object moves in one direction at uniform velocity we tend to
perceive it as continuing its motion in that direction. These perceptual tendencies
formed the basis of our smoothness assumption.

The temporal stereopsis algorithm follows the basic smoothness assumption; changes in feature characteristics will be small and continuous. In particular we
assume that when observing the motion of an object (one that does not disappear
nor is created), the object will be relatively unchanged from one frame to the next:

(1) The location of a given feature would be relatively unchanged from one
frame to the next.

(2) The three-dimensional vector velocity of a given feature would be relatively
unchanged from one frame to the next.

(3) The motion of objects has temporal continuity, i.e., the above two assump-
tions hold for sequential time intervals.

The first two assumptions define how moving objects (those that are not created
nor destroyed) act locally in time, while the third smoothness assumption says that
the first two will hold over temporally sequential inputs, and allow decisions to be made in time.

THE ALGORITHM

Figure 1 shows the basic geometry of the nonconvergent binocular vision system. The two eyes of the system are separated by a distance of $2e$ and are located at positions $(-e, 0, f)$ and $(e, 0, f)$, looking towards infinity in the minus $z$ direction. (In Fig. 1 the $y$ axis can be considered to project out of the plane of the paper towards the reader.) By similar triangles it is easy to show that a point $P = (x, y, z)$ on an object will have a projection on the plane $z = 0$ (the image plane) in the right and left eye's coordinate systems respectively as:

$$x_r = \frac{(x - e) \cdot f}{f - z} \quad (1a)$$

$$y_r = \frac{y \cdot f}{f - z} \quad (1b)$$

$$x_l = \frac{(x + e) \cdot f}{f - z} \quad (2a)$$

$$y_l = \frac{y \cdot f}{f - z} \quad (2b)$$

where $f$ is the focal length, and $(x_l, y_l)$ and $(x_r, y_r)$ are the projections of the point $P$ in the left and right eye views, respectively.

Provided that the point $P$ is visible by both the left and right eyes, then the three-dimensional coordinate of $P$ can be reconstructed from these projections as:

$$x = \frac{e \cdot (x_r + x_l)}{x_l - x_r} \quad (3a)$$

$$y = \frac{2 \cdot e \cdot y_r}{x_l - x_r} \quad (3b)$$

$$z = f - \frac{2 \cdot e \cdot f}{x_l - x_r} \quad (3c)$$

by solving for $x$, $y$, and $z$ in Eqs. (1) and (2). Note that $x_l - x_r$ must be "large enough" for the computation to be meaningful. For distant objects at the limit of the resolution of the imaging process, the stereo reconstruction is not computable.
Let \( L_i \) and \( R_r \) be the coordinates of point \( l \) in the left eye, \( r \) in the right. The three-dimensional point which can be constructed from \( L_i \) and \( R_r \), using Eqs. (3) will be denoted as \([l, r]\).

All possible combinations of \( L_i \) and \( R_r \) would result in roughly \( n^2 \) possible points from \( n \) real points in three-space. The number of matchings can be reduced by noting certain geometric properties of the nonconvergent binocular vision system.

From Eqs. (1) and (2) for a combination \([l, r]\) to be valid it must be true that the \( y \) coordinate of \( l \) in the left eye (denoted as \( y(L_i) \)) must be equal to the \( y \) coordinate of \( r \) in the right eye (denoted as \( y(R_r) \)). Thus from the geometry \([l, r]\) will be valid only if \( y(L_i) = y(R_r) \). For real image data, this constraint should be relaxed thus allowing for small vertical disparities.

In addition, from Eqs. (1a) and (2a) and as \( e \) is positive, and as \( z \) must be negative, then for \([l, r]\) to be a valid combination \( x(L_i) > x(R_r) \). As the image plane is not infinite in size as depicted in Fig. 1, there will be a minimum distance a point can lie from the image plane before it becomes visible to both eyes. In the limiting case \( z([l, r]) = 0 \) is the closest a point may lie to the eyes of the system. In this case \( x(L_i) - x(R_r) < 2e \) from Eqs. (3).

Let \( \tau_y \) the maximum amount of deviation permitted between \( y(R_r) \) and \( y(L_i) \) and let \( \tau_x \) be the restriction on \( x(R_r) \) and \( x(L_i) \) resulting from the finite image planes. Then \([l, r]\) can be restricted using the following two constraints:

\[
\begin{align*}
|y(L_i) - y(R_r)| &< \tau_y \quad (4a) \\
0 < x(L_i) - x(R_r) &< \tau_x \leq 2e. \quad (4b)
\end{align*}
\]

Figure 2 shows how the parameters \( \tau_x \) and \( \tau_y \) restrict the region for possible spatial matchings. Let the circle represent some feature in the left eye’s image. If we superimpose the left and right images then the rectangular region formed by \( \tau_x \) and \( \tau_y \) denotes the region in the right eye’s view from within which we may look for possible matches.

In relationship with the human visual system, the nonconvergent binocular system is a limiting case in which the eyes are fixed on a point an infinite distance from the viewer. The constraints \( \tau_x \) and \( \tau_y \) denote the region of fusion for the binocular system and can be compared to Panum’s fusional area in the human visual system. Results from psychology indicate that although matches must be limited to this area, it is possible to have more than one match per feature from within this area [7].

The problem can now be stated more formally; given \( P_i(t) \) to \( P_n(t) \), the three-dimensional location of the features at time \( t \), \( V_i(t) \) to \( V_n(t) \), the vector velocities of the same \( n \) points at time \( t \), and \( L_1 \) to \( L_n \), and \( R_1 \) to \( R_n \) (where \( L_i \) does not
necessarily correspond to $R_i$), the left and right eye views of the points at the next time interval (denoted by $t + 1$), the problem is to find mappings $\phi_l : N \rightarrow N$ and $\phi_r : N \rightarrow N$, where $N = \{1, \ldots, n\}$, such that $P_i$ has location $[\phi_l(i), \phi_r(i)]$ at time $t + 1$. Note that $P_i$ may have 0, 1, or more possible corresponding points at time $t + 1$. In addition, the algorithm must construct a list of points that have been “created” or that appear at time $t + 1$. Formally, it must find the points that are elements of $C = \{|[l, r]| \in L \wedge r \in R \wedge \exists P_i : [l, r] = [\phi_l(i), \phi_r(i)]\}$, where $L = \{L_1, \ldots, L_n\}$, $R = \{R_1, \ldots, R_n\}$, and $|[l, r]| = [\phi_l(i), \phi_r(i)]$ if the two points correspond to the same three-dimensional location. For the initial set of input points there are no previous inputs from which to construct the mappings $\phi_l$ and $\phi_r$. Thus for the first set of input points the algorithm only computes points that are elements of the set $C$.

From work on apparent motion [9] it is clear that spatial range limits the power of attraction; “when the distance between the circles is increased the quality of motion is sometimes affected, so that maintaining the perception of a smoothly moving figure may require some change of intervals or of the flash duration.” Korte’s third law of apparent motion [10] states that the increase in optimum difference in time is essentially proportional to any increase in separation in space, or equivalently, that the critical time increases linearly with distance. There is also the “visual momentum” effect reported by Ramachandran and Anstis [15]. Thus given a scene sampling rate the velocity of the object is also a limiting factor in object tracking. These two constraints can be incorporated within a general smoothness assumption by defining predicates velocity and distance which limit membership in $\phi_l$ and $\phi_r$ over the time interval $t \rightarrow t + 1$ as;

$$\text{distance (}i, j, k\text{) = }|P_i(t) - [j, k]| < \tau_d \quad (5a)$$

$$\text{velocity (}i, j, k\text{) = }\left|\frac{V_i(t) - ([j, k] - P_i(t))}{\text{ISI}}\right| < \tau_v \quad (5b)$$

where ISI is the inter-stimulus time interval, the notation $|a|$ denotes the length of the vector $a$, and $\tau_d$ and $\tau_v$ are constraints limiting the distance an object can move, and the change in velocity an object can exhibit during a given frame, respectively. Characteristics of monocular primitives that are used to restrict combinations of $[l, r]$ could also be used to restrict $\phi_l$ and $\phi_r$.

Consider the restriction of the tracking problem to two dimensions. Figure 3 shows how the values of $\tau_d$ and $\tau_v$ restrict the possible motions of an object being tracked. The small circle moving from left to right represents the motion of the feature being tracked. Suppose we are tracking it from its second position and that it has a constant velocity. The circle with radius $\tau_d$ shows the region which constrains the object’s next position using Eq. (5a). The object has been restricted from moving more than a distance $\tau_d$ between frames. The circle with radius $\tau_v$ restricts the region in which the object may move based on the maximum change in velocity an object may undergo between frames. This is the geometrical interpretation of Eq. (5b). The circle is centered at the predicted location of the feature in the next frame and has radius $\tau_v$. As both the distance and velocity predicates must hold we need look for valid matches only in the intersection of these two regions. If the velocity of the object were zero, then the circle with radius $\tau_v$ would be centered on the object, and
the two circles would be concentric. In this case their intersection would be the smaller of the two circles.

In three dimensions the circles become spheres and valid matches no longer are restricted to the plane. The intersection of the two spheres (which correspond to the two circles mentioned above) defines the region for possible matches (see Fig. 4).

The values of \( \tau_d \) and \( \tau_v \) must be set so that the motion of the object to be tracked does not violate distance and velocity. In the limiting case the two spheres must at least overlap so that it is possible to satisfy the conditions for the valid match. If there is some maximum velocity \( v \) and maximum acceleration \( a \) that an object may exhibit, then from (5), \( \tau_d \) and \( \tau_v \) must be set so that

\[
\tau_d \geq |v| \cdot \text{ISI} \tag{6a}
\]

\[
\tau_v \geq \frac{|a| \cdot \text{ISI}}{2} \tag{6b}
\]

\[
(|v| - \tau_v) \cdot \text{ISI} \geq \tau_d; \tag{6c}
\]

(6a) and (6b) relate the maximum velocity to \( \tau_d \), and the maximum acceleration to \( \tau_v \), while (6c) forces the intersection of the two spheres to be non-empty.
Unless the situation is ideal, or the values of \(\tau_v, \tau_y, \tau_d,\) and \(\tau_e\) are chosen so as to unreasonably constrain the problem there will still be a large number of possible combinations \([l, r]\) and possible mappings \(\phi_l\) and \(\phi_r\). In static stereopsis, even with sophisticated monocular primitives such as zero-crossings [11] or zero-crossings and peaks [12], not all local matches will be reduced. A common approach has been to use a global spatial relaxation process (in a possibly limited manner) to assign final disparity values [11]. Although spatial relaxation over disparities has been used with some success to assign final disparities, there is really no evidence to support such low-level "pulling" or global labelling [11].

One possible approach would be to apply some sort of relaxation on the vector velocity values of the features. Such an approach would have a psychological basis in the rule of common fate. There would be some difficulties with non-rigid objects and with scenes containing more than one object moving with different velocities, and such an approach would require a dense field of monocular primitives. This would, however, be an interesting starting point for future research.

Rather than use any sort of global mechanism to determine the final spatial and temporal correspondences, decisions are made locally in the spatial dimension but are made later in the temporal dimension resulting in a temporal smoothing of the tracked features. A very simple model for the motion of three-dimensional features was constructed. By considering possible temporal combinations of motion hypotheses inconsistencies can be identified. These inconsistencies can be used to locally trim hypotheses to determine the correct spatial and temporal matches.

For the purposes of this discussion let circles represent features. One of the following labels (see Fig. 5) can be identified with the between-frame motion for each point (see Table 1). In addition two or more points in one frame may have the same candidate point for tracking in the next. Thus branches of SPLIT and TRACK labellings may also be identified as MERGE hypothesis.

<table>
<thead>
<tr>
<th>Label</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREATE</td>
<td>The point is created from the void</td>
</tr>
<tr>
<td>DEATH</td>
<td>The point disappears</td>
</tr>
<tr>
<td>TRACK</td>
<td>The point moves from one frame to the next</td>
</tr>
<tr>
<td>SPLIT</td>
<td>The point undergoes fission to become two or more points</td>
</tr>
</tbody>
</table>
In a world with perfect monocular feature detectors the set of tracking hypotheses would be much simpler. As no perfect monocular feature detector exists it is quite possible that SPLIT and MERGE hypotheses are not just the result of poor matching, but that the monocular features do SPLIT and MERGE. Take a sheet of reflective material and bend it back and forth, the specular reflection will move along the surface of the material, splitting and fusing. A monocular primitive that is sensitive to peaks in the intensity of the image would find that the monocular primitives do SPLIT and MERGE. It would be incorrect to remove SPLIT and MERGE hypotheses in a real world application simply because most real world objects do not undergo these effects.

Possible labellings are assigned to a point \( P(t) \) as follows: Let \( M_i = \{ [l, r] | l \in L \land r \in L \land \text{distance}(i, l, r) \land \text{velocity}(i, l, r) \} \). Then \( P(t) \) is identified with a TRACK labelling if \( |M_i| = 1 \), it is identified with a SPLIT labelling if \( |M_i| > 1 \), and it is identified with a DEATH labelling if \( |M_i| = 0 \). Points \([l, r]\) which are not associated with any \( M_i \) are considered for the CREATE label.

\( |M_i| \) counts the number of possible motion hypotheses for a given point \( P_i \). If \( |M_i| = 1 \) then there is only one motion hypothesis for the point \( P_i \), and it can be associated with a TRACK labelling. If \( |M_i| > 1 \) then the point \( P_i \) has competing motion hypotheses. At this point it is not possible to determine if the point actually splits into two or more points, or if some of the possible motion hypotheses are invalid. If \( |M_i| = 0 \) then the point \( P_i \) has no possible motion hypotheses. This point is said to have died. Finally, some combinations \([l, r]\) will not have been candidates for any point \( P_i \). These points are considered for creation hypotheses.

Let \( P_i(t) \) be a possibly empty set of points for which both motion and positional information is known. Examine the monocular primitives for the next frame and construct all possible matches \([l, r]\). From \( P_i(t), V_i(t), \) and \([l, r]\) all possible motion hypotheses according to the model given above can be constructed. Rather than deciding the correct matches at this point, assume (temporarily) that all of the matches are correct. Thus each possible match \( P_i(t) \rightarrow [l, r] \) is considered to give rise to a new point at time \( t + 1 \). The process of examining the monocular input is repeated for time \( t + 2 \) and all labellings are then considered. Thus there are double mappings of the form \( P_i(t) \rightarrow [l, r] \rightarrow [l', r'] \).

The algorithm assumes that every possible matching \( P_i(t) \rightarrow [l, r] \) is correct. For each possible match \( P_i(t) \rightarrow [l, r] \) a point (with corresponding positional and velocity parameters) is constructed. The constructed points are temporarily assumed to be valid at time \( t + 1 \). Possible matchings are then considered between the constructed point at time \( t + 1 \) and all points constructable from the monocular features presented to the algorithm at time \( t + 2 \). Inconsistencies between the mapping \( P_i(t) \rightarrow [l, r] \) and \([l', r']\) are used by the test phase of the algorithm to restrict the double mappings and to decide on the correct mapping \( P_i(t) \rightarrow [l, r] \).

There are two sub-problems within dynamic stereopsis. The first is that there may be ghosts arising from the problem of static stereopsis, and the second is that true object motion may be hidden within SPLIT tracking results. Consider the tracking problem alone for a moment; assume that the motion of the features obey a general smoothness assumption, then the points will move smoothly from one frame to the next. A problem will arise only when for a given frame there exists more than one candidate point for tracking. If the incorrect choices arise due to noise (or from some other random source such as the accidental alignment of true features in an
TABLE 2
Hypothesis Reduction

<table>
<thead>
<tr>
<th>Current</th>
<th>Next</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEATH</td>
<td>—</td>
</tr>
<tr>
<td>CREATE</td>
<td>GHOST</td>
</tr>
<tr>
<td>TRACK</td>
<td>TRACK</td>
</tr>
<tr>
<td>SPLIT</td>
<td>GHOST</td>
</tr>
</tbody>
</table>

image), then one would expect that although they satisfy the smoothness assumption for the current frame, they will not satisfy it for a sequence of frames. A large class of these types of errors could be eliminated by simply examining the “matching potential” of the points for the next frame. If the points are truly invalid then they should die out as there will be no matches for these points in future frames which do satisfy the smoothness assumption. When discontinuities are present in the motion of the features (such that the motion violates the general smoothness assumption), the points will die. Once the discontinuity ends the points will be recreated and tracked correctly.

In order to trim the possible matches it seems that there are two cases of interest; when a CREATE label is followed by a DEATH label, and when one branch of a SPLIT labelling is followed by a DEATH label. The hypothesis reduction is summarized below (see Table 2). This hypotheses trimming can be demonstrated graphically (see Figs. 6, 7, and 8). There are two entries in Table 1 which are inconsistent with the general smoothness assumption. These can be used to eliminate possible matchings (treat them as GHOST matchings). The first is when a CREATE labelling is followed by a DEATH labelling (see Fig. 6). Some accidental combination of monocular features has given rise to a valid three-dimensional point in one frame which then disappears in the next. This “creation” of an object is considered to be a GHOST labelling and it is removed from the list of valid hypotheses.

---

**Fig. 6.** CREATE followed by DEATH.

---

**Fig. 7.** SPLIT followed by DEATH.
The second inconsistency occurs when a branch of a SPLIT labelling is followed by a DEATH labelling (see Fig. 7). One branch of the three-way SPLIT is shown to have a DEATH labelling in the following frame while all other branches have TRACK labellings. The set of hypotheses is trimmed by considering the branch of the SPLIT that was followed by a DEATH labelling as a GHOST, and removing it from the list of hypotheses. In performing this operation the number of branches of the given SPLIT labelling has been reduced by one.

There is another possibility when trimming branches of a SPLIT labelling (see Fig. 8). In this case all but one of the branches of a SPLIT labelling are followed by DEATH labels. When these branches are removed there remains a SPLIT label with only one branch. Consider the value of $M_i$. It will have been reduced from the condition $|M_i| > 1$ to $|M_i| = 1$. The SPLIT labelling of this node is replaced by TRACK to reflect the new value of $|M_i|$. The given SPLIT labelling has been reduced by one.

This trimming follows from the general smoothness assumption. When a label is identified as a GHOST it is removed from the list of possible labels. Thus it is possible that in reducing a SPLIT label that there exists a different branch of the SPLIT which has been reduced due to trimming to a TRACK.

At this stage the final correspondences must be determined. All surviving labels have been found to satisfy the general smoothness assumption. For all, except SPLIT labellings, final tracking and disparity assignments have now been obtained. For SPLIT labellings a final restriction can be made to further reduce SPLIT labels to TRACK, or to reduce the number of branches in the SPLIT labelling.

As mentioned earlier, distance limits the effect of attraction for motion correspondence. This effect is most apparent when a SPLIT labelling is being considered. Kolers [9] reported two experiments which showed that in a SPLIT situation the presence of near points reduced and in some cases eliminated the attraction of far ones. This can be used as a final constraint to remove some competing SPLIT branches. Remove competing SPLIT branches that lie at a distance greater than some fixed multiple $k$ of the distance that the minimum distance SPLIT branch lies from the father point in the previous frame. This has the added affect of favoring no motion hypotheses for stationary points having a stationary candidate point for tracking.

The use of $k$ to remove branches of SPLIT labelling will not reduce all SPLIT labels to TRACK. Setting the value of $k$ to 1 will reduce most SPLIT labels, but any point which splits into two equidistant points will not be reduced to a TRACK
labelling. It is not clear that all SPLIT labelling should be reduced, as some inputs may exhibit objects which undergo fission, and SPLIT labels must be retained if these points are to be correctly tracked.

The proximity rule used to obtain final matches is dependent upon the motion of the candidate feature with the smallest possible displacement. Trimming is performed in terms of multiples of the shortest distance. As this takes into account inputs with different speeds, or different inter-stimulus intervals, the choice of $k$ will not be dependent upon either the velocity of the features, or the length of inter-stimulus interval. $k$ must be set so that $k \geq 1$. If $k = 1$ then the algorithm will weight heavily against SPLIT results, choosing instead to interpret the SPLIT as a TRACK to the nearest possible match. If $(k \gg 1)$ then the algorithm will not reduce any SPLIT result to TRACK using the proximity rule.

Figure 9 shows how the value of $k$ is used to remove SPLIT branches. Figure 9 shows a point in one frame with arrows showing valid SPLIT branches labelled $A$ through $D$ in the next frame. Suppose that the value of $k$ was 2. Then as all branches that lie at a distance greater than $k$ times the distance to the nearest branch are eliminated, the nodes labelled $C$ and $D$ would be trimmed.

When all of the final decisions have been made from time $t$ to time $t + 1$, the points that were tracked to, or created at time $t + 1$, become the points for which tracking is required. The values of $P_i$ and $V_i$ are updated, with created points given an initial velocity of zero.

The algorithm does not weight against MERGE hypotheses except that inconsistencies are used to reduce the total number of possible matchings, and thus indirectly the MERGE hypotheses are removed. There appear to be two choices; either treat a MERGE as simply being a mapping to more than one point having the same physical location, or actually merge the points to form only one single point.

Why is this a problem? Consider the case of two points at the same depth moving at constant velocity at right angles having a frame in which there exists a single point of intersection. Depending upon the representation of a MERGE, different effects will be perceived by the algorithm. If more than one point is allowed to occupy the same physical location at the same time then there is no problem as each point will be seen to continue in a straight line. If points are forced to merge then the velocity of the merged point must be computed. It is not obvious how this should be done, and depending on the choice different effects will be perceived.

Neither approach seems to be particularly attractive. Aside from the physical impossibility of having more than one point occupy the same physical location at the same time there are other problems in allowing MERGEs to be treated as simple TRACK hypothesis. Unless the motion is actually a TRACK and not a MERGE there will be two points for tracking that will be indistinguishable by the algorithm;
they will have identical velocity and motion parameters. This would not be desirable. The other choice is unattractive for reasons that have already been mentioned. What is really desired is an approach that has the advantages of both schemes, but with none of their problems. The problem is this; if while tracking points are merged together, (assuming some function to obtain the velocity of the new point), what process should be used in the following frame to SPLIT the point apart if the merge was really a collision? If more than one point is allowed to occupy the same physical location how are points truely merged together once a true merge has been detected?

This problem was solved by allowing more than one point to occupy the same physical location provided that the points have different velocity parameters. Whenever two (or more) points have identical positions in the six-space \((P_x, P_y, P_z, V_x, V_y, V_z)\) a true MERGE was assumed to have taken place (as the points can not be distinguished by the system). Otherwise the points are considered to be distinct. This has the advantage that it solves both problems without having to remember "special" points that may (or may not have) been merged together. Note that when two points meet at a given location, the algorithm may interpret the points as crossing, bouncing, or the points may SPLIT, with one branch of the SPLIT crossing, while the other bounces. The interpretation depends upon the velocities of the points, their positions, and the values of \(\tau_v, \tau_d, \) and \(k\).

**SOME ALGORITHMIC PROPERTIES**

There are two properties that we would like the algorithm to have. First, the true track of the objects should be included within the list of tracks produced by the algorithm. Thus for a given point we should not overconstrain its possible matches. Second, it should remove as many invalid spatio-temporal matches as possible from the list of possible matches. Consider the first of these two properties.

Let \(P_t\) be a point that satisfies the general smoothness assumption from time \(t\) to time \(t + 2\). As \(P_t\) satisfies the general smoothness assumption from time \(t\) to time \(t + 2\) then \(P_t(t) \to P_t(t + 1) \to P_t(t + 2)\) will be a valid double mapping. Thus the second hypothesis will not be a DEATH hypothesis during the test phase of the tracking algorithm and the mapping \(P_t(t) \to P_t(t + 1)\) will survive up to the point of deciding the final match for the point \(P_t\). Then up to the point at which final matching assignments are made, \(P_t(t) \to P_t(t + 1)\) will be considered as a possible match.

Note that this does not say that points cannot be created or destroyed. It only says that provided the points are not created nor destroyed (i.e., they satisfy the general smoothness assumption), \(P_t(t) \to P_t(t + 1)\) will still be considered as a possible match.

Unless \(P_t\) is part of a SPLIT hypothesis \(P_t(t) \to P_t(t + 1)\) will be the only hypothesis for point \(P_t(t)\) and thus the correct match will be chosen. If \(P_t\) is part of a SPLIT hypothesis, then there exist \(Q_j(t + 1)\) and \(Q_k(t + 2)\) such that \(P_t(t) \to Q_j(t + 1) \to Q_k(t + 2)\) is valid double mapping. \(P_t(t) \to P_t(t + 1)\) will not be part of the final match only if it is trimmed during the decision phase of the algorithm. For this to be the case then for one of the competing matches, \(Q\) say, \(|P_t(t) - P_t(t + 1)| > k \cdot |P_t(t) - Q_i(t + 1)|\).
Thus except for this the algorithm will not exclude the correct match. In order for
the algorithm to overlook the correct match there must not only exist an incorrect
match that satisfies the general smoothness assumption, but the incorrect match
must also be closer to the parent point in space at time \( t \).

Now consider the ability of the algorithm to remove invalid matches. First, note
that if the inter-image time is ISI, then if \( V \) is the maximum velocity that we expect
a point to have then for a correct match \( \tau_c > |V| \) and \( \tau_u > |V| \) ISI. This will allow a
point starting from rest to accelerate to velocity \( V \) in one frame. A feature exceeding
these limits will be considered to be violating the general smoothness assumption.
The constant \( k \) used in the decision module must be defined to be at least one. The
larger the value of \( k \) the more SPLIT hypotheses that will be permitted. When
\( k = 1 \) there will be very few (if any) SPLIT hypotheses generated by the system.

A choice of \( k \leq \sqrt{2} \) has the effect of blocking crossed motion in a large number of
cases. Although increased path length is not the only reason for the absence of
crossed motion in apparent motion it may be a contributing factor.

What does the algorithm do when there is no motion? Each point will have a track
hypothesis with a distance of zero. Thus the final hypothesis will trim any and all
SPLIT hypothesis to a stationary TRACK. No ghosts will be eliminated by the
algorithm as a ghost that appears in one frame will appear in the next. Thus ghosts
will be tracked to other ghosts. In the limiting case of no motion the algorithm
degrades to whatever constraints were used to find the valid matches \([l, r]\).

THE SAMPLING PROBLEM

In limiting the types of motion combinations that are permitted, the implicit
assumption was made that the algorithm would base its decisions on information
available from two ISIs. A sampling problem now arises; would there be an
improvement in the algorithms performance if more than two ISIs were used by the
algorithm? In other words, how much temporal smoothing is required?

What advantages would a version of the algorithm using more than two ISIs have
over a version using only two. (Let \( \alpha_n \) be the application of a version of the
algorithm using \( n \) ISIs to make a decision.) The only case that could use the
additional information would be the case in which a SPLIT result for the first ISI
had more than one branch in the second ISI with a non-DEATH label (see Fig. 7).
Using \( \alpha_n \), where \( n > 2 \), it could be possible to detect that some of the non-DEATH
labels also die out in the \( n - 1 \) remaining ISIs. Thus \( \alpha_n \) with \( n > 2 \) can produce a
better result than \( \alpha_2 \). However, \( \alpha_n \) produces only a more refined set of labels.
TRACK, DEATH, CREATION, and GHOST results obtained using \( \alpha_2 \) will also be
obtained using \( \alpha_n \). A SPLIT result from \( \alpha_2 \) will appear as either a SPLIT or TRACK
result with \( \alpha_n \), but the features associated with these labels in \( \alpha_n \) will be associated
with the corresponding SPLIT label in \( \alpha_2 \). Using \( \alpha_n \) to represent more refined, the
above result can be expressed as \( V_n \geq 2 : \alpha_n \subseteq \alpha_2 \).

Although the algorithm does not currently do so, it would be possible to obtain \( \alpha_n \)
from \( \alpha_2 \) with a small amount of post processing. Apply \( \alpha_2 \) and obtain the final set of
labels determined by the algorithm. For all except SPLIT labels we are done. For
SPLIT labels, examine the next \( n - 1 \) ISIs. Reduce each branch of the SPLIT label
which does not have at least one corresponding feature in the last frame examined.
Fig. 10. Cutting's walking figure.

This will remove any competing branch of a SPLIT label that does not survive over the next \( n - 1 \) ISIs. This set of labels is identical to \( \alpha_n \).

**EXPERIMENTAL RESULTS**

Figure 10 shows the superposition of a number of frames of a three-dimensional moving light display of Cutting's walking figure [1] in which the points in a given frame have been joined up so as to make the figure more visible. The true motion of the figure was from the left to the right across the screen. The results of tracking these moving lights was displayed as three orthographic projections in the \( x-y \), \( x-z \), and \( z-y \) planes, and also as a perspective projection in the \( x-y \) plane. The perspective projection is a projection back into the left eye's view of the vision system. As the tracking results show (Fig. 11), the figure was observed "walking" from the left to the right along a straight line parallel to the plane of the vision system.

Figure 12 shows the tracking of the same figure, only in this case the figure was set up to walk at an angle towards the camera. Although the display is not as clear, the tracking does show the figure walking at an angle towards the camera.

The algorithm has also been applied to real world data. A digitized stereoscopic film was created of a small toy boat. The Moravec [13] operator was applied to each frame, and the points detected by the operator were used as input to the tracking algorithm. Figure 13 shows a frame from this stereoscopic film including the location of the identified feature points.

We chose as our monocular features, points identified by the Moravec operator as being interesting for two reasons; first, the operator produces a large reduction in the number of monocular features, thus reducing the computation required to analyze a sequence of images; and second, as the Moravec operator is notoriously unstable (small changes in an image may result in a large change in the location of a feature point), it was a good test of the effectiveness of the algorithm on poor and noisy data.
Fig. 11. Tracking of Cutting's walking figure.

Fig. 12. Tracking of Cutting's walking figure showing motion in depth.
Figure 13. Moravec responses.

Figure 14 shows the tracking of the feature points identified by the Moravec operator. The tracked points were displayed as a perspective projection back into the left eye of the vision system, and also as a perspective axis with the recovered depth of the feature points. The motion of the boat was from the left to the right across the screen, with the boat approaching the camera at an angle. Although the results were quite noisy, the true motion of the boat was observed.

Fig. 14. Tracking of Moravec operator response.
CONCLUSIONS

We have presented an algorithm for the dynamic stereo problem. The algorithm was based on a general smoothness assumption; a feature point will remain relatively unchanged from one frame to the next. The algorithm is conservative, in that provided that the features satisfy the general smoothness assumption and a proximal constraint then the correct matches will be included in the tracking observed by the algorithm. Competing matches are eliminated if they (i) violate the general smoothness assumption over the next two frames, or (ii) violate the proximal rule.

The algorithm utilizes spatio-temporal context constraints, and takes a wait and see approach to both the temporal matching and stereopsis problems. The algorithm performs the short-range matching function, in that if a given object leaves the system view and then returns the two objects will not be identified as being the same. It is assumed that there exists some long range matching function to interpret the results of the temporal stereopsis algorithm.

REFERENCES