MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ARTIFICIAL INTELLIGENCE LABORATORY

A.I. Memo No. 1382 October 1992

Visual Tracking

M. Ali Taalebinezhaad

Abstract

A typical robot vision scenario might involve a vehicle moving with an unknown 3D motion (translation and rotation) while taking intensity images of an arbitrary environment.

This paper describes the theory and implementation issues of tracking any desired point in the environment. This method is performed completely in software without any need to mechanically move the camera relative to the vehicle.

This tracking technique is simple and inexpensive. Furthermore, it does not use either optical flow or feature correspondence. Instead, the spatio-temporal gradients of the input intensity images are used directly.

The experimental results presented support the idea of *tracking in software*. The final result is a sequence of tracked images where the desired point is kept stationary in the images independent of the nature of the relative motion. Finally, the quality of these tracked images are examined using spatio-temporal gradient maps.

Keywords: Tracking, motion vision, direct method, spatio-temporal gradients.

© Massachusetts Institute of Technology, 1992

Acknowledgments: This paper describes the research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for research at this laboratory is provided in part by DARPA under the ONR contract N00014-91-J-4038.

1 Introduction

In many applications there is a definite need for tracking an environment point using vision sensory data. This task is equivalent to obtaining a sequence of tracked images which is, in essence, the well known *tracking* problem. People have been working on different aspects of this problem using various techniques for many years [13, 7, 18]. For example, Aloimonos & Tsakiris [2] propose a method for tracking a foveated target of known shape; Bandopadhay et al. [3] use optical flow and feature correspondence for tracking the principal point in order to find the motion in a special case (they assume that there is no rotation along the optical axis) without considering noise; and Sandini & Tistarelli [17] use an optical flow based tracking method for finding the depth in a special case (no rotation along the optical axis). All these methods use optical flow and/or feature correspondence and address only special cases.

Traditionally, tracking has been associated with mechanically moving the camera to keep the image of a particular point stationary at the image center. Some techniques even rely on such a system. For example, Thompson [23] introduces an optical flow method for recovering the motion in the special case where the rotational velocity along the optical axis is zero. His method requires a sequence of tracked images at the principal point but he acknowledges that the actual implementation of such tracking requirement in engineering systems is not possible yet.

Hardware tracking is done by physically moving the camera with respect to the environment. Considering that in general the point of interest has a motion relative to the observer, the second tracked image cannot be obtained in one step. As a result, a feedback control loop is required for the camera system to compensate for the errors resulting from the new position of the tracking point [14, 6, 8, 12, 24, 5]. These difficulties and other problems such as expense, need for real time response, and potential errors involved make mechanical (hardware) tracking unattractive, especially in vision systems.

This paper describes how a sequence of tracked images can be constructed from an arbitrary image sequence (resulting from an arbitrary 3D relative motion) using a purely software technique.

2 Equivalent Rotational Velocity

Figure 1 shows a viewer-centered coordinate system. The coordinate system $OXYZ$ is attached to the vision system. The viewer moves with arbitrary rotational and translational velocities relative to an arbitrary rigid environment and takes a sequence of images. We refer to any consecutive pair of images in the input sequence as *original images*. Our final goal is to obtain a sequence of tracked images where the image of a desired point (*fixation point* or *tracking point*) is kept stationary no matter what kind of relative motion is involved. In any pair of input images, we can use the *first original* image as the *first tracked* image so we only need to construct the *second fixated* image.

If point 1 is chosen for *tracking* in the first original image, then in general its corresponding image point in the *second original* image moves to a new location such as point 2; see fig. 1. Determining the location of point 2 is the same as finding the motion of the tracking point

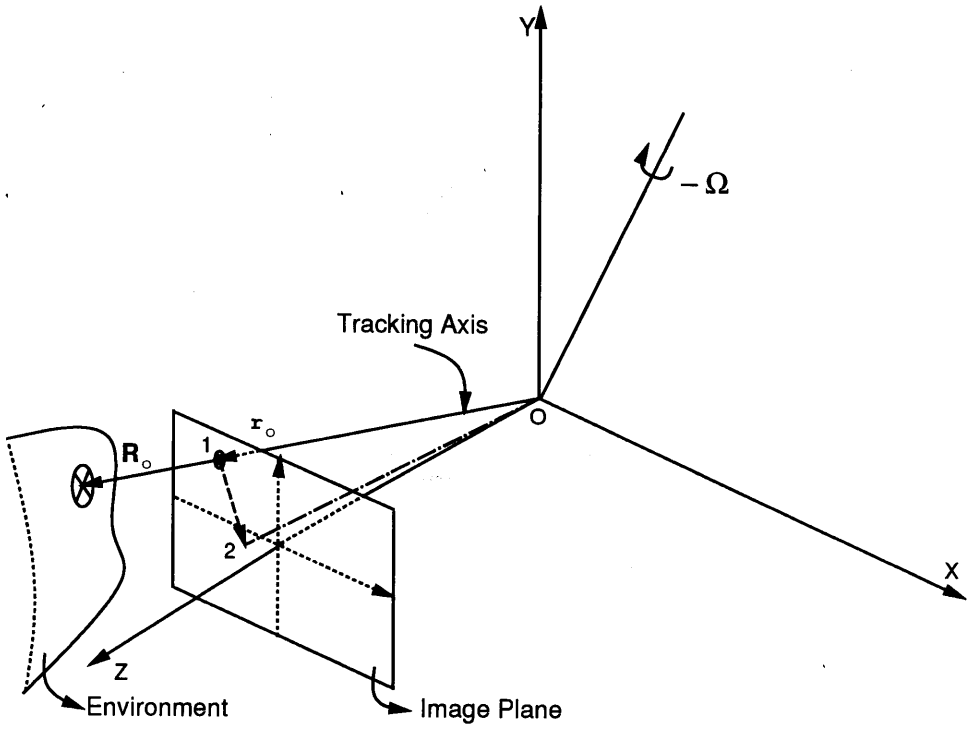

Figure 1: An imaginary rotation opposite to the *equivalent rotational velocity*, $-\mathbf{\Omega}$, is applied to the vision system to bring point 2 to point 1. This rotation transforms the *second original* image into the *second tracked* image.

in the image plane (the so called *fixation velocity*). Earlier, we introduced a simple technique for the estimation of the fixation velocity [19, 22]. The experimental results have shown that the fixation velocity can be estimated reliably even from real and noisy images [21, 22]. Accordingly, it is assumed here that the fixation velocity components (u_o, v_o) in the image plane have been already computed. In other words, we know the new location of the tracking point (point 2) in the image plane.

There are infinite combinations of translations and rotations which can be applied to the vision system or camera to bring the image point at 2 to the location 1 and result in a sequence of tracked (fixated) images. Among all these combinations, we choose to accomplish this task by a pure rotation because it does not require any depth information. To find the desired rotation, we first introduce an *equivalent rotational velocity*, $\mathbf{\Omega} = (\Omega_x, \Omega_y, \Omega_z)$, as a rotation which can result in the same fixation velocity (u_o, v_o) at the fixation point (x_o, y_o). It can be shown that the components of $\mathbf{\Omega}$ must satisfy the following set of equations [20]

$$\begin{cases} u_o = x_o y_o \Omega_x - (x_o^2 + 1)\Omega_y + y_o \Omega_z \\ v_o = (y_o^2 + 1)\Omega_x - x_o y_o \Omega_y - x_o \Omega_z. \end{cases} \quad (1)$$

There are also an infinite number of rotations $\mathbf{\Omega}$ that satisfy the system of equations in 1. However, we choose the only one which does not introduce any new rotational velocity along the fixation axis $\mathbf{r}_o$. Mathematically it is equivalent to having $\mathbf{\Omega} \cdot r_o = 0$ which results in an

extra constraint on the components of $\mathbf{\Omega}$,

$$x_o\Omega_x + y_o\Omega_y + \Omega_z = 0. \tag{2}$$

Considering that the fixation velocity (u_o, v_o) has already been computed and the fixation point coordinates x_o and y_o are known here, the equivalent rotational velocity $\mathbf{\Omega}$ is obtained by solving the combination of the three linear equations in 1 and 2. For example, in the special case that the tracking point is at the principal point, $x_o = y_o = 0$, the equivalent rotational velocity becomes simply,

$$\mathbf{\Omega} = (v_o, -u_o, 0). \tag{3}$$

However, it should be emphasized that the tracking point is *not* restricted to the principal point and virtually any point can be chosen for tracking.

3 Pixel Shifting Process

After obtaining the equivalent rotational velocity $\mathbf{\Omega}$, the task of obtaining the second tracked image is equivalent to finding the transformation exerted on the second original image if an imaginary rotation $-\mathbf{\Omega}$ is applied to the vision system.

Similar to eqn. 1, the following set of equations give the component of the corresponding shifting vector $(u,\ v)$ for any pixel $(x,\ y)$ of the second original image

$$\begin{cases} u &= & -xy\Omega_x &+ & (x^2+1)\Omega_y &- & y\Omega_z \\ v &= & -(y^2+1)\Omega_x &+ & xy\Omega_y &+ & x\Omega_z. \end{cases} \tag{4}$$

Here Ω_x, Ω_y and Ω_z are known values. As a result, the *shifting vector* (u, v) can be obtained for every pixel of the second original image. Note that this shifting vector is not uniform over the image but varies depending on the location of a pixel.

Figure 2 shows the process of constructing the *second tracked* image using the *second original* image. The process is called *pixel shifting*. The task of constructing the second tracked (fixated) image is equivalent to finding the brightness at any of its pixels. The brightness at pixel (x, y) of the second tracked image is the same as the brightness at the corresponding point $(x-Tu, y-Tv)$ in the second original image, where T is the time interval between two original images. In general, a computed original point will not be located at the center of a pixel in the second original image. As a result, its brightness cannot be read directly from the image file and should be computed by a method like *averaging*, *bilinear interpolation* or *bicubic interpolation* of the brightnesses at its neighboring pixels.

3.1 Bilinear interpolation

We showed that the brightness E at pixel (x, y) of the second tracked image is the same as the brightness at the pixel $(x - Tu, y - Tv)$ of the second original image where the shifting vector (u, v) is given by eqn. 4 and T is the time interval between two original images.

In practice, the point $(x-Tu, y-Tv)$ does not necessarily coincide with any pixel. Instead it is usually lies between four pixels whose brightnesses may be denoted by $E_{i,j}$, $E_{i,j+1}$, $E_{i+1,j}$,

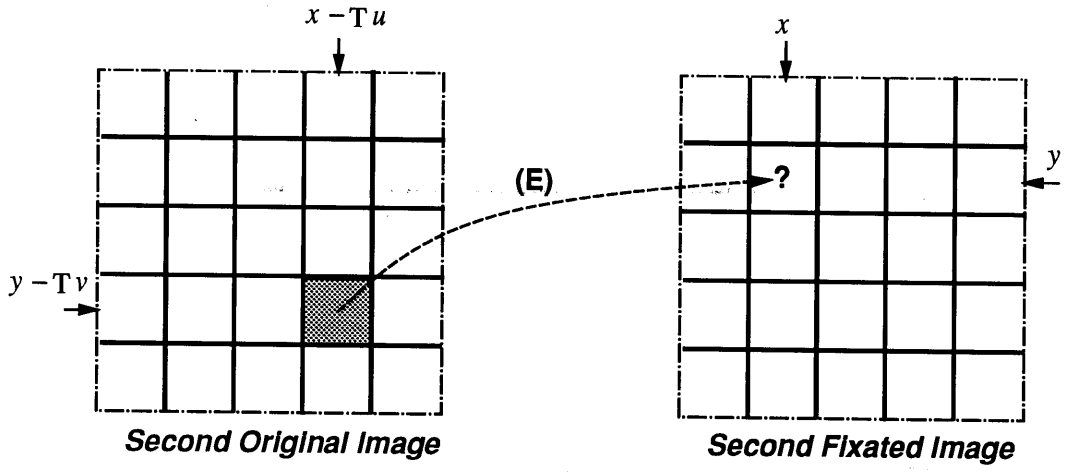

Figure 2: The *pixel shifting process* for constructing the *second tracked (fixated)* image from the *second original (initial)* image.

and $E_{i+1,j+1}$; see fig. 3. In this figure, p and q are the horizontal and vertical distances of the mapped point from pixel (i, j). Considering that this can happen for any pixel, the *average* $\frac{1}{4}(E_{i,j} + E_{i,j+1} + E_{i+1,j} + E_{i+1,j+1})$ is not a good estimation for E because it corrupts the constructed image by introducing aliasing.

Bilinear interpolation of the surrounding brightness levels has proven to be a good estimate for E. It is computed as,

$$E = (1-p)(1-q)E_{i,j} + p(1-q)E_{i,j+1} + q(1-p)E_{i+1,j} + pqE_{i+1,j+1}. \quad (5)$$

As shown in fig. 3, p and q represent the horizontal and vertical distance of the mapped point from pixel (i, j). Such an algorithm gives the largest weight to the pixel closest to the mapped point and results in the exact brightness value when it coincides with any pixel, $p = q = 0$.

All the images which we have constructed are obtained using *bilinear interpolation.* Our experimental results have shown that such interpolation is quite satisfactory. There are some other techniques such as *bicubic interpolation* [1, 4, 11, 15, 16] which are much more expensive, however, we did not find that we needed to use them in this work.

4 Experimental Results

Two successive original frames of the *landscape* image sequence (taken at the *Imaging Laboratory of Carnegie Mellon University*) are shown in figures 4 and 5. These are $8-bit$ images but the last two digits are usually too noisy to be reliable. The true motion between these frames is a combination of translation and rotation. The real rotation is 0.3 deg about the optical axis Z and the real translation is 2 mm along the horizontal axis X.

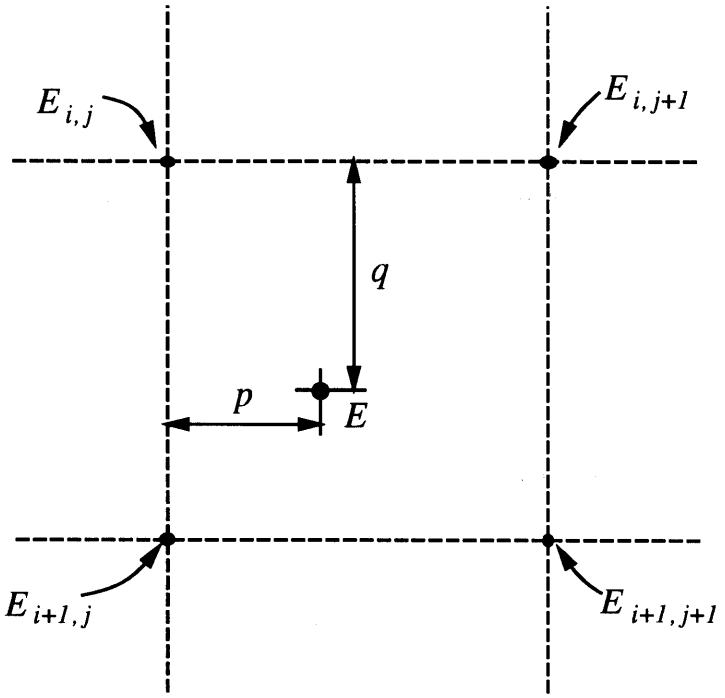

Figure 3: The mapped point in the second original image does not necessarily coincide with any single pixel. Instead it is usually lies between four pixels.

4.1 Gradient maps of the original images

Using the formulation given in the appendix, we can compute the brightness gradients. The spatio-temporal gradients are the primary source of input data for direct method algorithms which do not use either optical flow or feature correspondence. The corresponding spatial and temporal brightness gradients for the original landscape image sequence are shown in figures 6, 7 and 8 respectively.

In these maps, larger gradient values are shown brighter. Such gradient maps suggest a way of visually representing the brightness gradients which renders them more intuitively meaningful. The horizontal gradient map E_x in fig. 6 captures the vertical lines and feature in the images. Similarly, the vertical gradient map E_y in fig. 7 picks up the edge-like lines and features in the image. These experimental results show that the spatial gradients capture the geometric and shading characteristics of the images. It is important to notice that the computation behind spatial gradients is very simple. However, they implicitly capture the edges, features, and boundaries in the scene.

The temporal brightness gradient in fig. 8 tells us about the motion between two *original* images. First of all, the vertical lines and features are seen all over this temporal gradient map. This observation indicates that the motion has a horizontal translation component. Secondly, there are also horizontal lines in this gradient map but they become weaker as they get close to the left side of the map (this argument becomes more obvious if one compares the horizontal lines in here with those of E_y in fig. 7). This means that motion has a rotational

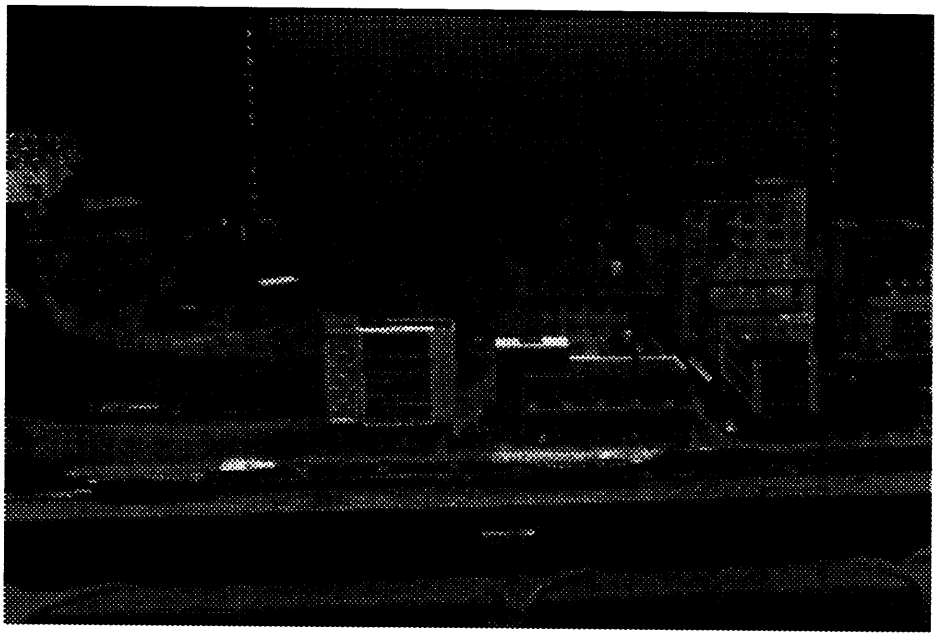

Figure 4: The *first original* frame in the landscape image sequence. The true motion is a 0.3 deg rotation about the nominal optical axis Z, and a 2 mm translation along the horizontal axis X.

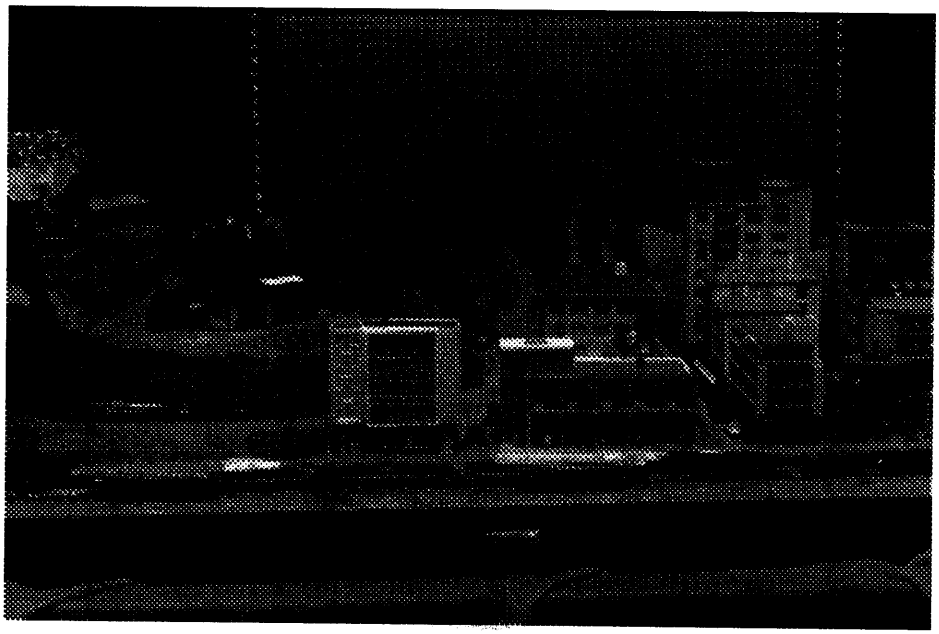

Figure 5: The *second original* frame in the landscape image sequence.

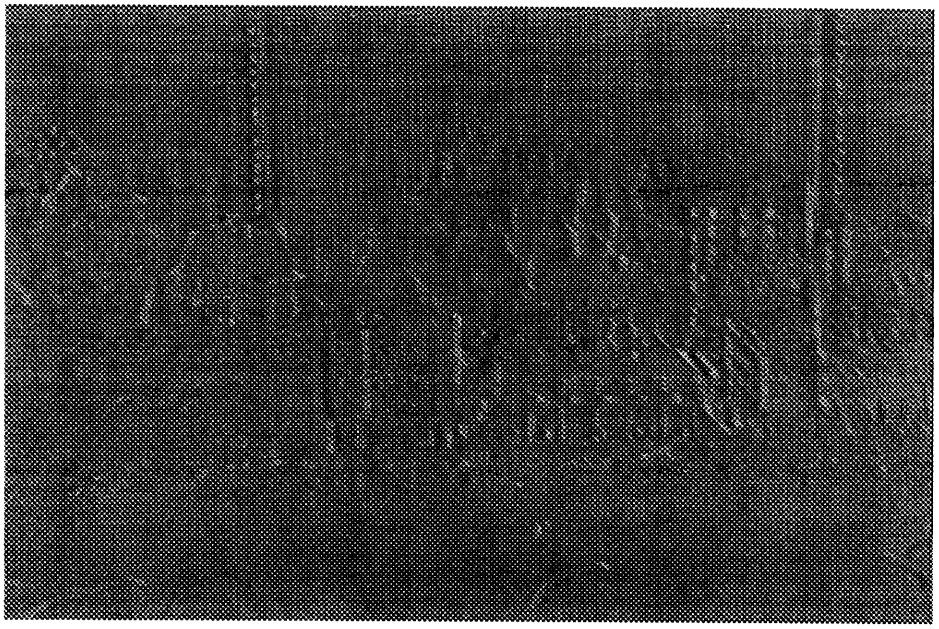

Figure 6: The visual representation of the spatial brightness gradient E_x for the *original* landscape image sequence. This horizontal gradient map captures the vertical edges and features in the image.

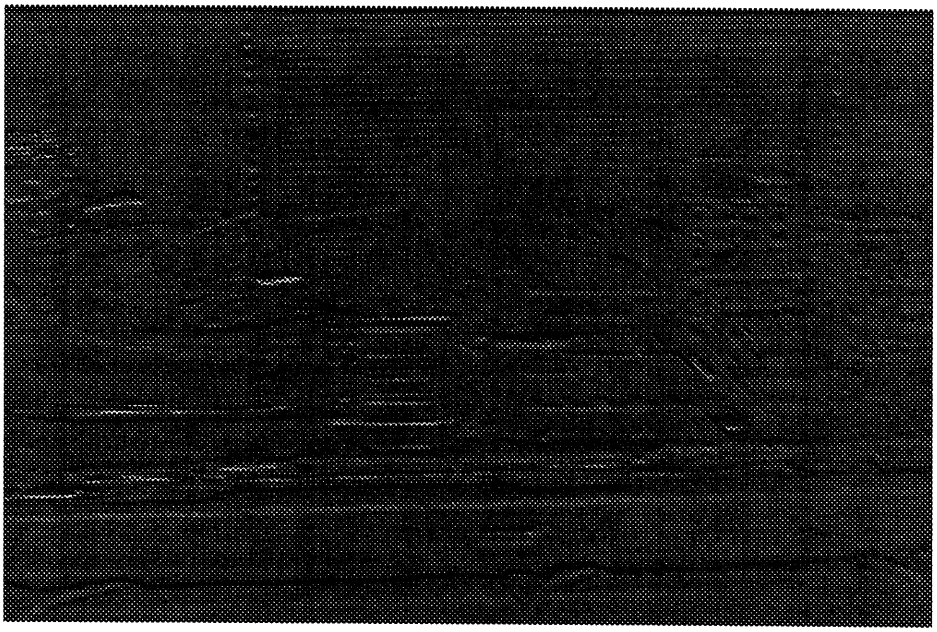

Figure 7: The visual representation of the spatial brightness gradient E_y for the *original* landscape image sequence. This vertical gradient map captures the horizontal edges and features in the image.

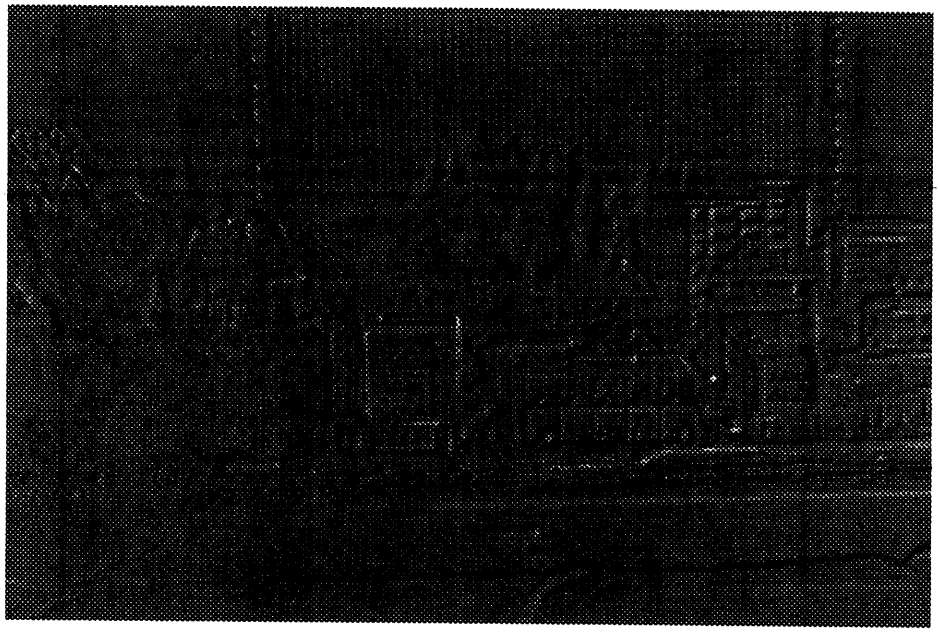

Figure 8: The visual representation of the temporal brightness gradient E_t for the *original* landscape image sequence. The vertical edges with relatively uniform strength suggest that motion has a horizontal translation component. The horizontal edges with decreasing strength towards left indicate that there is also a rotation centered at the left of the image center.

component which is centered in the left side of the image that is really the case for this image sequence [21, 22]. Also, we can observe that at any vertical stripe of the spatial gradient map, the horizontal lines become stronger as their distance from the center of the stripe increases. This observation indicates that the rotation center is located in the middle of the image.

4.2 Construction of the tracked images

The *landscape* images in figures 4 and 5 are used as input (original) images in our experiments. As we discussed earlier, the first original image (fig. 4) is directly used as the first tracked image. Then the *pixel shifting process* and the *bilinear interpolation* techniques (in section 3) are applied to the second original image in figure 5 to construct the second tracked image in fig. 9. This constructed image is quite good and looks as natural and crisp as the original images do. We will describe the quality of this constructed image further in the following section.

Depending on the size and direction of the equivalent rotational velocity $\mathbf{\Omega}$, the brightness E at some border pixels are not computable because they are mapped to points outside the original images domain. The brightness at such bordering pixels are given an arbitrary value of 0 which causes the appearance of bold black lines at the border of constructed images. This should not concern us because in general the results near the image borders are not

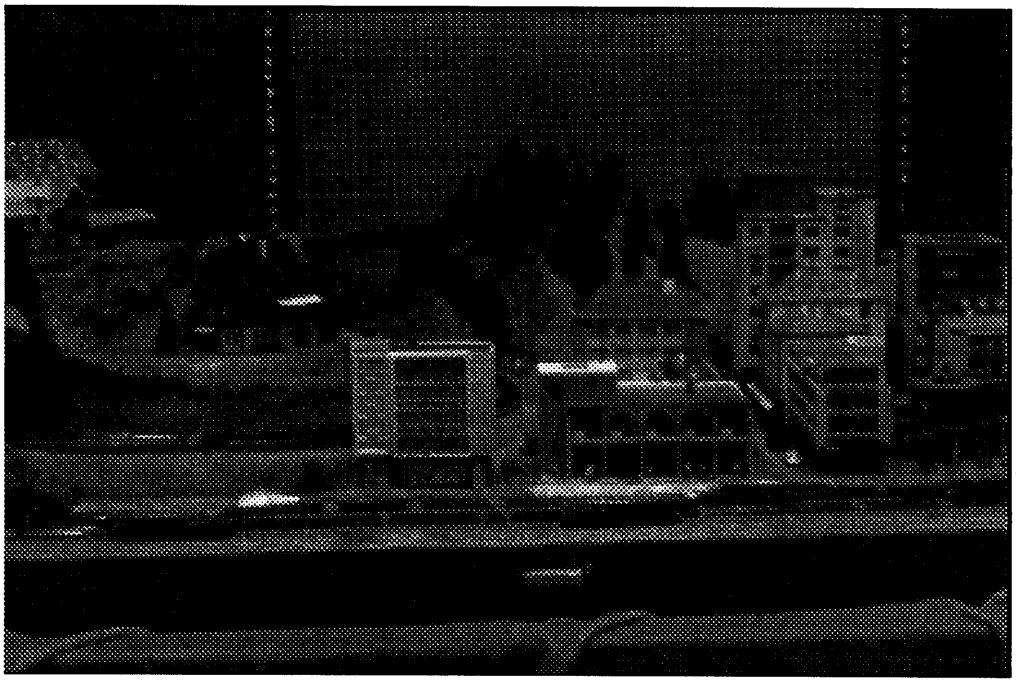

Figure 9: The constructed landscape image, *second tracked* image.

considered reliable anyway.

4.3 Gradient maps of the tracked images

The gradient maps are good measures for studying the quality and characteristics of a tracked image sequence. This section examines the gradient maps of the tracked image sequence constructed from the *landscape* original real image sequence.

The combination of the first original image in fig. 4 and the second tracked image in fig. 9 form the tracked image sequence. The corresponding spatial gradient maps in figures 10 and 11 show that these gradients contain valuable information. The vertical and horizontal lines and features of the original images are implicitly represented in these spatial gradients.

The temporal gradient map of the tracked image sequence is shown in fig. 12. This map contains very important information. First of all it clearly shows the characteristic of the tracked image sequence. Both the horizontal and vertical features of the image sequence become more obvious as their distance from the tracking point location (image center in this case) increases. Secondly, the appearance of the horizontal and vertical lines here provides hints about the existence of a rotational component about the fixation axis. And finally the dominant vertical lines are an indication that the equivalent rotational velocity has a major component about the vertical axis.

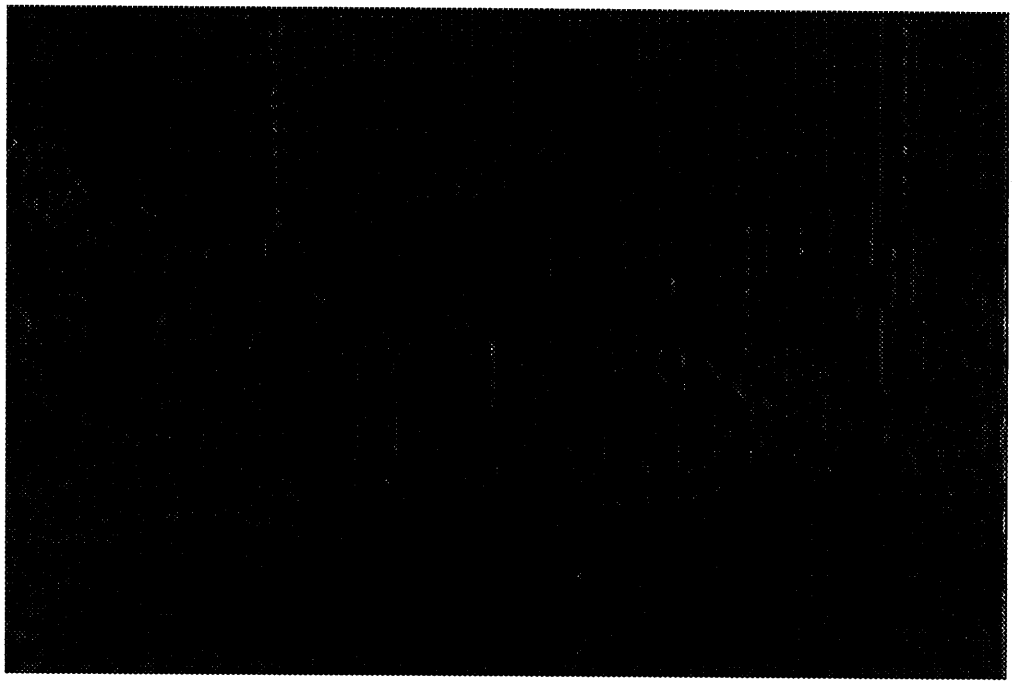

Figure 10: The spatial gradient map E_x of the *tracked* landscape image sequence in *horizontal direction.*

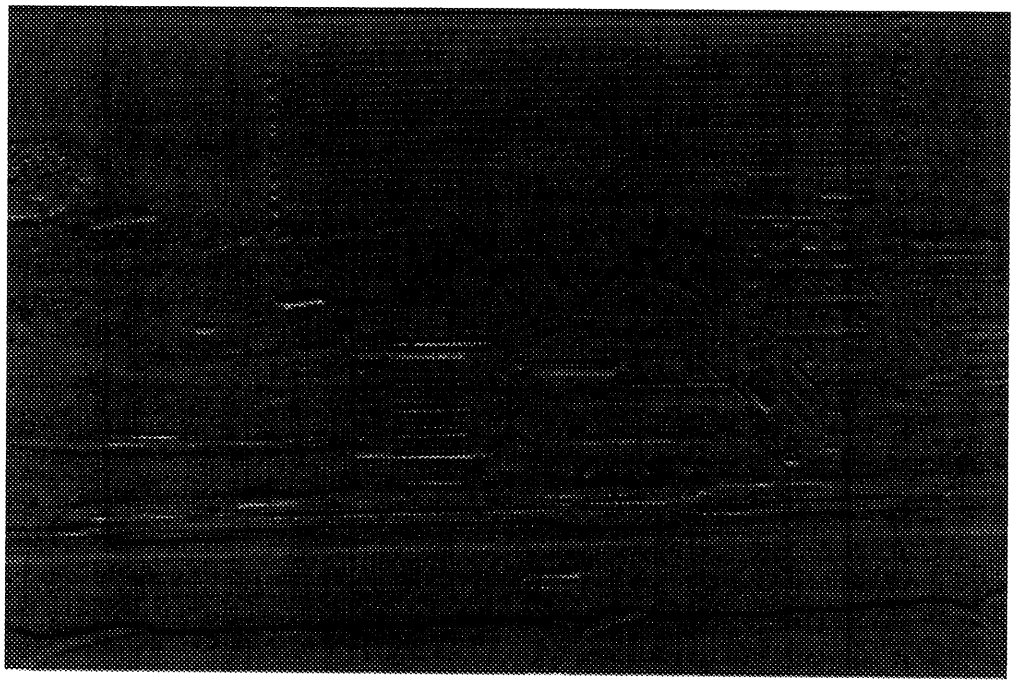

Figure 11: The spatial gradient maps of the *tracked* landscape image sequence in *vertical direction.*

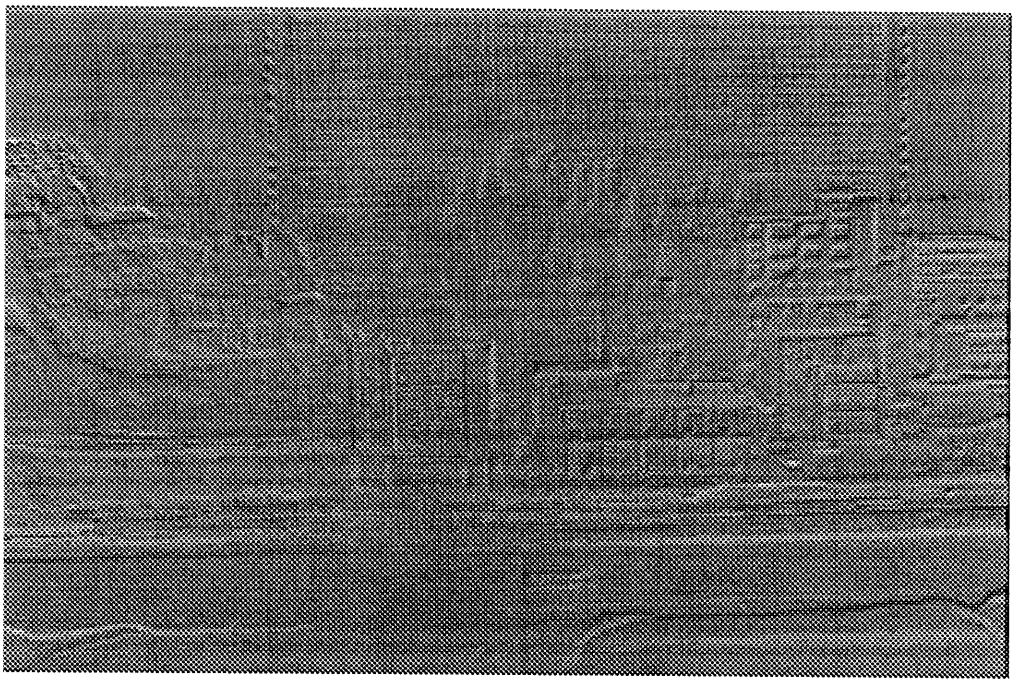

Figure 12: The temporal gradient map of the *tracked* landscape image sequence.

5 Summary

This paper described the *pixel shifting process* and presented the experimental results of constructing a sequence of fixated (tracked) images from an arbitrary image sequence resulting from an arbitrary 3D motion. This method solves the tracking problem in its most challenging case. In other words, it does not require any knowledge about the motion or shape. Furthermore, the tracking point is not restricted to the principal point (image center) and virtually any point can be chosen as the tracking point. Our technique is neither a simple 2D tracking nor an image feature alignment.

Our tracking technique is performed completely in software without any need to mechanically move the camera relative to the vehicle for tracking. It is computationally simple and inexpensive. It uses neither optical flow nor feature correspondence. Instead, brightness gradients of the original input images are used directly.

The quality of the tracked images are examined using spatio-temporal gradients which implicitly capture not only the features of the scene but also preserve the characteristics of the involved motion.

Appendix: Computation of Brightness Gradients

The spatial and temporal derivatives of the image brightnesses are the basic data blocks in the direct methods. This appendix describes the formulations behind the estimation of the brightness gradients in images [10, 9].

The spatial brightness gradients E_x, E_y, and temporal brightness gradient E_t are computed simply by using the first differences of image brightness values on a cubic grid; see fig. 13.

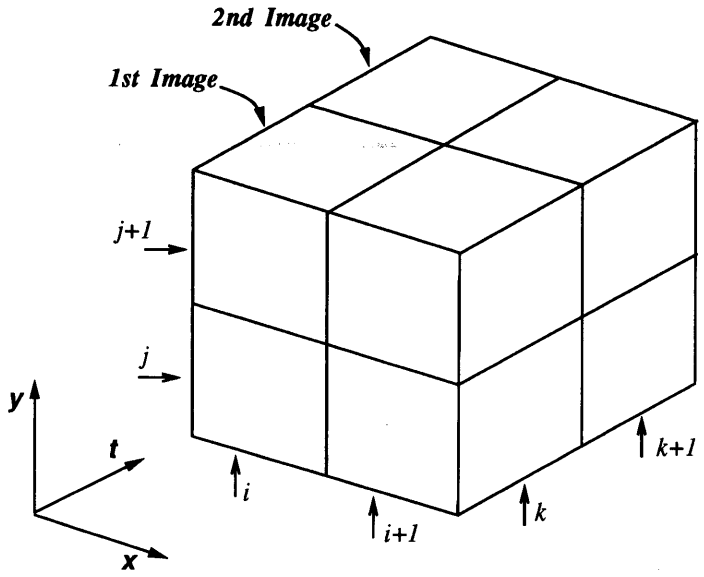

Figure 13: The first brightness derivatives required in the direct methods can be estimated using first differences in a $2 \times 2 \times 2$ cube of brightness values. The estimates apply to the point where four neighboring pixels in an image meet, and at a time halfway between two successive images.

Using the indices i, j, and k to represent x, y, and time t respectively, the estimates of spatial gradients E_x and E_y are give by:

$$E_x \approx \frac{1}{4\delta x}((E_{i+1,j,k} + E_{i+1,j,k+1} + E_{i+1,j+1,k} + E_{i+1,j+1,k+1}) - (E_{i,j,k} + E_{i,j,k+1} + E_{i,j+1,k} + E_{i,j+1,k+1})), \quad (6)$$

and

$$E_y \approx \frac{1}{4\delta y}((E_{i,j+1,k} + E_{i,j+1,k+1} + E_{i+1,j+1,k} + E_{i+1,j+1,k+1}) - (E_{i,j,k} + E_{i,j,k+1} + E_{i+1,j,k} + E_{i+1,j,k+1})), \quad (7)$$

and the temporal gradient E_t is

$$E_t \approx \frac{1}{4\delta t}((E_{i,j,k+1} + E_{i,j+1,k+1} + E_{i+1,j,k+1} + E_{i+1,j+1,k+1}) - (E_{i,j,k} + E_{i,j+1,k} + E_{i+1,j,k} + E_{i+1,j+1,k})). \quad (8)$$

These formulations give the brightness gradients at a point lying between four neighboring pixels, and between successive images.

Considering the fact that we perform spatial tessellation by using pixels and temporal tessellation by employing individual time varying frames, the above algorithms compensate for part of the tessellation errors involved in discrete digitized images.

Acknowledgments

The author would like to thank Professors Berthold Horn and Eric Grimson for their valuable comments on this work.

References

[1] I.E. Abdou and K.Y. Wong. Analysis of linear interpolation schemes for bi-level image applications. *IBM Jour. of Research and Develop.*, 26(6):667–686, Nov. 1982.

[2] J. Aloimonos and D.P. Tsakiris. On the mathematics of visual tracking. Technical Report CAR-TR-390, Computer Vision Laboratory, University of Maryland, MD, Sep. 1988.

[3] A. Bandopadhay, B. Chandra, and D.H. Ballard. Active navigation: Tracking an environmental point considered beneficial. In *Proc. of IEEE Workshop on Motion: Representation and Analysis*, pages 23–29, Kiawash Island, May 7-9 1986.

[4] R. Bernstein. Digital image processing of earth observation sensor data. *IBM Journal of Research and Development*, pages 40–57, Jan. 1976.

[5] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation*, 8(3):313–326, June 1992.

[6] J.T. Feddema, C.S.G. Lee, and O.R. Mitchell. Automatic selection of image features for visual servoing of a robot manipulator. In *Proceedings of IEEE Int'l Conf. on Robotics and Automation*, pages 832–837, May 1989.

[7] A.L. Gilbert, M.K. Giles, G.M. Flachs, R.B. Rogers, and Y.H. U. A real-time video tracking system. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(1):47–56, Jan. 1980.

[8] R.C. Harrell, D.C. Slaughter, and P.D. Adsit. A fruit-tracking system for robotic harvesting. *Machine Vision and Applications*, 2:69–80, 1989.

[9] B.K.P. Horn. *Robot Vision.* McGraw Hill and MIT Press, Cambridge, MA, 1986.

[10] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[11] R.G. Keys. Cubic convolution interpolation for digital image processing. *IEEE Trans. Acoustics, Speech and Signal Processing*, 29(6):1153–1160, Dec. 1981.

[12] R.C. Luo, R.E. Mullen Jr., and D.E. Wessel. An adaptive robotic tracking system using optical flow. In *Proceedings of IEEE Int'l Conf. on Robotics and Automation*, pages 568–573, 1988.

[13] S. Moskowitz. Terminal guidance by pattern recognition: A new approach. *IEEE Trans. on Aeros. Navig. Elecron.*, pages 254–265, Dec. 1964.

[14] N. Papanikolopoulos, P.K. Khosla, and T. Kanade. Vision and control techniques for robotic visual tracking. In *Proceedings of IEEE Int'l Conf. on Robotics and Automation*, pages 857–864, Apr. 1991.

[15] S.S. Rifman and D.M. McKinnon. Evaluation of digital correction techniques for ERTS images. Technical Report TRW 20634-6003-TU-00, NASA Goddard Space Flight Center.

[16] S.S. Rifman and D.M. McKinnon. Evaluation of digital correction techniques for ERTS images. Technical Report E74-10792, TRW Systems Group, July 1974.

[17] G. Sandini and M. Tistarelli. Active tracking strategy for monocular depth inference over multiple frames. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1):13–27, Jan. 1990.

[18] R.J. Schalkoff and E.S. McVey. A model and tracking algorithm for a class of video targets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 4(1):2–10, Jan. 1982.

[19] M.A. Taalebinezhaad. Autonomous fixation. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, pages 744–747, Champaign, Illinois, Nov. 1992.

[20] M.A. Taalebinezhaad. Direct recovery of motion and shape in the general case by fixation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(8):847–853, Aug. 1992.

[21] M.A. Taalebinezhaad. Robot motion vision by fixation. Technical Report AI-TR 1384, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Mass., Sep. 1992.

[22] M.A. Taalebinezhaad. Towards autonomous motion vision. Memo AIM 1334, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, Apr. 1992.

[23] W.B. Thompson. Structure-from-motion by tracking occlusion boundaries. *Biological Cybernetics*, 62:113–116, 1989.

[24] L.E. Weiss, A.C. Sanderson, and C.P. Neuman. Dynamic sensor based control of robots with visual feedback. *IEEE Transactions on Robotics and Automation*, 3(5):404–417, Oct. 1987.