A Unified Approach to Moving Object Detection in 2D and 3D Scenes[1]

Michal Irani P. Anandan

David Sarnoff Research Center
CN5300, Princeton, NJ 08543-5300, U.S.A.
Email: {michal,anandan}@sarnoff.com

Abstract

The detection of moving objects in important in many tasks. Previous approaches to this problem can be broadly divided into two classes: 2D algorithms which apply when the scene can be approximated by a flat surface and/or when the camera is only undergoing rotations and zooms, and 3D algorithms which work well only when significant depth variations are present in the scene and the camera is translating. In this paper, we describe a unified approach to handling moving object detection in both 2D and 3D scenes, with a strategy to gracefully bridge the gap between those two extremes. Our approach is based on a stratification of the moving object detection problem into scenarios which gradually increase in their complexity. We present a set of techniques that match the above stratification. These techniques progressively increase in their complexity, ranging from 2D techniques to more complex 3D techniques. Moreover, the computations required for the solution to the problem at one complexity level become the initial processing step for the solution at the next complexity level. We illustrate these techniques using examples from real image sequences.

1 Introduction

Moving object detection is an important problem in image sequence analysis. It is necessary for surveillance applications, for guidance of autonomous vehicles, for efficient video compression, for smart tracking of moving objects, and many other applications.

[1]This work was supported by ARPA under contract DAAA15-93-C-0061

The 2D motion observed in an image sequence is caused by 3D camera motion (the *ego-motion*) and by 3D motions of independently moving objects. The key step in moving object detection is accounting for (or compensating for) the camera-induced image motion. After compensation for camera-induced image motion, the remaining residual motions must be due to moving objects.

The camera induced image motion depends both on the ego-motion parameters and the depth of each point in the scene. Estimating all of these physical parameters (namely ego-motion and depth) to account for the camera-induced motion is, in general, an inherently ambiguous problem [2]. When the scene contains large depth variations, these parameters may be recovered. We refer to these scenes as *3D scenes*. However, in *2D scenes*, namely when the depth variations are not significant, the recovery of the camera and scene parameters is usually not robust or reliable [2]. Sample publications that treat the problem of moving objects in 3D scenes are [3, 17, 25, 26, 8]. A careful treatment of the issues and problems associated with moving object detection in 3D scenes is given in [24].

An effective approach to accounting for camera induced motion in 2D scenes is to model the image motion in terms of a global 2D parametric transformation. This approach is robust and reliable when applied to flat (planar) scenes, distant scenes, or when the camera is undergoing only rotations and zooms. However, the 2D approach cannot be applied to the 3D scenes. Examples of methods that handle moving objects in 2D scenes are [11, 6, 7, 9, 23, 19, 27, 4].

Therefore, 2D algorithms and 3D algorithms address the moving object detection problem in very different types of scenarios. These are two extremes in a continuum of scenarios: flat 2D scenes (i.e., *no 3D parallax*) vs. 3D scenes with dense depth variations (i.e., *dense 3D parallax*). Both classes fail on the other extreme case or even on the intermediate case (when 3D parallax is *sparse* relative to amount of independent motion).

In real image sequences it is not always possible to predict in advance which situation (2D or 3D) will occur. Moreover, both types of scenarios can occur within the same sequence, with gradual transitions between them. Unfortunately, no single class of algorithms (2D or 3D) can address the general moving object detection problem. It is not practical to

constantly switch from one set of algorithms to another, especially since neither class treats well the intermediate case.

In this paper, we present a unified approach to handling moving object detection in both 2D and 3D scenes, with a strategy to gracefully bridge the gap between those two extremes. Our approach is based on a stratification of the moving object detection problem into scenarios which gradually increase in their complexity: (i) scenarios in which the camera induced motion can be modeled by a single 2D parametric transformation, (ii) those in which the camera induced motion can be modeled in terms of a small number of *layers* of parametric transformations, and (iii) general 3D scenes, in which a more complete parallax motion analysis is required.

We present a set of techniques that match the above stratification. These techniques progressively increase in their complexity, ranging from 2D techniques to more complex 3D techniques. Moreover, the computations required for the solution to the problem at one complexity level become the initial processing step for the solution at the next complexity level. In particular, the 2D parametric motion compensation forms the basis to the solution of the multiple layer situation, and the single 2D or multiple-layer motion compensation forms the basis to the solution of the more general 3D case. Careful treatment is given to the intermediate case, when 3D parallax motion is sparse relative to amount of independent motion.

The goal in taking this approach is to develop a strategy for moving object detection, so that the analysis performed is tuned to match the complexity of the problem and the availability of information at any time. This paper describes the core elements of such a strategy. The integration of these elements into a single algorithm remains a task for our future research.

2 2D Scenes

When the scene viewed from a moving camera is at such a distance that it can be approximated by a flat 2D surface, then the camera induced motion can be modeled by a *single*

global 2D parametric transformation between a pair of successive image frames:

$$\begin{bmatrix} u(x,y) \\ v(x,y) \end{bmatrix} = \begin{bmatrix} p_1 x + p_2 y + p_5 + p_7 x^2 + p_8 xy \\ p_3 x + p_4 y + p_6 + p_7 xy + p_8 y^2 \end{bmatrix} \quad (1)$$

where $(u(x,y), v(x,y))$ denotes the image velocity at the point (x,y). The above equation is an exact description of the instantaneous image motion field induced by a planar surface viewed from a moving camera. In addition, this transformation also describes well the 2D image motion of an *arbitrary* 3D scene undergoing camera rotations, zooms, and small camera translations. More importantly, when the overall 3D range (Z) to the scene from the camera is much greater than the variation of the range within the scene (ΔZ), the above describes the image motion field to sub-pixel accuracy.

We refer to scenes that satisfy one or more of the abovementioned conditions (and hence Equation (1) is applicable), as *2D scenes*. In practice, these conditions are often satisfied in remote surveillance applications, when narrow field-of-view (FOV) cameras (typically 5° or less) are used to detect moving objects in a distant scene (typically at least $1km$ away).

Under these conditions, we can use a previously developed method [5, 11] in order to compute the $2D$ parametric motion. This technique "locks" onto a "dominant" parametric motion between an image pair, even in the presence of independently moving objects. It does not require prior knowledge of their regions of support in the image plane [11]. This computation provides only the $2D$ motion parameters of the camera-induced motion, but no explicit $3D$ shape or motion information.

Once the dominant 2D parametric motion has been estimated, it is used for warping one image towards the other. When the dominant motion is that of the camera, all regions corresponding to static portions of the scene are in completely aligned as a result of the 2D registration (except for non-overlapping image boundaries), while independently moving objects are not. Detection of moving objects is therefore performed by determining local misalignments [11] after the global 2D parametric registration.

Figure 1 shows an example of moving object detection in a "2D scene". This sequence was obtained by a video camera with a FOV of 4 degrees. The camera was mounted on a vehicle moving on a bumpy dirt road at about 15 km/hr and was looking sideways. Therefore,

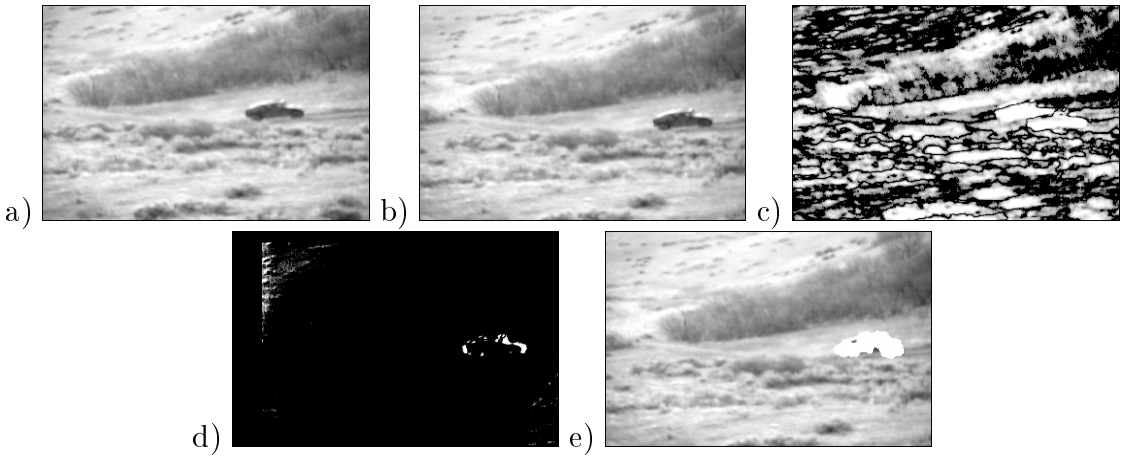

Figure 1: 2D moving object detection.
(a-b) Two frames in a sequence obtained by a translating and rotating camera. The scene itself was not planar, but was distant enough (about 1 km away from the camera) so that effects of 3D parallax were negligible. The scene contained a car driving on a road. (c) Intensity differences before dominant (background) 2D alignment. (d) Intensity differences after dominant (background) 2D alignment. Non-overlapping image boundaries were not processed. The 2D alignment compensates for the camera-induced motion, but not for the car's independent motion. (e) The detected moving object based on local misalignment *analysis. The white region signifies the detected moving object.*

the camera was both translating and rotating (camera jitter). The scene itself was not planar, but was distant enough (about 1 km away from the camera), so that 2D parametric transformations were sufficient to account for the camera-induced motion between successive frames. The scene contained a car moving independently on a road. Figure 1.a and Figure 1.b show two frames out of the sequence. Figure 1.c and Figure 1.d show intensity differences before and after dominant (background) 2D alignment, respectively. Figure 1.e shows the detected moving object based on local misalignment analysis [11].

The frame-to-frame motion of the background in remote surveillance applications can typically be modeled by a 2D parametric transformation. However, when a frontal portion of the scene enters the FOV, effects of 3D parallax motion are encountered. The simple 2D algorithm cannot account for camera-induced motion in scenes with 3D parallax. In the next two sections we address the problem of moving object detection in 3D scenes *with* parallax.

3 Multi-Planar Scenes

When the camera is translating, and the scene is not planar or is not sufficiently distant, then a *single* 2D parametric motion (Section 2) is insufficient for modeling the camera-induced

motion. Aligning two images with respect to a *dominant* 2D parametric transformation may bring into alignment a large portion of the scene, which corresponds to a planar (or a remote) part of the scene. However, any other (e.g., near) portions of the scene that enter the field-of-view cannot be aligned by the dominant 2D parametric transformation. These out-of-plane scene points, although they have the same 3D motion as the planar points, have substantially different induced 2D motions. The *differences* in 2D motions are called *3D parallax motion* [18, 20]. Effects of parallax are only due to camera translation and 3D scene variations. Camera rotation or zoom do not cause parallax (see Section 4.1).

Figure 2 shows an example of a sequence where the effects of 3D parallax are evident. Figure 2.a and 2.b show two frames from a sequence with the same setting and scenario described in Figure 1, only in this case a frontal hill with bushes (which was much closer to the camera than the background scene) entered the field of view (FOV).

Figure 2.c displays image regions which were found to be aligned after *dominant* 2D parametric registration (see Section 2). Clearly the global 2D alignment accounts for the camera-induced motion of the distant portion of the scene, but does *not* account for the camera-induced motion of the closer portion of the scene (the bushes).

Thus, simple 2D techniques, when applied to these types of scenarios, will not be able to distinguish between the independent car motion and the 3D parallax motion of the bush. There is therefore a need to model 3D parallax as well. In this section we describe one approach to modeling parallax motion, which builds on top of the 2D approach to modeling camera- induced motion. This approach is based on fitting multiple planar surfaces (i.e., multiple 2D "layers" [1, 27] to the scene. In Section 4 approaches to handling more *complex* types of scenes with (sparse and dense) 3D parallax will be described. They too build on top of the 2D (or layered) approach.

When the scene is piecewise planar, or is constructed of a few distinct portions at different depths, then the camera-induced motion can be accounted for by a few *layers* of 2D parametric transformations. This case is very typical of outdoor surveillance scenarios, especially when the camera FOV is narrow. The multi-layered approach is an extension of the simple 2D approach, and is implemented using a method similar to the sequential method presented

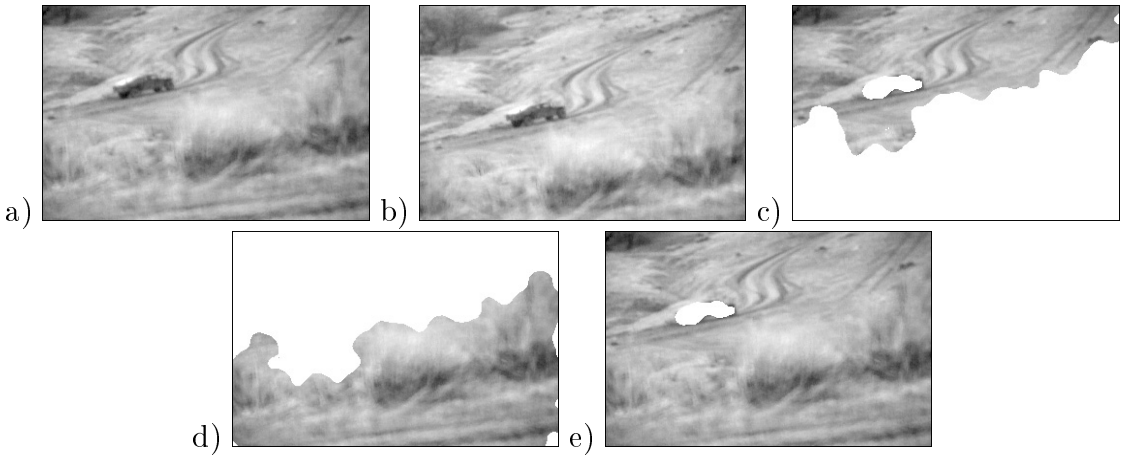

Figure 2: Layered moving object detection.
(a,b) Two frames in a sequence obtained by a translating and rotating camera. The FOV captures a distant portion of the scene (hills and road) as well as a frontal portion of the scene (bushes). The scene contains a car driving on a road. (c) The image region which corresponds to the dominant *2D parametric transformation. This region corresponds to the remote part of the scene. White regions signify image regions which were misaligned after performing global image registration according to the computed dominant 2D parametric transformation. These regions correspond to the car and the frontal part of the scene (the bushes). (d) The image region which corresponds to the* next *detected dominant 2D parametric transformation. This region corresponds to the frontal bushes. The 2D transformation was computed by applying the 2D estimation algorithm again, but this time only to the image regions highlighted in white in Fig. 2.c (i.e., only to image regions inconsistent in their image motion with the* first *dominant 2D parametric transformation). White regions in this figure signify regions inconsistent with the bushes' 2D transformation. These correspond to the car and to the remote parts of the scene. (e) The detected moving object (the car) highlighted in white.*

in [11]: First, the dominant 2D parametric transformation between two frames is detected (Section 2). The two images are aligned accordingly, and the misaligned image regions are detected and segmented out (Figure 2.c). Next, the *same* 2D motion estimation technique is re-applied, but this time only to the segmented (misaligned) regions of the image, to detect the *next* dominant 2D transformation and its region of support within the image, and so on. For each additional layer, the two images are aligned according to the 2D parametric transformation of that layer, and the misaligned image regions are detected and segmented out (Figure 2.d).

Each "2D layer" is continuously tracked in time by using the obtained segmentation masks. Moving objects are detected as image regions that are inconsistent with the image motion of any of the 2D layers. Such an example is shown in Figure 2.e.

A moving object is not detected as a layer by this algorithm if it is small. However, if the object is large, it may itself be detected as a 2D layer. A few cues can be used to distinguish

between moving objects and static scene layers:

1. Moving objects produce discontinuities in 2D motion everywhere on their boundary, as opposed to static 2D layers. Therefore, if a moving object is detected as a layer, it can be distinguished from real scene layers due to the fact that it appears "floating" in the air (i.e., has depth discontinuities all around it). A real scene layer, on the other hand, is always connected to another part of the scene (layer). On the connecting boundary, the 2D motion is continuous. If the connection to other scene portions is outside the FOV, then that layer is adjacent to the *image* boundary. Therefore, a 2D layer which is fully contained in the FOV, and exhibits 2D motion discontinuities all around it, is necessarily a moving object.

2. 3D-consistency over time of two 2D layers can be checked. In Section 4.2 we present a method for checking 3D-consistency of two scene points over time based on their parallax displacements alone. If two layers belong to a single rigid scene, the parallax displacement of one layer with respect to the other is yet another 2D parametric transformation (which is obtained by taking the difference between the two 2D parametric layer transformations). Therefore, for example, consistency of two layers can be verified over time by applying the 3D-consistency check to parallax displacements of one layer with respect to the other (see Section 4.2).

3. Other cues, such detecting negative depth, etc. can also be used.

In the sequence shown in Figures 1 and 2, we used the first cue (i.e., eliminated "floating" layers) to ensure moving objects were not interpreted as scene layers. The moving car was successfully and continuously detected over the entire two-minute video sequence, which alternated between the single-layered case (i.e., no 3D parallax; frontal scene part was not visible in the FOV) and the two-layered case (i.e., existence of 3D parallax).

4 Scenes With General 3D Parallax

While the single and multi-layered parametric registration methods are adequate to handle a large number of situations, there are cases when the parallax cannot be modeled in terms of layers. An example of such a situation is a cluttered scene which contains many small objects at multiple depths (these could be urban scenes or indoor scenes). In this section

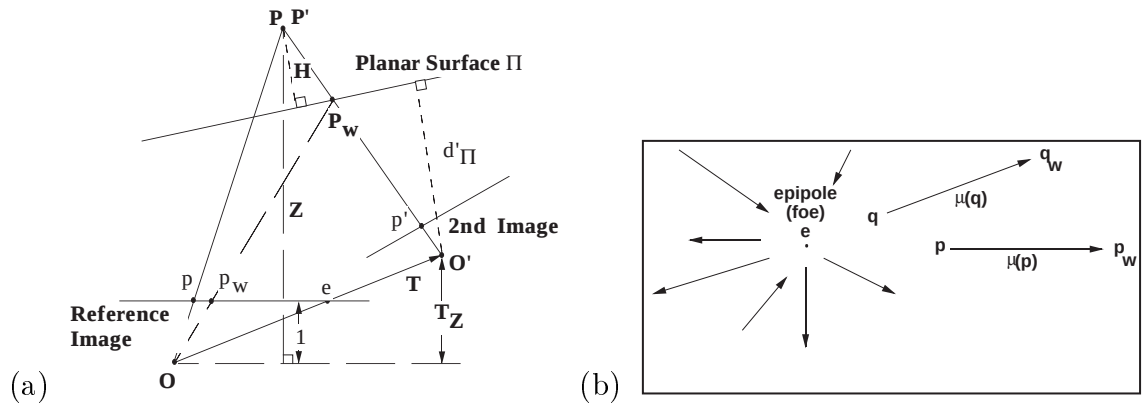

Figure 3: The plane+parallax decomposition.
(a) The geometric interpretation. (b) The epipolar field of the residual parallax displacements.

we develop an approach to handling these more complex 3D scenes.

4.1 3D Scenes with Dense Parallax

The key observation that enables us to extend the 2D parametric registration approach to general 3D scenes is the following: the plane registration process (using the dominant 2D parametric transformation) removes all effects of camera rotation, zoom, and calibration, *without explicitly computing them* [12, 14, 21, 22]. The residual image motion after the plane registration is due only to the *translational* motion of the camera and to the *deviations* of the scene structure from the planar surface. Hence, the residual motion is an *epipolar flow field*. This observation has led to the so-called "plane+parallax" approach to 3D scene analysis [13, 12, 14, 21, 22].

The Plane+Parallax Decomposition:

Figure 3 provides a geometric interpretation of the planar parallax. Let $\vec{\mathbf{P}} = (X, Y, Z)^T$ and $\vec{\mathbf{P}}' = (X', Y', Z')^T$ denote the Cartesian coordinates of a scene point with respect to two different camera views, respectively. Let $\vec{\mathbf{p}} = (x, y)^T$ and $\vec{\mathbf{p}}' = (x', y')^T$ respectively denote the corresponding coordinates of the corresponding image points in the two image frames. Let $\vec{\mathbf{T}} = (T_x, T_y, T_z)$ denote the camera translation between the two views. Let Π denote a (*real* or *virtual*) planar surface in the scene which is registered by the 2D parametric registration process mentioned in Section 2. It can be shown (see [15, 12, 21, 22]) that the 2D image

displacement of the point $\vec{\mathbf{P}}$ can be written as

$$\vec{\mathbf{u}} = (\vec{\mathbf{p}'} - \vec{\mathbf{p}}) = \vec{\mathbf{u}_\pi} + \vec{\mu}, \tag{2}$$

where $\vec{\mathbf{u}_\pi}$ denotes the *planar* part of the $2D$ image motion (the homography due to Π), and $\vec{\mu}$ denotes the residual *planar parallax* $2D$ motion. The homography due to Π results in an image motion field that can be modeled as a 2D parametric transformation. In general, this transformation is a *projective transformation*, however, in the case of instantaneous camera motion, it can be well approximated by the quadratic transformation shown in Equation (1). When $T_z \neq 0$:

$$\vec{\mathbf{u}_\pi} = (\vec{\mathbf{p}'} - \vec{\mathbf{p}_\mathbf{w}}) \quad ; \quad \vec{\mu} = \gamma \frac{T_z}{d'_\pi} (\vec{\mathbf{e}} - \vec{\mathbf{p}_\mathbf{w}}) \tag{3}$$

where $\vec{\mathbf{p}_\mathbf{w}}$ denotes the image point in the first frame which results from warping the corresponding point $\vec{\mathbf{p}'}$ in the second image, by the 2D parametric transformation of the plane Π. The $2D$ image coordinates of the epipole (or the *focus-of-expansion*, FOE) in the first frame are denoted by $\vec{\mathbf{e}}$, and d'_π is the perpendicular distance from the second camera center to the reference plane (see Figure 3). γ is a measure of the 3D shape of the point $\vec{\mathbf{P}}$. In particular, $\gamma = \frac{H}{Z}$, where H is the perpendicular distance from the $\vec{\mathbf{P}}$ to the reference plane, and Z is the "range" (or "depth") of the point $\vec{\mathbf{P}}$ with respect to the first camera. We refer to γ as the projective 3D structure of point $\vec{\mathbf{P}}$. In the case when $T_z = 0$, the parallax motion μ has a slightly different form: $\mu = \frac{\gamma}{d'_\pi} \vec{\mathbf{t}}$, where $t = (T_x, T_y)^T$.

The use of the plane+parallax decomposition for egomotion estimation is described in [12], and for 3D shape recovery is described in [14, 21]. The *plane+parallax* decomposition is more general than the more traditional decomposition in terms of *rotational* and *translational* motion (and includes the traditional decomposition as a special case). In addition, (i) the planar homography (i.e., the 2D parametric planar transformation) compensates for camera rotation, zoom and other changes in the internal parameters of the camera, (ii) this approach does not require any prior knowledge of the camera internal parameters (in other words, no prior camera calibration is needed), and (iii) the planar homography being a 2D parametric

transformation can be estimated in a more stable fashion than the rotation and translation parameters. In particular, it can be estimated even when the camera field-of-view is limited, the depth variations in the scene are small, and in the presence of independently moving objects (see Section 2).

An algorithm for detecting moving objects based on the plane+parallax decomposition is described in [16]. However, it should be noted that in general, the detection of moving objects does not require the estimation of the 3D shape. Since the residual parallax displacements are due to the camera translational component alone, they form a radial field centered at the epipole/FOE (see Figure 3.b). If the epipole is recovered, all that is required for detecting moving objects is the verification whether the residual 2D displacement associated with a given point is directed towards/away from the epipole. This is known as the *epipolar constraint* [24]. Residual 2D motion that violates this requirement can only be due to an independently moving object. Figure 4.a graphically illustrates this situation.

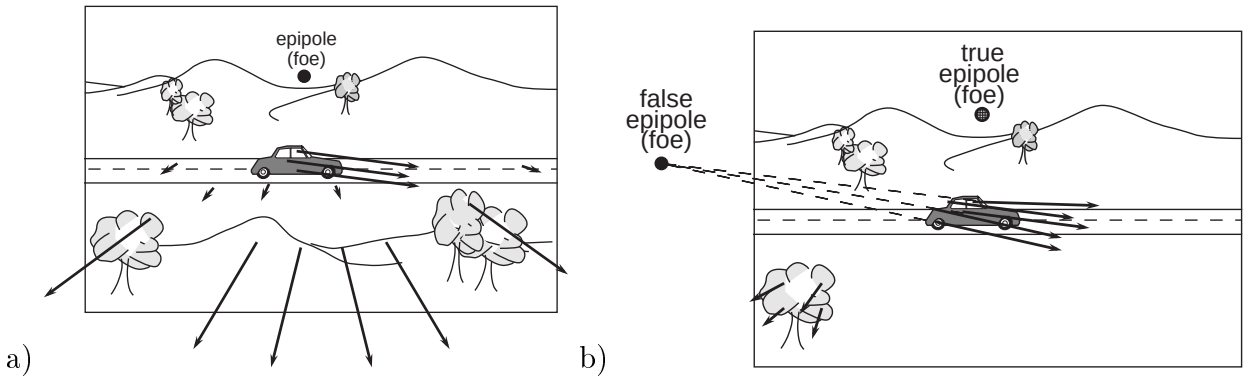

Figure 4: *(a) Moving object detection based on inconsistency of parallax motion with radial epipolar motion field. (b) False epipole estimation when 3D parallax is sparse relative to independent motion.*

Difficulty of Epipole Recovery:

While the plane+parallax strategy works generally well when the epipole (FOE) recovery is possible, its performance depends critically on the ability to accurately estimate the epipole. Since the epipole recovery is based on the residual motion vectors, those vectors that are due to the moving object are likely to bias the estimated epipole away from the true epipole. (Note that this is true even of the "direct" methods that do not explicitly recover the residual motion vectors, but instead rely on spatiotemporal image gradients [14], since the informa-

tion provided by the points on moving objects will influence the estimate.)

The problem of estimating the epipole is acute when the scene contains sparse parallax information and the residual motion vectors due to independently moving object are significant (either in magnitude or in number). A graphic illustration of such a situation is provided in Figure 4.b. In the situation depicted in this figure, the magnitude and the number of parallax vectors on the tree is considerably smaller than the residual motion vectors on the independently moving car. As a result, the estimated epipole is likely to be consistent with the motion of the car (in the figure this would be somewhere outside the field-of-view on the left side of the image) and the tree will be detected as an independently moving object.

There are two obvious ways to overcome the difficulties in estimating the epipole. The first is to use prior knowledge regarding the camera/vehicle motion to reject potential outliers (namely the moving objects) during the estimation. However, if only limited parallax information is available, any attempt to refine this prior information will be unstable. A more general approach would be to defer, or even completely eliminate, the computation of the epipole. In the next section, we develop an approach to moving object detection by directly comparing the parallax motion of pairs of points *without estimating the epipole.*

4.2 3D Scenes With Sparse Parallax

In this section we present a method we have developed for moving object detection in the difficult "intermediate" cases, when 3D parallax information is sparse relative to independent motion information. This approach can be used to bridge the gap between the 2D cases and the dense 3D cases.

The parallax based shape constraint:

Theorem 1: Given the planar-parallax displacement vectors $\vec{\mu_1}$ and $\vec{\mu_2}$ of two points that belong to the static background scene, their *relative 3D projective structure* $\frac{\gamma_2}{\gamma_1}$ is given by:

$$\frac{\gamma_2}{\gamma_1} = \frac{\vec{\mu_2}^T (\Delta\vec{\mathbf{p}_\mathbf{w}})_\perp}{\vec{\mu_1}^T (\Delta\vec{\mathbf{p}_\mathbf{w}})_\perp}. \tag{4}$$

where, as shown in Figure 5, $\vec{\mathbf{p_1}}$ and $\vec{\mathbf{p_2}}$ are the image locations (in the reference frame) of two points that are part of the static scene, $\Delta\vec{\mathbf{p_w}} = \vec{\mathbf{p_{w_2}}} - \vec{\mathbf{p_{w_1}}}$, the vector connecting the "warped" locations of the corresponding second frame points (as in Equation (3)), and $\vec{v}_\perp$ signifies a vector perpendicular to $\vec{v}$.

Proof: See Appendix (also see [10]). ∎

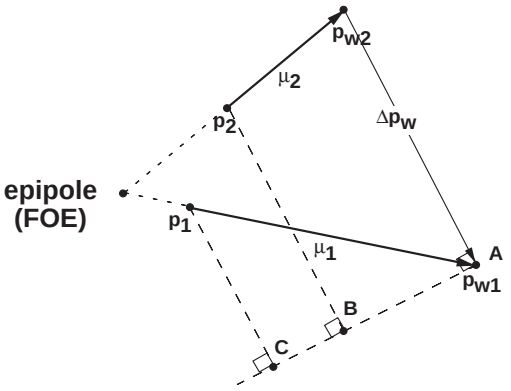

Figure 5: The pairwise parallax-based shape constraint. *This figure geometrically illustrates the relative structure constraint (Eq. 4):* $\frac{\gamma_2}{\gamma_1} = \frac{\vec{\mu_2}^T(\Delta\vec{\mathbf{p}}_\mathbf{w})_\perp}{\vec{\mu_1}^T(\Delta\vec{\mathbf{p}}_\mathbf{w})_\perp} = \frac{\mathbf{AB}}{\mathbf{AC}}$.

Note that this constraint directly relates the relative projective structure of two points to their parallax displacements alone: no camera parameters, in particular the *epipole* (FOE), are involved. Neither is any additional parallax information required at other image points. Application of this constraint to the recovery of 3D structure of the scene is described in [10]. Here we focus on its application to moving object detection.

The parallax based rigidity constraint:

Theorem 2: Given the planar-parallax displacement vectors of two points that belong to the background static scene over *three* frames, the following constraint must be satisfied:

$$\frac{\vec{\mu_2}^{\mathbf{j^T}}(\Delta\vec{\mathbf{p}}_\mathbf{w})^\mathbf{j}_\perp}{\vec{\mu_1}^{\mathbf{j^T}}(\Delta\vec{\mathbf{p}}_\mathbf{w})^\mathbf{j}_\perp} - \frac{\vec{\mu_2}^{\mathbf{k^T}}(\Delta\vec{\mathbf{p}}_\mathbf{w})^\mathbf{k}_\perp}{\vec{\mu_1}^{\mathbf{k^T}}(\Delta\vec{\mathbf{p}}_\mathbf{w})^\mathbf{k}_\perp} = 0. \tag{5}$$

where $\vec{\mu^\mathbf{j}_1}, \vec{\mu^\mathbf{j}_2}$ are the parallax displacement vectors of the two points between the reference frame and the jth frame, $\vec{\mu^\mathbf{k}_1}, \vec{\mu^\mathbf{k}_2}$ are the parallax vectors between the reference frame and the kth frame, and $(\Delta\vec{\mathbf{p}}_\mathbf{w})^j, (\Delta\vec{\mathbf{p}}_\mathbf{w})^k$ are the corresponding distances between the warped points

as in Equation (4) and Figure 5.

Proof: The relative projective structure $\frac{\gamma_2}{\gamma_1}$ is invariant to camera motion. Therefore, using Equation (4), for any two frames j and k we get:

$$\frac{\gamma_2}{\gamma_1} = \frac{\vec{\mu_2}^{\mathbf{j}^{\mathbf{T}}}(\Delta\vec{\mathbf{p}_\mathbf{w}})^{\mathbf{j}}_{\perp}}{\vec{\mu_1}^{\mathbf{j}^{\mathbf{T}}}(\Delta\vec{\mathbf{p}_\mathbf{w}})^{\mathbf{j}}_{\perp}} = \frac{\vec{\mu_2}^{\mathbf{k}^{\mathbf{T}}}(\Delta\vec{\mathbf{p}_\mathbf{w}})^{\mathbf{k}}_{\perp}}{\vec{\mu_1}^{\mathbf{k}^{\mathbf{T}}}(\Delta\vec{\mathbf{p}_\mathbf{w}})^{\mathbf{k}}_{\perp}}. \qquad \blacksquare$$

As in the case of the parallax based shape constraint (Equation (4)), the parallax based rigidity constraint (Equation (5)) relates the parallax vectors of pairs of points over three frames without referring to the *camera geometry* (especially the epipole/FOE). Furthermore, this constraint does not even explicitly refer to the *structure* parameters of the points in consideration. The rigidity constraint (5) can therefore be applied to detect inconsistencies in the $3D$ motion of two image points (i.e., say whether the two image points are projections of $3D$ points belonging to a same or different $3D$ moving objects) based on their *parallax* motion among three (or more) frames alone, without the need to estimate either *camera geometry*, *camera motion*, or *structure* parameters, and without relying on parallax information at other image points. A consistency measure is defined as the left-hand side of Equation (5), after multiplying by the denominators (to eliminate singularities). The farther this quantity is from 0, the higher is the 3D-inconsistency of the two points.

4.3 Applying the Parallax Rigidity Constraint to Moving Object Detection

Fig. 6.a graphically displays an example of a configuration in which estimating the epipole in presence of multiple moving objects can be very erroneous, even when using clustering techniques in the epipole domain as suggested by [17, 25]. Relying on the epipole computation to detect inconsistencies in $3D$ motion fails in detecting moving objects in such cases.

The parallax rigidity constraint (Equation (5)) can be applied to detect inconsistencies in the $3D$ motion of one image point relative to another directly from their "parallax" vectors over multiple (three or more) frames, without the need to estimate either *camera geometry*,

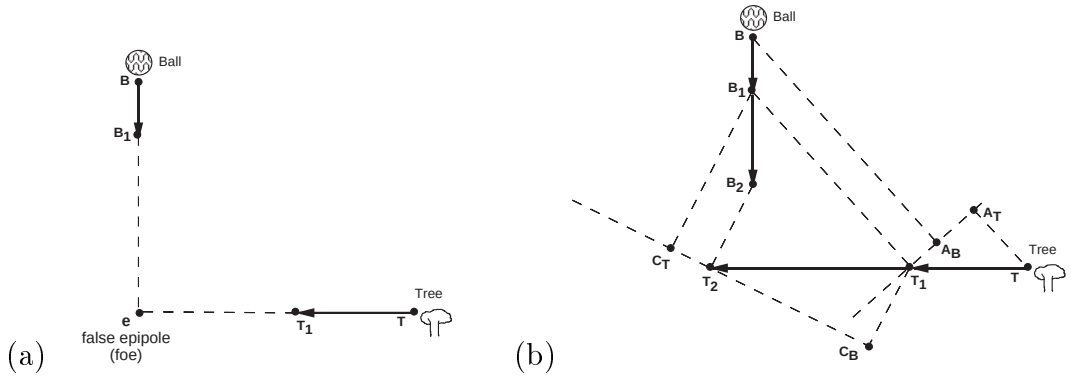

Figure 6: Reliable detection of 3D motion inconsistency with sparse parallax information. *(a) Camera is translating to the right. The only static object with pure parallax motion is that of the tree. Ball is falling independently. The epipole may be incorrectly be computed as* $\mathbf{e}$*. The false epipole* $\mathbf{e}$ *is consistent with both motions. (b) The rigidity constraint applied to this scenario detects 3D inconsistency over three frames, since* $\frac{\mathbf{T_1 A_B}}{\mathbf{T_1 A_T}} \neq \frac{\mathbf{T_2 C_B}}{-\mathbf{T_2 C_T}}$*. In this case, even the signs do not match.*

camera motion, or *shape* parameters. This provides a useful mechanism for clustering (or segmenting) the "parallax" vectors (i.e., the residual motion after planar registration) into consistent groups belonging to consistently $3D$ moving objects, even in cases such as in Fig. 6.a, where the parallax information is minimal, and the independent motion is not negligible. Fig. 6.b graphically explains how the rigidity constraint (5) detects the $3D$ inconsistency of Fig. 6.a over three frames.

Fig. 7 shows an example of using the rigidity-based inconsistency measure described earlier to detect $3D$ inconsistencies. In this sequence the camera is in motion (translating from left to right), inducing parallax motion of different magnitudes on the house, road, and road-sign. The car moves independently from left to right. The detected $2D$ planar motion was that of the house. The planar parallax motion was computed after $2D$ registration of the three images with respect to the house (see Fig. 7.d). A single point on the road-sign was selected as a point of reference (see Fig. 7.e). Fig. 7.f displays the measure of *in*consistency of each point in the image with respect to the selected road-sign point. Bright regions indicate large values when applying the inconsistency measure, i.e., violations in $3D$ rigidity detected over three frames with respect to the road-sign point. The region which was detected as moving $3D$-*in*consistently with respect to the road-sign point corresponds to the car. Regions close to the image boundary were ignored. All other regions of the image

were detected as moving $3D$-consistently with the road-sign point. Therefore, assuming an *uncalibrated* camera, this method provides a mechanism for segmenting all non-zero residual motion vectors (after $2D$ planar stabilization) into groups moving *consistently* (in the $3D$ sense).

Fig. 8 shows another example of using the rigidity constraint (5) to detect $3D$ inconsistencies. In this sequence the camera is mounted on a helicopter flying from left to right, inducing some parallax motion (of different magnitudes) on the house-roof and trees (bottom of the image), and on the electricity poles (by the road). Three cars move independently on the road. The detected $2D$ planar motion was that of the ground surface (see Fig. 8.d). A single point was selected on a tree as a point of reference (see Fig. 8.e). Fig. 8.f displays the measure of *in*consistency of each point in the image with respect to the selected reference point. Bright regions indicate $3D$-inconsistency detected over three frames. The three cars were detected as moving *in*consistently with the selected tree point. Regions close to the image boundary were ignored. All other image regions were detected as moving consistently with the selected tree point.

The ability of the parallax rigidity constraint (Equation (5)) to detect $3D$-inconsistency with respect to a *single* point, provides a natural way to *bridge* between $2D$ algorithms (which assume that any $2D$ motion different than the planar motion is an independently moving object), and $3D$ algorithms (which rely on having prior knowledge of a consistent set of points, or alternatively, dense parallax data).

5 Conclusion

Previous approaches to the problem of moving object detection can be broadly divided into two classes: 2D algorithms which apply when the scene can be approximated by a flat surface and/or when the camera is only undergoing rotations and zooms, and 3D algorithms which work well only when significant depth variations are present in the scene and the camera is translating. These two classes of algorithms treat two extremes in a continuum of scenarios: *no 3D parallax* (2D algorithms) vs. *dense 3D parallax* (3D algorithms). Both classes fail on the other extreme case or even on the intermediate case (when 3D parallax is *sparse* relative

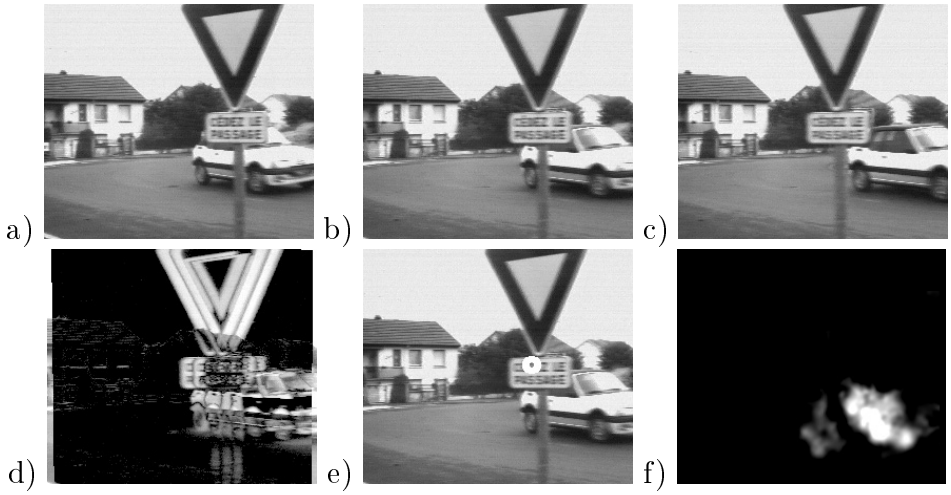

Figure 7: Moving object detection relying on a single parallax vector.
(a,b,c) Three image frames from a sequence obtained by a camera translating from left to right, inducing parallax motion of different magnitudes on the house, road, and road-sign. The car moves independently from left to right. The middle frame (Fig. 7.b) was chosen as the frame of reference. (d) Differences taken after $2D$ *image registration. The detected* $2D$ *planar motion was that of the house, and is canceled by the* $2D$ *registration. All other scene parts that have different* $2D$ *motions (i.e.,* parallax *motion or* independent *motion) are misregistered. (e) The selected point of reference (a point on the road-sign) highlighted by a white circle. (f) The measure of* $3D$-in*consistency of all points in the image with respect to the road-sign point. Bright regions indicate violations in* $3D$ *rigidity detected over three frames with respect to the selected road-sign point. These regions correspond to the car. Regions close to the image boundary were ignored. All other regions of the image appear to move* $3D$*-consistently with the road-sign point.*

to amount of independent motion).

In this paper, we have described a unified approach to handling moving object detection in both 2D and 3D scenes, with a strategy to gracefully bridge the gap between those two extremes. Our approach is based on a stratification of the moving object detection problem into scenarios which gradually increase in their complexity: We presented a set of techniques that match the above stratification. These techniques progressively increase in their complexity, ranging from 2D techniques to more complex 3D techniques. Moreover, the computations required for the solution to the problem at one complexity level become the initial processing step for the solution at the next complexity level.

The goal in taking this approach is to develop a strategy for moving object detection, so that the analysis performed is tuned to match the complexity of the problem and the availability of information at any time. This paper describes the core elements of such a strategy. The integration of these elements into a single algorithm remains a task for our future research.

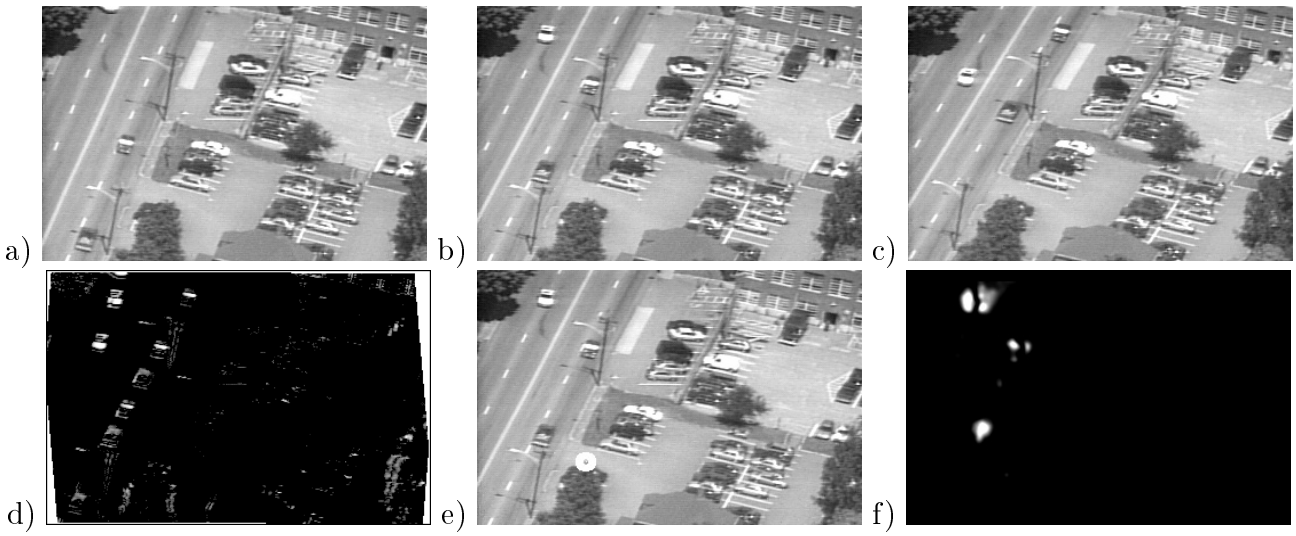

Figure 8: Moving object detection relying on a single parallax vector.
(a,b,c) Three image frames from a sequence obtained by a camera mounted on a helicopter (flying from left to right while turning), inducing some parallax motion (of different magnitudes) on the house-roof and trees (bottom of the image), and on the electricity poles (by the road). Three cars move independently on the road. The middle frame (Fig. 8.b) was chosen as the frame of reference. (d) Differences taken after $2D$ *image registration. The detected* $2D$ *planar motion was that of the ground surface, and is canceled by the* $2D$ *registration. All other scene parts that have different* $2D$ *motions (i.e.,* parallax *motion or* independent *motion) are misregistered. (e) The selected point of reference (a point on a tree at the bottom left of the image) highlighted by a white circle. (f) The measure of* $3D$-in*consistency of each point in the image with the tree point. Bright regions indicate violations in* $3D$ *rigidity detected over three frames with respect to the selected tree point. These regions correspond to the three cars (in the reference image). Regions close to the image boundary were ignored. All other regions of the image appear to move* $3D$*-consistently with the tree point.*

References

[1] E.H. Adelson. Layered representations for image coding. Technical Report 181, MIT Media Lab. Vision and Modeling Group, December 1991.

[2] G. Adiv. Inherent ambiguities in recovering 3D motion and structure from a noisy flow field. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11:477–489, May 1989.

[3] Y. Aloimonos, editor. *Active Perception.* Erlbaum, 1993.

[4] S. Ayer and H. Sawhney. Layered representation of motion video using robust maximum-liklihood estimation of miture models and mdl encoding. In *International Conference on Computer Vision*, pages 777–784, Cambridge, MA, June 1995.

[5] J.R. Bergen, P. Anandan, K.J. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *European Conference on Computer Vision*, pages 237–252, Santa Margarita Ligure, May 1992.

[6] J.R. Bergen, P.J. Burt, R. Hingorani, and S. Peleg. A three-frame algorithm for estimating two-component image motion. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14:886–895, September 1992.

[7] P.J. Burt, R. Hingorani, and R.J. Kolczynski. Mechanisms for isolating component patterns in the sequential analysis of multiple motion. In *IEEE Workshop on Visual Motion*, pages 187–193, Princeton, New Jersey, October 1991.

[8] J. Costeira and T. Kanade. A multi-body factorization method for motion analysis. In *International Conference on Computer Vision*, pages 1071–1076, Cambridge, MA, June 1995.

[9] T. Darrell and A. Pentland. Robust estimation of a multi-layered motion representation. In *IEEE Workshop on Visual Motion*, pages 173–178, Princeton, New Jersey, October 1991.

[10] M. Irani and P. Anandan. Parallax geometry of pairs of points for 3d scene analysis. In *European Conference on Computer Vision*, Cambridge, UK, April 1996.

[11] M. Irani, B. Rousso, and S. Peleg. Computing occluding and transparent motions. *International Journal of Computer Vision*, 12:5–16, February 1994.

[12] M. Irani, B. Rousso, and S. Peleg. Recovery of ego-motion using image stabilization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 454–460, Seattle, Wa., June 1994.

[13] J.J. Koenderink and A.J. van Doorn. Representation of local geometry in the visual system. *Biol. Cybern.*, 55:367 – 375, 1987.

[14] R. Kumar, P. Anandan, and K. Hanna. Direct recovery of shape from multiple views: a parallax based approach. In *Proc 12th ICPR*, 1994.

[15] Rakesh Kumar, P. Anandan, and K. Hanna. Shape recovery from multiple views: a parallax based approach. In *DARPA IU Workshop*, Monterey, CA, November 1994.

[16] Rakesh Kumar, P. Anandan, M. Irani, J. R. Bergen, and K. J. Hanna. Representation of scenes from collections of images. In *Workshop on Representations of Visual Scenes*, 1995.

[17] J.M. Lawn and R. Cipolla. Robust egomotion estimation from affine motion parallax. In *European Conference on Computer Vision*, pages 205–210, May 1994.

[18] H.C. Longuet-Higgins and K. Prazdny. The interpretation of a moving retinal image. *Proceedings of The Royal Society of London B*, 208:385–397, 1980.

[19] F. Meyer and P. Bouthemy. Region-based tracking in image sequences. In *European Conference on Computer Vision*, pages 476–484, Santa Margarita Ligure, May 1992.

[20] J.H. Rieger and D.T. Lawton. Processing differential image motion. *J. Opt. Soc. Am. A*, A2(2):354–359, 1985.

[21] Harpreet Sawhney. 3d geometry from planar parallax. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 1994.

[22] A. Shashua and N. Navab. Relative affine structure: Theory and application to 3d reconstruction from perspective views. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 483–489, Seattle, Wa., June 1994.

[23] M. Shizawa and K. Mase. Principle of superposition: A common computational framework for analysis of multiple motion. In *IEEE Workshop on Visual Motion*, pages 164–172, Princeton, New Jersey, October 1991.

[24] W.B. Thompson and T.C. Pong. Detecting moving objects. *International Journal of Computer Vision*, 4:29–57, 1990.

[25] P.H.S. Torr and D.W. Murray. Stochastic motion clustering. In *European Conference on Computer Vision*, pages 328–337, May 1994.

[26] P.H.S. Torr, A. Zisserman, and S.J. Maybank. Robust detection of degenerate configurations for the fundamental matrix. In *International Conference on Computer Vision*, pages 1037–1042, Cambridge, MA, June 1995.

[27] J. Wang and E. Adelson. Layered representation for motion analysis. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 361–366, New York, June 1993.

Appendix

In this appendix, we prove theorem 1, i.e., we derive Equation (4).

Let $\vec{\mu_1}$ and $\vec{\mu_2}$ be the planar-parallax displacement vectors of two points that belong to the static background. From Equation (3), we know that

$$\vec{\mu_1} = \gamma_1 \frac{T_z}{d'_\pi}(\vec{\mathbf{e}} - \vec{\mathbf{p_{w_1}}}) \quad ; \quad \vec{\mu_2} = \gamma_2 \frac{T_z}{d'_\pi}(\vec{\mathbf{e}} - \vec{\mathbf{p_{w_2}}}). \tag{6}$$

Therefore,

$$\vec{\mu_1}\gamma_2 - \vec{\mu_2}\gamma_1 = \gamma_1\gamma_2 \frac{T_Z}{d'}(\vec{\mathbf{p_{w_2}}} - \vec{\mathbf{p_{w_1}}}) \tag{7}$$

This last step eliminated the epipole $\vec{\mathbf{e}}$. Eq. (7) entails that the vectors on both sides of the equation are parallel. Since $\gamma_1\gamma_2\frac{T_Z}{d'}$ is a scalar, we get: $(\vec{\mu_1}\gamma_2 - \vec{\mu_2}\gamma_1) \parallel \vec{\Delta\mathbf{p}}_\mathbf{w}$, where $\vec{\Delta\mathbf{p}}_\mathbf{w} = (\vec{\mathbf{p_{w_2}}} - \vec{\mathbf{p_{w_1}}})$. This leads to the *pairwise parallax constraint*

$$(\vec{\mu_1}\gamma_2 - \vec{\mu_2}\gamma_1)^T (\vec{\Delta\mathbf{p}}_\mathbf{w})_\perp = 0, \tag{8}$$

where $\vec{v}_\perp$ signifies a vector perpendicular to $\vec{v}$. When $T_Z = 0$, a constraint stronger than Eq. (8) can be derived: $(\vec{\mu_1}\frac{\gamma_2}{\gamma_1} - \vec{\mu_2}) = 0$, however, Eq. (8), still holds. This is important, as we do not have a-priori knowledge of T_Z to distinguish between the two cases.

From Eq. (8), we can easily derive: $\frac{\gamma_2}{\gamma_1} = \frac{\vec{\mu_2}^T(\vec{\Delta\mathbf{p}}_\mathbf{w})_\perp}{\vec{\mu_1}^T(\vec{\Delta\mathbf{p}}_\mathbf{w})_\perp}$, which is the same as Equation (4) of Theorem 1.