CAR-TR-935
CS-TR-4096

N00014-95-1-0521
January 2000

On the Synthesis of Dynamic Scenes from Reference Views

Y. Wexler
Center for Automation Research
University of Maryland
College Park, MD 20742-3275
wexler@cs.umd.edu

A. Shashua
School of Computer Science and Engineering
The Hebrew University of Jerusalem
91904 Jerusalem, ISRAEL
shashua@cs.huji.ac.il

Abstract

We consider a scene, containing many objects moving with constant velocity along straight line paths, seen from three reference viewpoints at three different times. The scene may even consist **only** of moving objects with no static features. We wish to create a new image sequence showing the scene from any arbitrary viewing position. We make use of a newly discovered tool, the "dual Htensor"[1], that connects together three views of a coplanar configuration of (unlabeled) static and moving points. Using the dual Htensor we show how to factor out the viewing transformation and create a new sequence in which the static features are stable and the moving features are synthesized as if moving with constant velocity. The constant velocity is achieved by introducing a 1D collineation that transforms the positions of the objects in the three views into their positions at new times.

The support of this research by the Office of Naval Research under contract N00014-95-1-0521, is gratefully acknowledged, as is the help of Sara Larson in preparing this paper.

1 Introduction

Consider the following dynamic image stabilization problem. We are given a coplanar configuration of static and moving objects (along straight line paths) seen in three different views, each taken at a different time. Alternatively, the scene is 3D but the cameras' optical centers are aligned, or the cameras are affine — in other words, the image to image transformations are 3×3 homography matrices. The number of moving objects can be very large — at the extreme, the entire scene may consist of moving objects — and we are not given prior information about which object is static and which objects are moving. The only information is the three views, and we assume we can find dense correspondences between them by means of image correlation, for example.

We wish to stabilize the sequence, i.e., factor out the viewing transformation, and create a new sequence in which the static features are stable whereas the moving features are synthesized at new time steps as if they had been moving at constant velocities. The synthesis can take the form of interpolation or extrapolation.

This type of dynamic synthesis problem is useful for creating representations, such as an image mosaic [11, 12], that contain a temporal dimension in addition to the spatial dimensions. Also of interest are graphics applications like view-morphing or the more recent dynamic view morphing [6]. Other applications include collision analysis between a moving vision platform and approaching vehicles, and image sequence compression. Our approach is most useful to these applications in situations where the scene is mostly planar, or the camera is undergoing mostly rotational motion, and the scene is rich in dynamic information.

The major challenges in this task are twofold. First, the process of factoring out the viewing transformation boils down to recovering the image-to-image transformations, i.e., the pairwise homography matrices. However, the features are not necessarily static in space, and moreover, *there may be no static features at all*, i.e., all the matching triplets arise from moving features in space. In other words, we must use a technique that can treat the measurements (matching triplets) arising from static and moving points alike. To this end, we adopt the "dual Htensor" recently introduced in [1], described in Section 2, and modify it to our needs. During the synthesis process the objects on the world plane should be moving with constant velocity, but because of perspective effects the constant velocity is not necessarily conserved in the image coordinate system. We show that constant velocity in the world plane can be generated by constructing a 1D projective transformation between the time function and the the warping transformation. In this manner, physically correct interpolation and extrapolation can both be achieved during the synthesis process. It is then also possible to use the transformation for collision analysis or for any other application that requires us to predict the positions of moving features, assuming constant-velocity motion, at any time step.

A relatively large body of research exists on the synthesis of physically correct new views from a small number of reference views (cf. [2, 8, 9, 5]), but none of these apply directly to dynamic scenes. Recently, the problem of reconstruction (using known camera-to-camera transformations) of moving points along straight line and conic paths was introduced in [3, 7]. However, this method assumes that five or more views are available (nine, for conics). More closely related to our work is the dynamic view interpolation between pairs of views introduced in [6]. As in our case, the camera-to-camera transformation is modeled by a homography matrix, but it is assumed to be known (or recoverable by matching static points). Each moving object must somehow be segmented out (by forming layers, one per object) and a number of matching points must be identified on each object for the purpose of recovering its relative motion (object fundamental matrix) and computing a pre-warping transformation for the object.

In our approach, it is not required to segment the scene into static and moving points, and there is no requirement for separate pre-warping of each moving object in order to create a constant-velocity synthesis. Furthermore, the synthesis process is physically correct at all time steps, not only for those between (interpolation) the original reference views but also for time steps beyond them (extrapolation).

We will start with a brief description of necessary background information about homography matrices, tensor notations and dual Htensors, and will continue with our main subject in Section 3.

2 Background: Homographies and Dual Htensors

The camera-to-camera (or image-to-image) transformation takes the form of a 3×3 matrix (called a homography matrix) under the following situations: (i) the scene is planar and is viewed from a general collection of camera positions, (ii) the scene is general, but is viewed by a collection of cameras whose camera projection centers are coincident (i.e., a purely rotating camera), and (iii) the scene is general, the projection model is affine (orthographic), and the image coordinate systems are centered (3D translation drops out). For convenience, we will work from now on with interpretation (i), i.e., a planar scene and general camera positions. However, our results apply to any situation in which the camera-to-camera transformation can be modeled by a homography matrix (after some pre-processing like centering the image data).

Therefore, we will be working with the projective plane, i.e., the space $\mathcal{P}^2$. A point in $\mathcal{P}^2$ is defined by three numbers, not all zero, that form a coordinate vector defined up to a scale factor. The dual projective plane represents the space of lines which are also defined by triplets of numbers. A point p in the projective plane coincides with a line s if and only if $p^\top s = 0$, i.e., the scalar product vanishes. In other words, the set of lines coincident with the point p are represented by the coordinate vectors s that satisfy $p^\top s = 0$, and vice versa: a point represented by the coordinate vector p can be thought of as the set of lines through it (a.k.a the pencil of lines through p). A line s going through two points p_1, p_2 is represented by the cross product $s \cong p_1 \times p_2$ where $\cong$ denotes equality up to scale. Likewise, the point of intersection p of the lines s_1, s_2 is represented by $p \cong s_1 \times s_2$.

In the projective plane any four points in general position can be uniquely mapped into any other four points. Such a mapping is called a *collineation* and is defined, up to scale, by a 3×3 invertible matrix. Such matrices are sometimes referred to as *homographies*. A collineation is defined by four pairs of matching points; each pair provides two linear constraints on the entries of the homography matrix. If H is a homography matrix defined by four matching pairs of points, then H^{-T} (inverse transpose) is the dual homography that maps lines onto lines.

The projective plane is useful for modeling the image plane. Consider a collection of points $P_1, ..., P_n$ in space lying on a plane π viewed from two viewpoints. The projections of P_i are p_i, p'_i in views 1,2 respectively. Because the collineations form a group, there exists a unique homography matrix H_π that satisfies the relation $H_\pi p_i \cong p'_i$, $i = 1, ..., n$, and where H_π is uniquely determined by four matching pairs from the set of n matching pairs. Moreover, $H_\pi^{-T} s \cong s'$ will map between matching lines s, s' arising from 3D lines lying in the plane π. Likewise, $H_\pi^\top s' \cong s$ will map between matching lines from view 2 to view 1.

Consider three views of a planar surface with the homography matrices A, B from views 2 to 1 and from 3 to 1 respectively. Let a point P be moving on the planar surface along some straight line path simultaneously with the motion of the camera. Let the projection of P at time t_1 onto view 1 be p, the projection of P at time t_2 onto view 2 be p', and the projection of P at time t_3 onto view 3 be p'' — see Fig. 1). Because P traces a straight line path we must

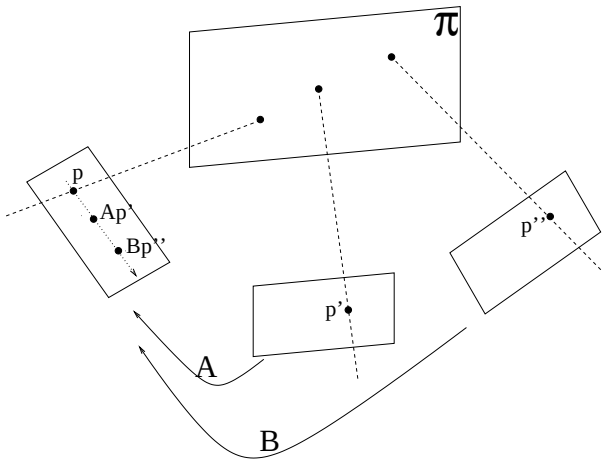


Figure 1: The dual homography tensor and moving points. The collineations A, B are from view 2 to 1 and 3 to 1 respectively. If the triplet p, p', p'' are projections of a moving point along a line on π then p, Ap', Bp'' are collinear in view 1. Thus, $p^\top(Ap' \times Bp'') = 0$, or $p^i p'^j p''^k H_{ijk} = 0$ where $H_{ijk} = \epsilon_{inu} a^n_j b^u_k$.

have

$$p^\top(Ap' \times Bp'') = det(p, Ap', Bp'') = 0$$

whether the point P did not move, i.e., the optical rays through p, p', p'' intersect at a point, or P did move (the optical rays intersect on a line). Therefore, the triplet of matching points p, p', p'' contributes a *measurement* regardless of whether P is static or dynamic. The measurement is towards the following object:

$$H_{ijk} = \epsilon_{inu} a^n_j b^u_k. \tag{1}$$

where ϵ_{inu} is the cross product tensor, and the measurement itself is simply $p^i p'^j p''^k H_{ijk} = 0$. To understand what this means we must make a detour into tensor notation.

When working with tensor objects it matters whether the coordinate vectors stand for points or lines. A point is an object whose coordinates are specified with superscripts, i.e., $p^i = (p^1, p^2, p^3)$. These are called contravariant vectors. A line in $\mathcal{P}^2$ is called a covariant vector and is represented by subscripts, i.e., $s_j = (s_1, s_2, s_3)$. Indices repeated in covariant and contravariant forms are summed over, i.e., $p^i s_i = p^1 s_1 + p^2 s_2 + p^3 s_3$. This is known as a contraction. For example, if p is a point incident to (i.e., lying on) a line s in $\mathcal{P}^2$, then $p^i s_i = 0$.

Vectors are also called tensors of valence 1. 2-valent tensors (matrices) have two indices and the transformation they represent depends on the covariant-contravariant positioning of the indices. For example, a^j_i is a mapping from points to points (a collineation, for example), and hyperplanes (lines in $\mathcal{P}^2$) to hyperplanes, because $a^j_i p^i = q^j$ and $a^j_i s_j = r_i$ (in matrix form: $Ap = q$ and $A^\top s = r$); a_{ij} maps points to hyperplanes; and a^{ij} maps hyperplanes to points. When viewed as a matrix the row and column positions are determined accordingly: in a^j_i and a_{ji} the index i runs over the columns and j runs over the rows; thus $b^k_j a^j_i = c^k_i$ is $BA = C$ in matrix form. An outer product of two 1-valent tensors (vectors), $a_i b^j$, is a 2-valent tensor c^j_i whose i, j entries are $a_i b^j$; note that in matrix form $C = ba^\top$. A 3-valent tensor has three indices, say H^{jk}_i. The positioning of the indices reveals the geometric nature of the mapping: for example, $p^i s_j H^{jk}_i$ must be a point because the i,j indices drop out in the contraction process and we are left with a contravariant vector (the index k is a superscript). Thus H^{jk}_i maps a

point in the first coordinate frame and a line in the second coordinate frame into a point in the third coordinate frame. A single contraction, say $p^i H_i^{jk}$, of a 3-valent tensor leaves us with a matrix. Note that when p is $(1,0,0)$ or $(0,1,0)$, or $(0,0,1)$ the result is a "slice" of the tensor.

The cross product (vector product) operation $c = a \times \mathrm{b}$ is defined for vectors in $\mathcal{P}^2$. The product operation can also be represented as the product $c = [a]_\times b$ where $[a]_x$ is called the "skew-symmetric matrix of a" and has the form

$$[a]_\times = \begin{pmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{pmatrix}$$

In tensor form we have $\epsilon_{ijk} a^i b^j = c_k$, representing the cross product of two points (contravariant vectors) resulting in the line (covariant vector) c_k. Similarly, $\epsilon^{ijk} a_i b_j = c^k$ represents the point intersection of the two lines a_i and b_j. The tensor ϵ is defined such that $\epsilon_{ijk} a^i$ produces the matrix $[a]_\times$ (i.e., ϵ contains $0, -1, 1$ in its entries such that its operation on a single vector produces the skew-symmetric matrix of that vector).

The tensor H_{ijk} defined above was introduced in [1] and referred to as a "dual homography tensor" or dual Htensor in short. We see that each matching triplet p, p', p'' contributes one linear equation

$$p^i p'^j p''^k H_{ijk} = 0$$

to the 27 entries of the dual Htensor, regardless of whether the matches arose from a static or moving point (along a straight-line path). Furthermore, in [1] it was shown that if among the measurements, x triplets are known to arise from static points, then the minimal number of moving points in the total set of measurements should be at least $16 - 4x$. In other words, in an completely unsegmented situation, i.e., it is not known whether a matching triplet has arisen from a static or moving point, one needs at least 26 matching triplets, out of which 16 must arise from moving points. At the other extreme, if four matching triplets are known to arise from static points, then these four matching triplets are all one needs to solve for H_{ijk}. Also, in [1] it was shown how to extract the constituent homography matrices A, B from the dual Htensor, but we will not need this here and instead show later how one can make direct use of H_{ijk} in image-to-image mappings.

3 Synthesis of Dynamic Scenes

Suppose we are given a dense set of matching triplets p, p', p'' in views 1,2,3 respectively. The reason we require a dense matching is simply for convenience of implementing the basic synthesis engine using point-wise image forward warping. There are other methods of image warping, for example using a series of sparse line segment correspondences of [10], but these will not be considered here. The matching triplets may arise from static or from moving points. In the completely unlabeled configuration, i.e., when there is no prior information as to what measurement is static and what is dynamic, we will need at least 26 measurements of matching triplets (out of which at least 16 arise from dynamic points) for a linear solution for the dual Htensor from $p^i p'^j p''^k H_{ijk} = 0$. If some of the measurements are labeled as arising from static points, fewer matching triplets are necessary (see the previous section).

Once H_{ijk} is recovered from the image measurements we can factor out the viewing transformation between pairs of images using the image-to-image mapping induced by the planar surface. Suppose we would like to map view 1 onto view 3, i.e., for every point p in view 1 we back-project the intersection of the optical ray through p and the planar surface onto p'' in view 3. This is done as follows (see Fig. 2):

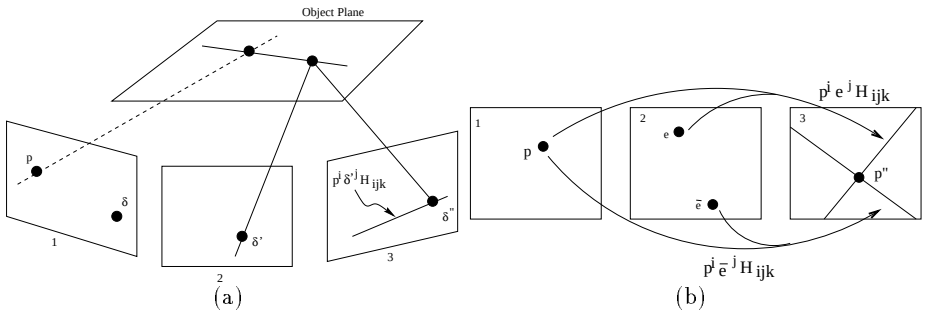


Figure 2: The dual Htensor can form a direct image-to-image mapping (a collineation) between pairs of views. (a) Consider a point δ' in view 2 and its matching points δ, δ'' in views 1,3. The matrix $\delta'^j H_{ijk}$ maps between two pencils of lines, one through δ in view 1 and the other through δ'' in view 3. Thus, $p^i \delta'^j H_{ijk}$ is a line through the matching point p''. (b) We can therefore represent p'' as the intersection of two lines $(p^i e^j H_{ijk}) \times (p^i \bar{e}^j H_{ijk})$ where $e, \bar{e}$ are any two vectors, say the standard basis $e = (1,0,0)$ and $\bar{e} = (0,1,0)$.

The double contraction of $p^i p'^j H_{ijk}$ is the projection of the straight line path of the scene point onto view 3. If the pair p, p' arise from a static point then $p^i p'^j H_{ijk}$ vanishes. Let p' range over the two unit vectors $e = (1,0,0)$ and $\bar{e} = (0,1,0)$; then

$$p'' \cong (p^i e^j H_{ijk}) \times (p^i \bar{e}^j H_{ijk})$$

Hence every point in one view can be mapped directly onto its matching point in any of the remaining two views, for example

$$\begin{aligned} p' &\cong (p^i e^k H_{ijk}) \times (p^i \bar{e}^k H_{ijk}) \\ p &\cong (p'^j e^k H_{ijk}) \times (p'^j \bar{e}^k H_{ijk}) \end{aligned}$$

Therefore, once the dual Htensor is recovered from any configuration of static and dynamic points it can be used to stabilize the static portion of the scene by warping the views onto a canonical coordinate frame — which could be any one of the original three frames (the reference view). The warping process brings the static regions in the scene into alignment with the reference view, whereas the dynamic regions are shifted along the projection of the straight-line path onto the reference view. We will now describe (i) the warping process, (ii) constant velocity synthesis, and (iii) collision analysis of dynamic features.

3.1 Dynamic Warping

Suppose we wish to synthesize the scene from the viewing position of view 1 ($t = 1$) at the time steps $1 \leq t \leq 2$. Let p_t be the position of point p at time t. Let p'_h the back-projected matching point p' onto view 1, i.e.,

$$p'_h \cong (p'^j e^k H_{ijk}) \times (p'^j \bar{e}^k H_{ijk}).$$

Let $\sigma(p)$ denote the non-homogeneous coordinates of a vector p created by normalizing by the third coordinate: $\sigma(p) = (p^1/p^3, p^2/p^3)$ which will be denoted by (x, y) — the image coordinates. Let U_{12} denote the image displacement from view 2 to view 1, i.e., $U_{12} = \sigma(p) - \sigma(p')$. Then the flow (dx, dy) needed to "forward warp" view 2 onto the reference view at time t, i.e.,

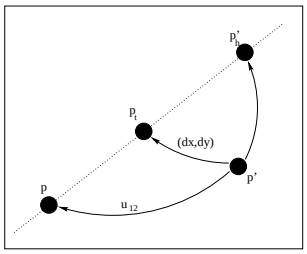


Figure 3: The point p' is forward-warped to position p_t in the reference frame (view 1). The back-projection of p' onto view 1 is p'_h which coincides with p if p, p' arise from a static point in space.

$\sigma(p_t) = \sigma(p') + (dx, dy)$, is

$$\begin{pmatrix} dx \\ dy \end{pmatrix} = U_{12} + (t-1)(\sigma(p'_h) - \sigma(p)) \tag{2}$$

Note that if p, p' arise from a static point, then $p'_h \cong p$, i.e., $\sigma(p'_h) = \sigma(p)$; therefore we have $U_{12} + (t-1)0 = U_{12}$ for all values of t, so that p_t will remain fixed at p regardless of the value of t. A dynamic point, on the other hand, will move along the line connecting $\sigma(p)$ and $\sigma(p'_h)$. See Fig. 3 for an illustration.

Next, we wish to determine the time step t that conforms to *constant-velocity* motion in world coordinates, i.e., in the scene coordinate system. This will enable us to synthesize both interpolated and extrapolated motion of the dynamic regions.

We want to produce physically valid images, so we need to preserve the correct speeds of the objects. Note that an object moving at constant speed in the world can produce a point moving at varying speed on the image. Since the projection process is a projective transformation, we know that the cross ratio must be preserved. When converting T into t, we want to preserve this cross ratio.

We will use the fact that we have three images at our disposal and create a 1D collineation between the progression of time T in world coordinates and the term t in equation (2). A constant velocity in world coordinates is a succession $1, 2, 3, T$ where $1, 2, 3$ correspond to the times of taking the three original images, respectively. We wish to find for every choice of T the corresponding value of t. Let p''_h be the back-projected matching point p'' onto view 1, i.e.

$$p''_h \cong (p''^k e^j H_{ijk}) \times (p''^k \bar{e}^j H_{ijk}).$$

Then at time $T = 3$ we should have

$$t = \frac{\|\sigma(p''_h) - \sigma(p)\|}{\|\sigma(p'_h) - \sigma(p)\|} + 1$$

Hence we have a 1D collineation $\mathcal{A}$ that maps the basis $(1,1), (2,1), (3,1)$ onto the basis $(1,1), (2,1), (\|\sigma(p''_h) - \sigma(p)\| + \|\sigma(p'_h) - \sigma(p)\|, \|\sigma(p'_h) - \sigma(p)\|)$ For every chosen value of T we have

$$\begin{pmatrix} t \\ 1 \end{pmatrix} \cong \mathcal{A} \begin{pmatrix} T \\ 1 \end{pmatrix}.$$

Because three points uniquely determine a 1D collineation, we can determine in this way the synthesized position p_t for any time T, not necessarily between 1 and 3.

To handle problems arising from occlusion we note that when $1 \leq t \leq 2$, p_t can be generated by forward-warping p' as described above or by forward-warping p — the result of which should be the same except in areas under occlusion. We characterize the regions that have less information (due to occlusion) by measuring a "dilation" factor from the dense correspondence field. A region which is revealed during the camera motion or object motion will stretch or shrink depending on the direction of flow measurement (view 1 to 2 or 2 to 1). It is always preferable to take a region from the direction that results in a shrinkage rather than an expansion. Therefore, we consider the flow in both directions U_{12} and U_{21}, and for each pixel we choose the source image from which to perform forward-warping that results in the smallest expansion, as measured by the perimeter of the triangle defined by forward-warping the neighbors in directions (N,SW,SE) using either U_{12} or U_{21}.

4 Experiments

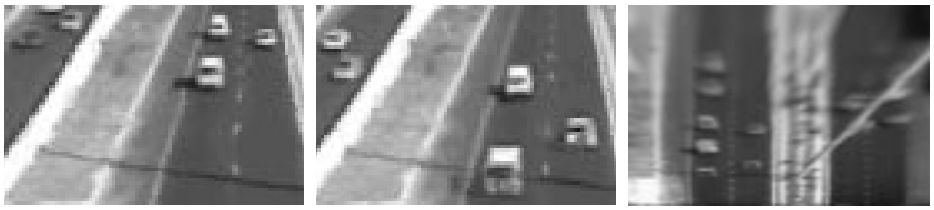
We have conducted a number of experiments in synthesizing new images at extrapolated time steps, i.e., $T > 3$ and $T < 1$, and also from novel viewing positions. We have also used the concepts above to determine time to collision in moving vehicle situations. Regarding synthesis of novel images, consider Fig. 4, depicting images of a road sequence. The motion of the camera was mostly rotational, as can be seen from the overlay of the first and last images in Fig. 4c. The dual Htensor was recovered from a dense flow field that was computed in a coarse-to-fine framework [4]. Fig. 4d shows a synthesized image extrapolated backwards in time to $T = -4$. The overlay of a portion of this image onto the reference image is shown in Fig. 4g. Likewise, Figs. 4e,h show an image extrapolated forward in time to $T = 6$ and its overlay onto the reference image. Finally, Figs. 4f,i show a synthesized top view of the two time-extrapolated scenes of Figs. 4d,e.

Fig. 5 depicts another situation involving three images of a mostly planar scene with moving toy vehicles. Fig. 5d shows an overlay of two out of the three views, and in Fig. 5e,f an time-extrapolated image is synthesized overlayed on the reference image. Note that the static features are aligned and the dynamic features are displaced as a function of time.

In the last experiment, shown in Fig. 7, we made use of the ability to correctly synthesize the position of dynamic features across time to predict time to collision between a forward-moving vision platform (on a vehicle) and neighboring vehicles. This is done as follows. We are given three images of the scene when the camera is moving forward (along a straight line), and the dual Htensor computed from the roadway (robust estimation picks out the roadway as the dominant planar region in the scene). We pick a fixed location $p = p' = p'' = (x, y, 1)$ in the three views corresponding to the center of the roadway. Because the vehicle is moving along a straight line we are guaranteed that the corresponding object points trace a straight-line path (see Fig. 6). Therefore, the dual Htensor will back-project p', p'' onto view 1 and create three collinear points p, p'_h, p''_h. Suppose, we have a moving vehicle tracked along the three views; then its matching points q, q', q'' also create a triplet of collinear points q, q'_h, q''_h in view 1. The intersection of the two lines will predict the point of collision in view 1. The collision will occur if there exist a time t such that $p_t \cong q_t$ using the 1D collineation framework defined in the previous section. Alternatively, a collision will occur if the cross ratio of the quadruples p, p'_h, p''_h, S and q, q'_h, q''_h, S are equal to one another, where S is the collision point in image 1. Fig. 7 illustrates this idea by marking the collision point in (d) and the back-projected points in (e).

(a) First Image. (b) Last Image. (c) First and Last.

(d) Extrapolation, time=-4. (e) Extrapolation, time=+6. (f) Top view of (d)



(g) Zoom of (d) (h) Zoom of (e) (i) Top view of (d) and (e) combined

Figure 4: Time-extrapolation experiments with a road sequence. (a,b) are the two extreme views, and (c) is their overlay, demonstrating that the camera has undergone mostly rotational motion. (d) Synthesized image extrapolated backwards in time to $T = -4$. (e) Image extrapolated forward in time to $T = 6$. (g,h) Overlay of (d,e) with reference image. (f,i) Top view of images (d,e).

(a) First Image. (b) Second Image. (c) Third Image.

(d) Edges of the second view overlayed on the third view. (e) Edges of the first view overlayed on the extrapolated image. (f) Enlarged section of 5(e).

Figure 5: Time-extrapolation experiment. (a,b,c) are the original images, (d) displays the overlay of two of the original images, and (e,f) are extrapolated views overlaid on top of the reference image; note that the static features are aligned.

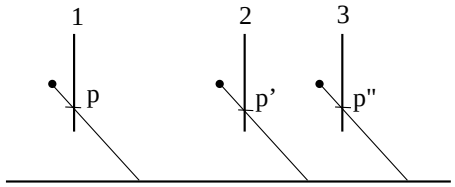
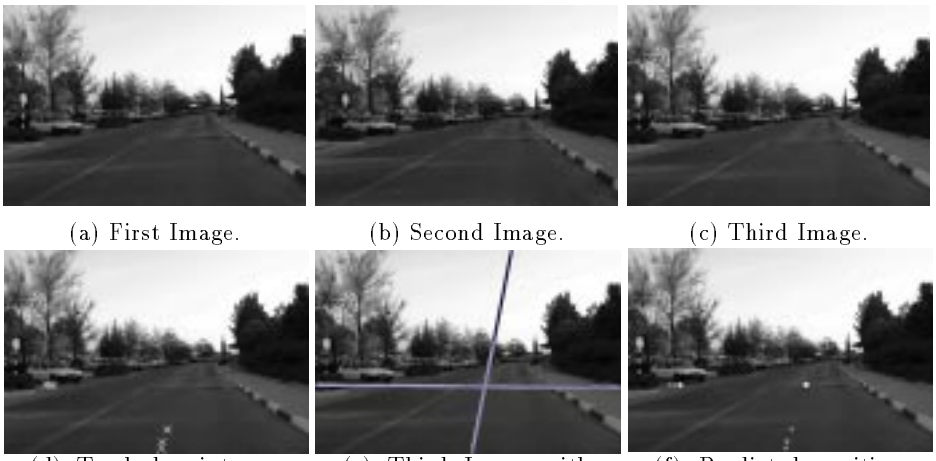


Figure 6: To find the point and time of collision between a moving platform (along a straight line) and an approaching vehicle (also along a straight line) one can back-project a fixed image location $p = p' = p''$. The result is p, p'_h, p''_h which are collinear in view 1.



(a) First Image. (b) Second Image. (c) Third Image.

(d) Tracked points on third image (e) Third Image with predicted motion lines. (f) Predicted position of camera and car.

Figure 7: Collision analysis. (a,b,c) Three original images of a visual platform moving along a straight-line path and a vehicle on the left-hand side moving toward a collision course. (d) The points back-projected onto view 3 (see text). (e) The point of collision. (f) The predicted position of the vehicles at the point of collision; note that the vehicle approaching from the side will not collide with the moving platform.

5 Summary

We have presented a technique for synthesizing new images of dynamic scenes containing many objects moving along straight-line paths. The method can handle any mixture of static and dynamic features, including the extreme case in which all the measurements arise from dynamic features. Our method is based on two principles: (i) the use of the dual Htensor as an image-to-image mapping, and (ii) introducing a 1D collineation between the time steps in the world coordinates and the time steps in the reference image coordinates. The latter allows us to create interpolations and extrapolations in time and to predict the positions of dynamic features, assuming constant-velocity motion, at any time step. We have implemented these idea for two applications: (i) dynamic morphing, (ii) collision analysis. Other applications which were not addressed here, that may benefit from these results, include dynamic image mosaicking and image sequence compression.

[1] A. Shashua and L. Wolf Homography tensors: On algebraic entities that represent three views of static or moving planar points.

[2] S. Avidan and A. Shashua. Novel view synthesis by cascading trilinear tensors. *IEEE Transactions on Visualization and Computer Graphics*, 4(3), 1998. A short version can be found in CVPR'97.

[3] S. Avidan and A. Shashua. Trajectory triangulation of lines: Reconstruction of a 3D point moving along a line from a monocular image sequence. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 1999.

[4] J.R. Bergen, P. Anandan, K.J. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *Proceedings of the European Conference on Computer Vision*, June 1992.

[5] O.D. Faugeras and L. Robert. What can two images tell us about a third one? In *Proceedings of the European Conference on Computer Vision*, pages 485–492, May 1994.

[6] R.A. Manning and C.R. Dyer. Interpolating view and scene motion by dynamic view morphing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 388–394, June 1999.

[7] A. Shashua, S. Avidan, and M. Werman. Trajectory triangulation over conic sections. In *Proceedings of the International Conference on Computer Vision*, pages 330–336, September 1999.

[8] L. McMillan and G. Bishop. Plenoptic modeling. In *Proceedings of SIGGRAPH 95*, pages 39–46, 1995.

[9] S.M. Seitz and C.R. Dyer. View morphing. In *Proceedings of SIGGRAPH 96*, pages 21–30, 1996.

[10] T. Beier and S. Neely. Feature-based image metamorphosis. In *Proceedings of SIGGRAPH 92*, pages 35–42, 1992.

[11] S. Peleg and J. Herman. Panoramic mosaics by manifold projection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 338–343, 1997.

[12] R. Szeliski. Video mosaics for virtual environments. *IEEE Computer Graphics and Applications*, pages 22–30, March 1996.