Novel View Synthesis in Tensor Space

Shai Avidan Amnon Shashua
Institute of Computer Science,
The Hebrew University,
Jerusalem 91904, Israel
e-mail: avidan@cs.huji.ac.il

Categories: Image-based Rendering, Applications, Shape Representation and Recovery.

Novel View Synthesis in Tensor Space

Categories: Image-based Rendering, Applications, Shape Representation and Recovery

Abstract

This paper presents a new method for synthesizing novel views of a 3D scene from two reference images and a tensor. No use of a 3D model of the scene is made, nor any use of camera geometry (relative locations of the cameras) of the two views is required — only the computation of the tensor. The novel views are modeled by specifying rotation and translation of the virtual camera starting from one of the reference views — which is an ideal "driving" mode for generating a synthetic movie. The synthesis process is governed by a simple equation (relating the model tensor and the new tensor) in which the user-specified virtual camera is plugged in, followed by a warping step — hence, the only time consuming component in this process is image warping. The virtual camera can be anywhere in 3D and the quality of the synthesized images are limited only by the quality of the (dense) correspondence between the two reference images and the amount of self-occlusions in the scene.

1 Introduction

This paper addresses the problem of synthesizing a novel image, from an arbitrary viewing position, from two or three model images of the scene. The minimal number of model images we will use is two, and in the process we are assuming that it is possible to find dense correspondence between two of the model images and a few (at least seven) matching points with the third model image, in case it is present. Under these conditions, we first recover the *trilinear tensor* of the model views, and then with the user specification of a virtual camera position (rotation and translation) relative to one of the model views, we generate a *new* tensor, and synthesize the new image from the two model views (in dense correspondence) and the new tensor. Because the process of view generation is governed ("derived") by virtual camera location, this approach lends itself to creating sequences (a "movie") of novel views as well.

The synthesis process proceeds as follows. A new equation relating the model tensor (of the model views) and virtual tensor (of two model views and the virtual camera) is derived. The user-specified location of the virtual camera (rotation and translation) is plugged into this equation to produce the tensor (virtual tensor) that governs the warping from two of the model views to the virtual view. Thus, at run time, the only computational (and time intensive) component is simply *image warping*.

1.1 Novelty Over Previous Work

The notion of image-based rendering systems is gaining momentum in both the computer graphics and computer vision communities. The general idea is to avoid the computational-intensive process of acquiring a 3D model followed by rendering, and instead to use a number of model images of the object (or scene) as a representation from which novel views can be synthesized *directly* by means of image warping.

The work in this area can be roughly divided into three classes: (i) image interpolation, (ii) Off-line (Mosaic-based) synthesis, and (iii) On-line synthesis. The first class, image interpolation, is designed to create "in-between" images among two or more model images. This includes image morphing [3], direct interpolation from image-flows ("multi-dimensional morphing") [5, 18], image interpolation using 3D models instead of image-flow [6], and "physically correct" image interpolation [20, 31]. All but the last two references do not guarantee to produce physically correct images, and all cannot extrapolate from the set of input images — that is, create novel viewing positions that are outside of the viewing cone of the model images.

In the second class, the off-line (mosaic-based) synthesis, the synthesis is not created at run time — instead many overlapping images of the scene are taken and then "stitched" together. The simplest stitching occurs when the camera motion includes only rotation — in which case the transformation between the views is parametric and does not include any 3D shape (the transformation being a 2D projective transformation, a homography). This was cleverly done by [10, 7, 16] in what is known as "QuickTime VR". The problem with this approach is that one cannot correctly simulate translational camera motion from the set of model images. A full 3D synthesis proceeds by recovering the (projective) 3D coordinates of the scene in order to correctly compensate for the image distortions when stitching neighboring images. This approach is less relevant to the theme of this paper where we are interested in image synthesis without first recovering a 3D model of the world, but nevertheless, preliminary work in this area can be found in [13] and under "orthophoto" maps (for calibrated cameras) in the photogrammetric literature.

The third class, on-line synthesis, is about deriving an on-line warping function from a set of model images to create novel views on-the-fly based on user specification of location of the virtual camera. This approach has the advantage of avoiding the computation of 3D model or explicit camera geometry and relies instead on algebraic constraints that all views of the same 3D scene must obey. The basic approach stem from the work on image "reprojection" (also known in photogrammetric literature as "image transfer") for which a small number of image correspondences in a novel view determine the coefficients of algebraic constraints (be it the epipolar geometry or the trilinear tensor) that govern the transfer functions from two model views (in full correspondence) onto the novel view — for all remaining points (cf.[1, 21, 22, 9, 25]). The missing link from reprojection to image-based rendering is the possibility to determine the algebraic constraints (the transfer functions), not on the basis of few image correspondences with the novel view (which are not available), but on the basis of user specification of virtual camera location, and (yet) without recovering 3D model of the scene or explicit recovery of camera geometry among the model views.

The closest, and most relevant, work on on-line synthesis to our work is that of Laveau & Faugeras [14]. In their work, the epipolar geometry (in the algebraic form of the fundamental matrix of two views) is used for the reprojection component (cf. [1, 9]). The reprojection onto a novel view is determined by the user specifying the following details: (i) the projection of the virtual camera's center of projection onto the model views (epipolar points), and (ii) three matching points across the model views which are used to determine the desired virtual image plane (of the virtual camera). Hence, the first detail specifies (indirectly) the translational component of the virtual camera location, and the second detail specifies where the virtual camera is looking at (rotational component). This way of "driving" the synthesis process is somewhat indirect and requires a certain degree of trial-and-error for obtaining the desired translational component of the virtual camera, and even more so for obtaining the rotational component. In other words, Laveau & Faugeras avoid the requirement for matching points in the novel image, but instead require a set of matching points between the model images which indirectly describe the location of the virtual camera.

Our work differs from [14] in two key aspects. First, we use the trilinear tensor (trilinearity constraint across three views [22]), instead of the fundamental matrix, as the reprojection component. Second, because of the use of the trilinear constraint, the synthesis process is derived by specification of translation and rotation of the virtual camera directly and not by means of matching points. The use of the trilinear constraint for reprojection has two practical advantages: (i) the reprojection process does not suffer from critical camera configurations — notably the cases when the camera centers are collinear, or when the novel view intersects the trifocal plane — thus suggesting a more stable synthesis process, and (ii) empirical studies demonstrate that the reprojection process is significantly more accurate when done using the trilinear constraints than by epipolar intersection (see [2] for comparative analysis, and [22]). But, first and foremost, the major advantage for introducing the trilinear constraints in the context of image-based rendering is the possibility for driving the synthesis process without requiring any matching points — and instead by directly specifying the translational and rotational components of the virtual camera position relative to one of the model images.

2 Necessary Preliminaries

The derivation of image-based rendering is based on two theoretical components which we will review briefly here: (i) the trilinear tensor, its recovery from image measurements across three views, and its use for reprojection [22], and (ii) the infinitesimal rotational component of camera motion between two model views can be recovered in closed form and directly from the tensor [19].

The trilinear tensor concatenates together the camera transformation matrices (camera locations) across three views, as follows. Let P be a point in 3D projective space projecting onto p, p', p'' in three views ψ, ψ', ψ'' respectively, represented by the two dimensional projective space. The relationship between the 3D and the 2D spaces is represented by the 3×4 matrices, $[I, 0]$, $[A, v']$ and $[B, v'']$, i.e.,

$$p = [I, 0]P$$
$$p' \cong [A, v']P$$
$$p'' \cong [B, v'']P$$

We may adopt the convention that $p = (x, y, 1)^\top$, $p' = (x', y', 1)^\top$ and $p'' = (x'', y'', 1)^\top$, and therefore $P = [x, y, 1, \rho]$. The coordinates $(x, y), (x'y'), (x'', y'')$ are matching points (with respect to some arbitrary image origin — say the geometric center of each image plane). It will be useful for later to note that the 3×3 matrices A and B are 2D projective transformations (collineation, homography matrix) from ψ to ψ' and ψ'', respectively, induced by *some* plane in space (the plane $\rho = 0$). The vectors v' and v'' are known as epipolar points (the projection of O, the center of projection of the first camera, onto views ψ' and ψ'', respectively).

The trilinear tensor is an array of 27 entries:

$$\alpha_i^{jk} = v'^k b_i^j - v''^j a_i^k. \qquad i, j, k = 1, 2, 3 \qquad (1)$$

where superscripts denote contravariant indices (representing points in the 2D plane, like v') and subscripts denote covariant indices (representing lines in the 2D plane, like the rows of A). Thus, a_i^k is the element of the k'th row and i'th column of A, and v'^k is the k'th element of v'. The tensor α_i^{jk} forms the set of coefficients of certain trilinear forms that vanish on any corresponding triplet p, p', p'' (i.e., functions of views that are invariant to object structure). These functions have the following form: let s_k^l be the matrix,

$$s = \begin{bmatrix} 1 & 0 & -x' \\ 0 & 1 & -y' \end{bmatrix}$$

and, similarly, let r_j^m be the matrix,

$$r = \left[\begin{array}{ccc} 1 & 0 & -x'' \\ 0 & 1 & -y'' \end{array} \right]$$

Then, the tensorial equations are:

$$s_k^l r_j^m p^i \alpha_i^{jk} = 0, \tag{2}$$

with the standard summation convention that an index that appears as a subscript and superscript is summed over (known as a contraction). Hence, we have four trilinear equations (note that $l, m = 1, 2$). In more explicit form, these functions (referred to as "trilinearities") are:

$$\begin{aligned} x''\boldsymbol{\alpha}_i^{13}p^i - x''x'\boldsymbol{\alpha}_i^{33}p^i + x'\boldsymbol{\alpha}_i^{31}p^i - \boldsymbol{\alpha}_i^{11}p^i &= 0, \\ y''\boldsymbol{\alpha}_i^{13}p^i - y''x'\boldsymbol{\alpha}_i^{33}p^i + x'\boldsymbol{\alpha}_i^{32}p^i - \boldsymbol{\alpha}_i^{12}p^i &= 0, \\ x''\boldsymbol{\alpha}_i^{23}p^i - x''y'\boldsymbol{\alpha}_i^{33}p^i + y'\boldsymbol{\alpha}_i^{31}p^i - \boldsymbol{\alpha}_i^{21}p^i &= 0, \\ y''\boldsymbol{\alpha}_i^{23}p^i - y''y'\boldsymbol{\alpha}_i^{33}p^i + y'\boldsymbol{\alpha}_i^{32}p^i - \boldsymbol{\alpha}_i^{22}p^i &= 0. \end{aligned}$$

Since every corresponding triplet p, p', p'' contributes four linearly independent equations, then seven corresponding points across the three views uniquely determine (up to scale) the tensor α_i^{jk}. More details and applications can be found in [22]. Other relevant literature can be found in [29, 8, 30, 12, 11, 28, 23, 24].

One can readily see that given two views in full correspondence and the tensor (recovered using 7 matching points with a third view), the entire third view can be synthesized by means of (forward) warping: simply, from each trilinearity one can extract either x'' or y'', thus for every matching pair p, p' we can obtain p'' and copy the appropriate brightness value (take the average from the two model images, for example). This process is referred to as "reprojection" in the literature. There are alternative ways of performing reprojection, but if we would like to do it without recovering first a 3D model of the scene, the trilinear tensor generally provides the best results (see [2]).

Before we demonstrate the reprojection process, we will briefly mention the process we use for robustly estimating the tensor from three images. The tensor coefficients can be recovered linearly from 7 point matches, and via linear least-squares when more matching points are available. Given three views, we use a combination of optic-flow and robust statistics to estimate the tensor, this is done as follows. We use a gradient-based coarse-to-fine optic-flow method whose details can be found in [4, 15]. This results in a pixel-wise correspondence map between model image 1 and model image 2 and 3. The rank of the local Hessian matrix of image brightness around every pixel indicates the quality of the match (whether the local region is a corner, edge or flat), and we select the pixels that pass a quality threshold criteria (any optic-flow method, not necessarily the one we use, will provide a confidence measure for each match). Those

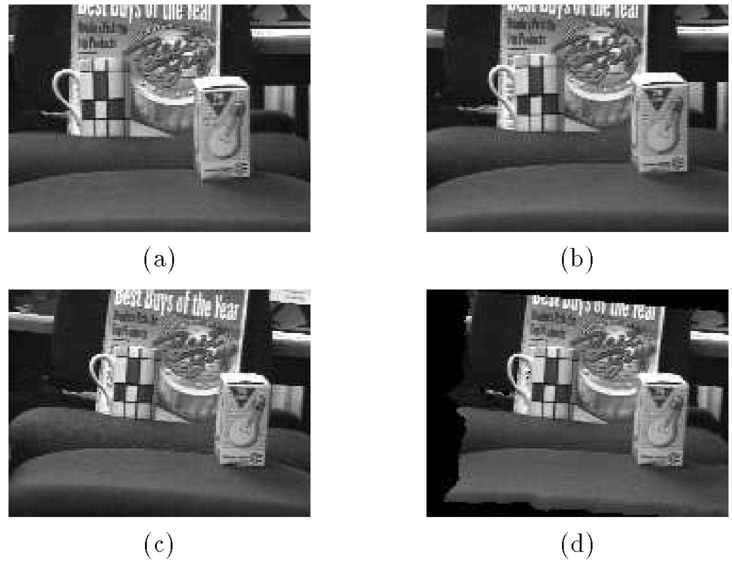

Figure 1: An example of image reprojection using optic-flow and a tensor. The tensor is recovered from the original three images ((a),(b) and (c)) and with the optic-flow between the first two images, we reproject the third image (d) using the trilinear equations.

points are referred to as "good points". On the set of good points we then estimate the tensor using a robust method in order to remove the effects of outliers. This is done by a repetitive lottery of a subset of 7 good points, recovering the tensor, and verifying the quality of the tensor on the remaining set of good points. More details on this process can be found in [17].

At this stage we have obtained a robust estimate of the tensor from three model images, and the estimation process is automatic. Figure 1 demonstrates the reprojection process on a set of three real images. The distances among the images is not large in order to facilitate the optic-flow process (and which would have hindered a 3D model reconstruction attempt because the camera base-line is small). The reprojected image is almost identical to the original image (to sub-pixel accuracy at most textured regions), indicating a robust reprojection mechanism.

We have described so far the implementation of the reprojection paradigm via the trilinear equations. In other words, given two model views and a tensor, the third view is uniquely determined and can be synthesized by means of a warping function applied to the two model images. In image-based rendering we would like to obtain the tensor (the warping function) via user specification of location of virtual camera, rather than by the specification of (at least) seven matching points. In order to do that we need to review three more components.

Another building block we will require is the relation between the tensor and the rotational component of camera motion. Assuming the cameras are internally calibrated it is possible to extract the rotational component of camera motion from at least two images. Typically, this involves recovering first the translational component of camera motion (the epipoles), followed then by recovering the rotation. However, the translation is the most error sensitive component in the process. If we further assume that the rotation is small and can be approximated by a small angle rotation matrix of the form

$$\begin{bmatrix} 1 & \Omega_Z & -\Omega_Y \\ -\Omega_Z & 1 & \Omega_X \\ \Omega_Y & -\Omega_X & 1 \end{bmatrix},$$

where the vector $\Omega = (\Omega_X, \Omega_Y, \Omega_Z)^\top$ is the rotation axis, and the magnitude of the vector is the magnitude of the rotation around this axis, then the rotation between views 1,2 can be recovered directly from the tensor as follows:

$$\begin{aligned} \Omega_X &= det \begin{pmatrix} \alpha_2^{j\,3} \\ \alpha_2^{j3} + \alpha_3^{j\,2} \\ \alpha_3^{j3} - \alpha_2^{j\,2} \end{pmatrix} / K \\ \Omega_Y &= det \begin{pmatrix} -\alpha_1^{j\,3} \\ \alpha_2^{j3} + \alpha_3^{j\,2} \\ \alpha_3^{j3} - \alpha_2^{j\,2} \end{pmatrix} / K \\ \Omega_Z &= det \begin{pmatrix} \alpha_1^{j\,2} \\ \alpha_2^{j3} + \alpha_3^{j\,2} \\ \alpha_3^{j3} - \alpha_2^{j\,2} \end{pmatrix} / K \\ K &= det \begin{pmatrix} \alpha_2^{j\,2} \\ \alpha_2^{j3} + \alpha_3^{j\,2} \\ \alpha_3^{j3} - \alpha_2^{j\,2} \end{pmatrix} \end{aligned} \qquad (3)$$

where $\alpha_2^{j\,2}$ stands for $(\alpha_2^{12},\ \alpha_2^{22},\ \alpha_2^{32})$, etc. This expression recovers directly and simply small rotations from the trilinear tensor. This formula has been derived in [19] for the purpose of stabilization of image sequences. We will use this formula as part of the off-line setup we do with the three model images, as described in the next section.

3 Image-based Rendering via the Trilinear tensor

The goal we would like to achieve is to recompute the tensor of three model views $< 1, 2, 3 >$ and produce a new tensor of views $< 1, 2, \psi >$ where the view ψ should correspond to the user specified new location of the camera. We derive a new equation relating the rotation and translation of the novel camera from its position of view 3, the tensor of views $< 1, 2, 3 >$ and the new tensor $< 1, 2, \psi >$. Thus the new tensor is obtained in closed-form and what is left is simply image-warping to reproject images 1,2 onto the desired image ψ.

Going back to the tensor of views $< 1, 2, 3 >$

$$\alpha_i^{j\,k} = v'^{\,k} b_i^j - v''^{\,j} a_i^k,$$

we note that the homographies A and B correspond to an arbitrary plane, and the choice of the plane does not change the tensor coefficients [22]. In particular, we may select the plane at infinity, i.e.,

$$A = M'' R' M'^{-1}$$
$$B = M''' R'' M'^{-1}$$

where M', M'', M''' are the internal parameters of the camera at view 1, 2 and 3, respectively (each is an upper-diagonal matrix with entries representing the location of the origin — principle point — focal length of the camera, and skew in case the camera is digital), and R', R'' are the rotational components of camera motion from view 1 to view 2 and 3, respectively.

We make now the assumption that the camera is internally calibrated, i.e., that $M' = M'' = M''' = I$. We must make this assumption (or recover off-line the focal length and principle point of the camera) because otherwise the problem of synthesizing *rigid* motions is unconstrained if the internal parameters are unknown and allowed to change across the three model views. Hence, the tensor of the three model views is:

$$\alpha_i^{j\,k} = v'^{\,k} R''^{j}_{\ i} - v''^{\,j} R'^{k}_{\ i}.$$

Note that v', v'' are the translational components of camera motion — now that the camera is assumed internally calibrated.

Let R denote the desired rotation matrix of the camera relative to view 3, and t denote the desired translation of the camera from its position in view 3. Consider the new tensor, $H_i^{j\,k} = R_l^j \alpha_i^{lk}$. We can readily show (by substitution) that $H_i^{j\,k}$ is the tensor of views $< 1, 2, 3' >$, where $3'$ is the view seen by camera 3 after the image plane is rotated by R (i.e., camera 3 only rotates around its center of projection).

Proposition 1 *Let $\alpha_i^{j\,k}$ be the tensor of views $< 1, 2, 3 >$ and R be a rotation matrix. Then the tensor $H_i^{j\,k} = R_l^j \alpha_i^{lk}$ is of views $< 1, 2, 3' >$, where the view $3'$ results from rotating the third camera around its coordinate axes by R.*

Proof 1

$$H_i^{j\,k} = R_l^j \alpha_i^{lk} = v'^{k}(R_l^j R''^{l}_{\ i}) - (v''^{l} R_l^j) R'^{k}_{\ i}.$$

It is therefore clear that $H_i^{j\,k}$ is a tensor $< 1, 2, \psi >$ for some view ψ. Note that $R_l^j R''^{l}_{\ i}$ is the combined rotation RR'' and $v''^{l} R_l^j$ is Rv'' which, taken together, means that the view ψ is a result of rotating the third image plane by R. $\square$

Next, we wish to translate the center of projection of the third camera after the rotation R. Let v''' be the resulting new epipolar point in the synthesized view, and let G_i^{jk} be the desired tensor, then:

$$\begin{aligned} G_i^{jk} &= v'^k (R_l^j R''^l_i) - v'''^j R'^k_i \\ &= H_i^{jk} + (v''^l R_l^j - v'''^j) R'^k_i \\ &= R_l^j \alpha_i^{lk} + t^j R'^k_i . \end{aligned}$$

Note that the desired translation t is $Rv'' - v'''$, i.e., the line connecting the center of projection of the third camera and the center of projection of the virtual camera. We have thus proved the following theorem:

Theorem 1 *Given a tensor α_i^{jk} and the rotation R' between model images $1, 2$, then any other tensor G_i^{jk} of views $< 1, 2, \psi >$, where the view ψ corresponds the virtual camera position given by rotation R and translation t from the position of view 3, is given by:*

$$\boxed{G_i^{jk} = R_l^j \alpha_i^{lk} + t^j R'^k_i} \tag{4}$$

We are ready now to describe the algorithm for on-line synthesis of novel views. The algorithm has an off-line component done once on the three model views:

1. Given the three model views, $1, 2, 3$, find the tensor α_i^{jk} of those three views (in that order). This is done by first recovering optic-flow between images 1 to 2 and 1 to 3, detecting "good points", then a robust estimation of the tensor from the set of good matches along the three views.

2. Given the tensor α_i^{jk}, recover the rotation R' from image 1 to 2 using the formula in eqn. 3. The formula is exact if the rotation is small, otherwise repeat the following procedure: cancel the small-angle rotation by warping the two images, recover the tensor again, recover R' again; stop when R' is the identity matrix — the final rotation matrix is the product of the incremental rotations expressed as a Taylor series.

and an on-line component which is evoked repeatedly for every desired synthetic image:

1. Given the desired rotation R and translation t from the third camera position, recover the new tensor G_i^{jk} by

$$G_i^{jk} = R_l^j \alpha_i^{lk} + t^j R'^k_i .$$

2. Take model images 1,2 with the (dense) correspondence map between the two images and reproject onto the the desired image by using the tensor G_i^{jk} (reprojection using the trilinear equations).

3.1 Handling Visibility Issues

The image-based rendering algorithm described above does not handle visibility issues, i.e., the surface is assumed transparent. It is possible to use a simple "projective Z-buffering" procedure for enforcing the constraint that the surface is opaque (cf. [14, 13, 27]). If two pixels are reprojected onto the same cell, we simply choose the one close to the epipole in image 1 corresponding to the projection of the new center of projection onto image 1. The epipolar points can be recovered linearly from the tensor G_i^{jk} [28]. Note that we recover the epipolar points only for resolving visibility problems, not for reprojection, thus inaccuracy in the epipoles would not affect the accuracy of reprojection. It is also important to note that the "projective Z-buffering" is not always guaranteed to correctly resolve visibility problems, as a more rigorous treatment is needed for the general case, nevertheless, the procedure is fairly robust in practice.

4 Tensor of Two views

The paper, so far, considered the case of three model views to generate the trilinear tensor. Once the tensor is at hand we use equation 4 to generate new tensors and therefor new views. While two of the model images are used to compute the dense correspondence, thus capturing the 3D world, the third model view is used solely to compute the initial tensor and is not used afterwards. We can eliminate the use of the third image by introducing the the tensor of two, rather than three, views. In [26] we derive the tensor of two views and show it to be a legitimate tensor, possessing the properties of the general trilinear tensor of three views. The elements of this two-view tensor are those of the fundamental matrix between the two views. Explicitly, The two-view tensor can be constructed from the fundamental matrix elements as follows:

$$\begin{array}{lcll} \alpha_i^{jk} & = & f_{(6-k-j)i} & if \quad k = (j+1) mod 3 \\ \alpha_i^{jk} & = & -f_{(6-k-j)i} & if \quad j = (k+1) mod 3 \\ \alpha_i^{jk} & = & 0 & if \quad j = k \end{array} \tag{5}$$

where f_{ji} means the element in the j-th row and the i-th column of the fundamental matrix F. Recovering the tensor of two views is straight forward now. Compute the fundamental matrix and construct the tensor of two views using equation 5. Proceed as before to recover the rotation component between the two model views and to synthesize new tensors, using the tensor of two, rather then three, views as the initial tensor.

5 Experimental Results

The tensor-based rendering method was implemented on synthetic and real image images. We show here several examples on real images. In each example, a "movie" was created by specifying a set of key frames, each by a rotation and translation of the virtual camera from the second model frame. The parameters of rotation and translation were then linearly interpolated (not the images, only the user-specified parameters) to the desired number of frames. Fig. 3

shows the model images used to generate the movie as well as several frames from the synthetic movie. The movie can be seen in the attached web page http:www-seq-1.html . The two model images were captured with a single CANON VC-C1 camera by taking the pictures one after the other.

Notice that the synthetic sequence contains a considerable degree of extrapolation (i.e., views that are outside the viewing cone of the original two model views). The second example is shown in Fig. 4. The two model images were captured using a standard indy camera at a resolution of 640×480 pixels. The attached movie can be viewed in http:www-seq-2.html .

6 summary

We have derived a simple formula connecting a user-specified virtual camera motion R and t, the trilinear tensor of three model views α_i^{jk}, and the new tensor G_i^{jk} describing the warping function from two of the model views onto the virtual (desired) view:

$$G_i^{jk} = R_l^j \alpha_i^{lk} + t^j R'^k_i,$$

where R' is a function of α_i^{jk} (eqn. 3). Both α_i^{jk} and R' are computed only once as an off-line pre-process, thus the image synthesis is an on-line process in which R, t are plugged in the formula above to produce a new tensor, which in turn determines the warping process (reprojection) of the two model images to produce a synthetic image. During the entire process no 3D model of the scene is necessary, nor is it necessary to recover camera geometry or epipolar geometry of the model images.

The experiments have demonstrated the ability to synthesize new images from three closely-spaced model images, where the viewing range of the synthesized images far exceed the viewing cone of the model images (extrapolation of viewing position, rather than interpolation).

The concept of image-based rendering by direct warping is critical for further progress in real image animation in computer graphics, for the following reason. If a 3D model of the scene is required for the synthesis of new images, then the quality of the synthesized images can be very high, but at the expense of requiring very high computing power and expensive graphic equipment. A realistic model of a free-form scene can require millions of 3D points, and therefore a real-time "virtual tour" would require the most expensive graphics equipment that exist today. On the other hand, if the only computing needs required for synthesis is pixel-wise warping and the ability to store many images (in this paper we store only two or three model images, but future progress in this direction would undoubtly involve a warping of many images simultaneously), then storage is already cheap today, and pixel-wise warping can be implemented in hardware in a much lesser cost than dedicated 3D graphics hardware and high speed CPUs that high-end computer graphics workstations require today. Therefore, the challenge in this line of research is to devise direct

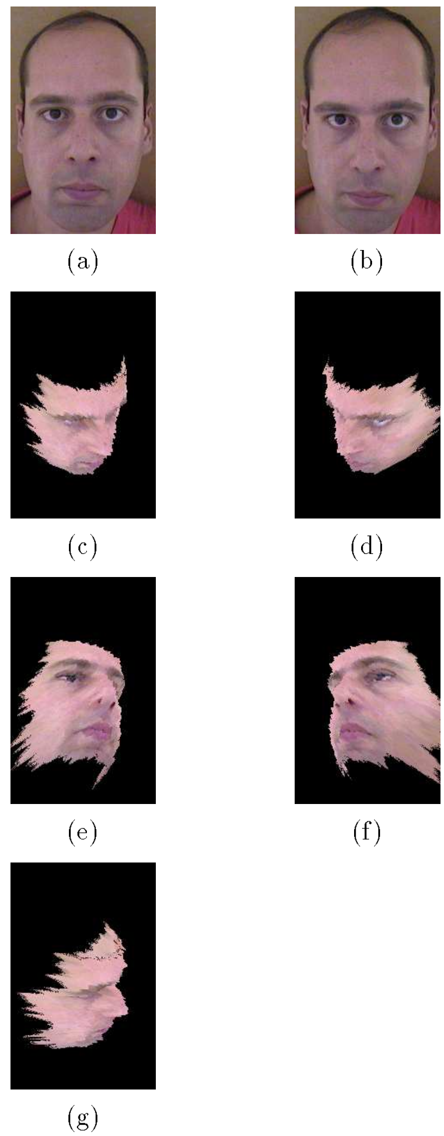

Figure 2: An example of image synthesis using optic-flow and a tensor. The original images are (a) and (b), the rest of the images are taken from the generated movie. They should be seen left to right, top to bottom.

warping-from-model-images with a quality comparable to synthesis based on 3D models.

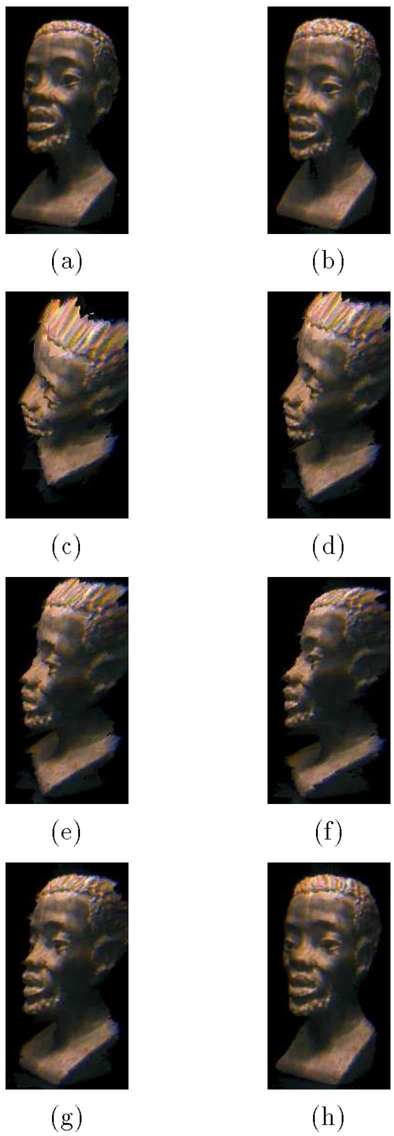

Figure 3: An example of image synthesis using optic-flow and a tensor. The original images are (a) and (b), the rest of the images are taken from the generated movie. They should be seen left to right, top to bottom.

References

[1] E.B. Barrett, M.H. Brill, N.N. Haag, and P.M. Payton. Invariant linear methods in photogrammetry and model-matching. In J.L. Mundy and A. Zisserman, editors, *Applications of invariances in computer vision*. MIT Press, 1992.

[2] E.B. Barrett, P.M. Payton, and G. Gheen. Robust algebraic invariant methods with applications in geometry and imaging. In *Proceedings of the SPIE on Remote Sensing*, San Diego, CA, July 1995.

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

[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*, Santa Margherita Ligure, Italy, June 1992.

[5] D. Beymer, A. Shashua, and T. Poggio. Example based image analysis and synthesis. A.I. Memo No. 1431, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, October 1993.

[6] S.E. Chen and L. Williams. View interpolation for image synthesis. In *SIGGRAPH*, pages 279–288, Anahiem, CA, August 1993.

[7] Shenchang Eric Chen. QuickTimeVR - an image-based approach to virtual environment navigation. In *SIGGRAPH*, pages 29–38, 1995.

[8] O.D. Faugeras and B. Mourrain. On the geometry and algebra of the point and line correspondences between N images. In *Proceedings of the International Conference on Computer Vision*, Cambridge, MA, June 1995.

[9] 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, Stockholm, Sweden, May 1994.

[10] N. Greene. Creating raster omnimax images from multiple perspective views using the elliptical weighted average filter. *IEEE Computer Graphics and Applications*, 6(6):21–27, 1986.

[11] R. Hartley. A linear method for reconstruction from lines and points. In *Proceedings of the International Conference on Computer Vision*, pages 882–887, Cambridge, MA, June 1995.

[12] A. Heyden. Reconstruction from image sequences by means of relative depths. In *Proceedings of the International Conference on Computer Vision*, pages 1058–1063, Cambridge, MA, June 1995.

[13] Rakesh Kumar, P. Anandan, Michal Irani, James Bergen, and Keith Hanna. Representation of scenes from collections of images. In *IEEE Workshop on Representation of Visual Scenes*, Boston, USA, June 1995.

[14] S. Laveau and O.D. Faugeras. 3-d scene representation as a collection of images. In *Proceedings of the International Conference on Pattern Recognition*, pages 689–691, Jerusalem, Israel, October 1994.

[15] B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings IJCAI*, pages 674–679, Vancouver, Canada, 1981.

[16] Leonard McMillan and Gary Bishop. Plenoptic modeling: An image-based rendering system. In *SIGGRAPH*, pages 39–46, 1995.

[17] Torr P.H.S., Zisserman A., and Murray D. Motion clustering using the trilinear constraint over three views. In *Workshop on Geometrical Modeling and Invariants for Computer Vision*. Xidian University Press., 1995.

[18] T. Poggio and R. Brunelli. A novel approach to graphics. A.I. Memo No. 1354, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, February 1992.

[19] B. Rousso, S. Avidan, A. Shashua, and S. Peleg. Robust recovery of camera rotation from three frames. In *Proceedings of the ARPA Image Understanding Workshop*, Palm Springs, CA, February 1996.

[20] Steven M. Seitz and Charles R. Dyer. Physically-valid view synthesis by image interpolation. In *IEEE Workshop on Representation of Visual Scenes*, Boston, USA, June 1995.

[21] A. Shashua. Projective structure from uncalibrated images: structure from motion and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(8):778–790, 1994.

[22] A. Shashua. Algebraic functions for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):779–789, 1995.

[23] A. Shashua and P. Anandan. The generalized trilinear constraints and the uncertainty tensor. In *Proceedings of the ARPA Image Understanding Workshop*, Palm Springs, CA, February 1996.

[24] A. Shashua and S. Avidan. The rank4 constraint in multiple view geometry. In *Proceedings of the European Conference on Computer Vision*, Cambridge, UK, April 1996. Also in CIS report #9520, November 1995, Technion.

[25] A. Shashua and N. Navab. Relative affine structure: Theory and application to 3D reconstruction from perspective views. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 483–489, Seattle, Washington, 1994.

[26] S. Avidan and A. Shashua Unifying Two-View and Three-View Geometry Subitted to CVPR 97.

[27] A. Shashua and S. Toelg. The quadric reference surface: Applications in registering views of complex 3d objects. In *Proceedings of the European Conference on Computer Vision*, Stockholm, Sweden, May 1994.

[28] A. Shashua and M. Werman. On the trilinear tensor of three perspective views and its underlying geometry. In *Proceedings of the International Conference on Computer Vision*, June 1995.

[29] M.E. Spetsakis and J. Aloimonos. Structure from motion using line correspondences. *International Journal of Computer Vision*, 4(3):171–183, 1990.

[30] B. Triggs. Matching constraints and the joint image. In *Proceedings of the International Conference on Computer Vision*, pages 338–343, Cambridge, MA, June 1995.

[31] T. Werner, R.D. Hersch, and V. Hlavac. Rendering real-world objects using view interpolation. In *Proceedings of the International Conference on Computer Vision*, pages 957–962, 1995.