M.I.T Media Laboratory Perceptual Computing Section Technical Report No. 338
Submitted for publication

Video Orbits of the Projective Group: A New Perspective on Image Mosaicing. [*]

S. Mann and R. W. Picard
MIT Media Lab; 20 Ames Street; Cambridge, MA 02139
steve@media.mit.edu, picard@media.mit.edu
(Tel) 617-253-0611 (Fax) 617-253-8874

Abstract

We present a new technique for estimating the projective (homographic) coordinate transformation between pairs of images, taken with a camera that is free to pan, tilt, rotate about its optical axis, and zoom. The technique solves the problem for two cases of static scenes: images taken from the same location of an arbitrary 3-D scene, or images taken from arbitrary locations of a flat scene. A new algorithm is presented for the parameter estimation and applied to the task of constructing high resolution still images from video. This approach generalizes inter-frame camera motion estimation methods which have previously used an *affine* model and/or which have relied upon finding points of correspondence between the image frames. The new projective algorithm which operates directly on the image pixels is shown to be superior in accuracy and ability to enhance resolution. The proposed method works well on image data collected from both good-quality and poor-quality video under a wide variety of conditions (sunny, cloudy, day, night). This new fully-automatic technique is also shown to be robust to deviations from the assumptions of static scene and no parallax.

1 Introduction

Many problems require finding the coordinate transformation between two images of the same scene or object. Whether to recover camera motion between video frames, to stabilize video images, to relate or recognize photographs taken from two different cameras, to compute depth within a 3-D scene, or to align images for mosaicing and high-resolution enhancement, it is important to have both a precise description of the coordinate transformation between a pair of images or video frames, and some indication as to its accuracy.

Traditional *block matching* (e.g. as used in *motion estimation*) is really a special case of a more general *coordinate transformation*. In this paper we demonstrate a new solution to the *motion estimation* problem using this more general estimation of a coordinate transformation, and propose a technique for automatically finding the 8-parameter projective coordinate transformation that relates two frames taken of the same static scene. We show, both by theory and example, how the new technique is more accurate and robust than previous methods which tend to rely on affine coordinate transformations, approximations to perspective, and/or the finding of point correspondences between the images. The new technique takes as input two frames, and automatically outputs the eight parameters of the exact model, to align the frames. It does not require the tracking or correspondence of explicit features, yet is computationally practical.

Although the theory we present makes the typical assumptions of static scene and no parallax, we show that the new estimation technique is robust to deviations from these assumptions. In particular, we apply the technique to image resolution enhancement and mosaicing, illustrating its success on a variety of practical and difficult cases, including some that violate the non-parallax and static scene assumptions.

An example of an *image mosaic* is shown in Fig 1, where the spatial extent of the image is increased by panning the camera while mosaicing (e.g. by making a *panorama*) and the spatial resolution is increased by zooming the camera and by combining overlapping frames from different viewpoints.

2 Background

Hundreds of papers have been published on the problems of motion estimation and frame alignment. (For review and comparison, see [1].) In this section we review the basic differences between coordinate transformations and emphasize the importance of using the "exact" 8-parameter projective coordinate transformation.

2.1 Coordinate transformations

A coordinate transformation maps the image coordinates, $\mathbf{x} = [x, y]^T$ to a new set of coordinates, $\mathbf{x}' = [x', y']^T$. Generally, the approach to "finding the coordinate transformation" relies on assuming it will take one of the forms in Table 1, and then estimating the two to twelve parameters in the chosen form. An illustration showing the effects possible with each of these forms is in Fig. 2.

[*]This work sponsored in part by Hewlett-Packard Research Labs and by BT, PLC.

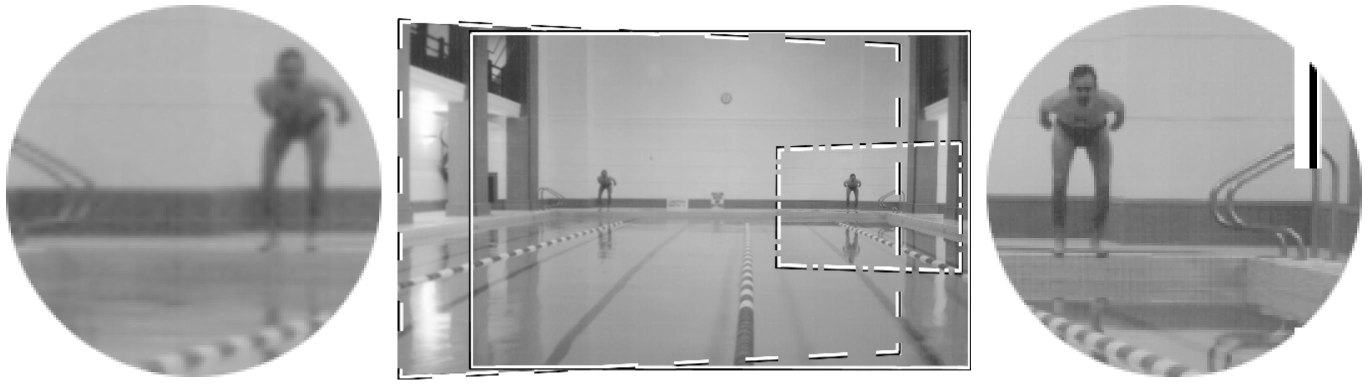

Figure 1: *Image mosaic* made from three images, one taken looking straight ahead (outlined in a solid line), one taken panning to the left (outlined in a dashed line), and the third taken panning to the right with substantial zoom-in (outlined in a dot-dash line). The second two have undergone a coordinate transformation to put them into the same coordinates as the one outlined in a solid line (which we call the *reference frame*). This image mosaic, made from NTSC-resolution images, occupies about 2000 pixels across, and, in places shows good detail down to the pixel level. Note increased sharpness in regions visited by the zooming-in, compared to other areas. (See magnified portions of the picture at sides.) This figure only shows the result of combining three images, but in the final production, many more images were used, resulting in a high resolution full-color image showing most of the room.

The most common assumption (especially in motion estimation for coding, and optical flow for computer vision) is that the coordinate transformation between frames is translation. Tekalp, Ozkan, and Sezan [2] have applied this assumption to high-resolution image reconstruction. Although translation is the least constraining and simplest to implement of the six coordinate transformations in Table 1, it is poor at handling large changes due to camera zoom, rotation, pan and tilt.

Zheng and Chellappa [3] considered a subset of the affine model — translation, rotation and scale — in image registration. Other researchers [4][5] have assumed affine motion (six parameters) between frames. For the assumptions of static scene and no parallax, the affine model exactly describes rotation about the optical axis of the camera, zoom of the camera, and pure shear, which the camera does not do, except in the limit as the lens focal length approaches infinity. The affine model cannot capture camera pan and tilt, and therefore cannot accurately express the "chirping" and "keystoning" we see in the real world (see Fig. 2). Consequently, the affine model tries to fit the wrong parameters to these effects. When the parameter estimation is not done properly to align the images, then a greater burden is placed on designing post-processing to enhance the poorly aligned images.

The 8-parameter *projective* model gives the exact eight desired parameters to account for all the possible camera motions. However, its parameters have traditionally been mathematically and computationally too hard to find. Consequently, a variety of approximations have been proposed. Before we present our new solution to estimating the projective parameters, it will be helpful to discuss these approximate models.

Going from first order (affine), to second order, gives the 12-parameter 'biquadratic' model. This model properly captures both the chirping (change in spatial frequency with position) and converging lines (keystoning) effects associated with projective coordinate transformations, although, despite its larger number of parameters, there is still considerable discrepancy between a projective coordinate transformation and the best-fit biquadratic coordinate transformation. Why stop at 2nd order? Why not use a 20-parameter 'bicubic model? While an increase in the number of model parameters will result in a better fit, there is a tradeoff, where the model begins to fit noise. The physical camera model fits exactly in the 8-parameter projective group; therefore, we know that "eight is enough." Hence, it is appealing to find an approximate model with only eight parameters.

The 8-parameter bilinear model is perhaps the most widely-used [6] in the fields of image processing, medical imaging, remote sensing, and computer graphics. This model is easily obtained from the biquadratic model by removing the four x^2 and y^2 terms. Although the resulting bilinear model captures the effect of converging lines, it completely fails to capture the effect of chirping.

The 8-parameter *pseudo-perspective* model [7]) does, in fact, capture both the converging lines and the chirping of a projective coordinate transformation. This model may be thought of as first, removal of two of the quadratic terms ($bfq_{x'y^2} = \mathbf{q}_{y'x^2} = 0$), which results in a ten parameter model (the 'q-chirp' of [8]) and then constraining the four remaining quadratic parameters to have two degrees of freedom. These constraints force the "chirping effect" (captured by $\mathbf{q}_{x'x^2}$ and $\mathbf{q}_{y'y^2}$) and the "converging effect" (captured by $\mathbf{q}_{x'xy}$ and $\mathbf{q}_{y'xy}$) to work together in the "right" way to match, as closely as possible, the effect of a projective coordinate transformation. By setting $\mathbf{q}_\alpha = \mathbf{q}_{x'x^2} = \mathbf{q}_{y'xy}$, the chirping in the x-direction is forced to correspond with the converging of parallel lines in the x-direction (and likewise for the y-direction). Therefore, of the 8-parameter approximations to the true projective, we would expect the *pseudo-perspective* model to perform the best. Indeed, we have verified this experimentally.

Model	Coordinate transformation from $\mathbf{x}$ to $\mathbf{x}'$	Parameters
Translation	$\mathbf{x}' = \mathbf{x} + \mathbf{b}$	$\mathbf{b} \in \mathbb{R}^2$
Affine	$\mathbf{x}' = \mathbf{A}\mathbf{x} + \mathbf{b}$	$\mathbf{A} \in \mathbb{R}^{2\times 2}, \mathbf{b} \in \mathbb{R}^2$
Bilinear	$x' = q_{x'xy}xy + q_{x'x}x + q_{x'y}y + q_{x'}$ $y' = q_{y'xy}xy + q_{y'x}x + q_{y'y}y + q_{y'}$	$bfq_* \in \mathbb{R}$
Projective	$\mathbf{x}' = \frac{\mathbf{A}\mathbf{x}+\mathbf{b}}{\mathbf{c}^T\mathbf{x}+1}$	$\mathbf{A} \in \mathbb{R}^{2\times 2}, \mathbf{b}, \mathbf{c} \in \mathbb{R}^2$
Pseudoperspective	$x' = q_{x'x}x + q_{x'y}y + q_{x'} + q_\alpha x^2 + q_\beta xy$ $y' = q_{y'x}x + q_{y'y}y + q_{y'} + q_\alpha xy + q_\beta y^2$	$\mathbf{q}_* \in \mathbb{R}$
Biquadratic	$x' = q_{x'x^2}x^2 + q_{x'xy}xy + q_{x'y^2}y^2 + q_{x'x}x + q_{x'y}y + q_{x'}$ $y' = q_{y'x^2}x^2 + q_{y'xy}xy + q_{y'y^2}y^2 + q_{y'x}x + q_{y'y}y + q_{y'}$	$bfq_* \in \mathbb{R}$

Table 1: Image coordinate transformations discussed in this paper

Of course, the desired "exact" eight parameters come from the projective model, but they have been notoriously difficult to estimate. The parameters for this model have been solved by Tsai and Huang [9], but their solution assumed that features had been identified in the two frames, along with their correspondences. In this paper, we present a simple featureless means of registering images.

Other researchers have looked at projective estimation in the context of obtaining $3-D$ models. Faugeras and Lustman [10], Shashua and Navab [11], and Sawhney [12] have considered the problem of estimating the projective parameters while computing the motion of a rigid planar patch, as part of a larger problem of finding 3-D motion and structure using parallax relative to an arbitrary plane in the scene. Kumar *et al.* [13] have also suggested registering frames of video by computing the flow along the *epipolar* lines, for which there is also an initial step of calculating the gross camera movement assuming no parallax. However, these methods have relied on feature correspondences, and were aimed at 3-D scene modeling. Our focus is not on recovering the 3-D scene model, but on aligning 2-D images of 3-D scenes. Feature correspondences greatly simplify the problem; however, they also have many problems which we review below. The focus of this paper is a simple featureless approach to estimating the projective coordinate transformation between image frames.

Two similar efforts exist to the new work presented here. Mann [14], and Szeliski and Coughlan [15] independently proposed featureless registration and compositing of either pictures of a nearly flat object, or of pictures taken from approximately the same location. Both used a 2-D projective model, and searched over its 8-parameter space to minimize the mean-square error (or maximize the inner product) between one frame and a 2-D projective coordinate transformation of the next frame. However, in both these earlier works the algorithm relies on nonlinear optimization techniques which we are able to avoid with the new technique presented here.

2.2 Camera motion: common assumptions and terminology

Two assumptions are typical in this area of research. The first assumption is that the scene is constant – changes of scene content and lighting are small between frames. The second assumption is that of an ideal pinhole camera – implying unlimited depth of field with everything in focus (infinite resolution) and implying that straight lines map to straight lines[1]. Consequently, the camera has three degrees of freedom in 2-D space and eight degrees of freedom in 3-D space: translation (X, Y, Z), zoom (scale in each of the image coordinates x and y), and rotation (rotation about the optical axis, pan, and tilt. These two assumptions are also made in this paper.

In this paper, an "uncalibrated camera" refers to one in which the principal point[2] is not necessarily at the center (origin) of the image and the scale is not necessarily isotropic[3] We assume that the zoom is continually adjustable by the camera user, and that we do not know the zoom setting, or whether it changed between recording frames of the image sequence. We also assume that each element in the camera sensor array returns a quantity that is linearly proportional to the quantity of light received[4]. With these assumptions, the exact camera motion that can be recovered is summarized in Table 2.

2.3 Video orbits

Tsai and Huang [9] pointed out that the elements of the projective *group* give the true camera motions with respect to a planar surface. They explored the group structure associated with images of a 3-D rigid planar patch, as well as the associated *Lie algebra*, although they assume that the correspondence problem has been solved. The solution presented in this paper (which does not require prior solution of correspondence) also relies on projective group theory. We briefly review the basics of this theory, before presenting the new solution in the next section.

[1]When using low cost wide-angle lenses, there is usually some barrel distortion which we correct using the method of [16].

[2]The principal point is where the optical axis intersects the film.

[3]Isotropic means that magnification in the x and y directions is the same. Our assumption facilitates aligning frames taken from different cameras.

[4]This condition can be enforced over a wide range of light intensity levels, by using the Wyckoff principle [17][18].

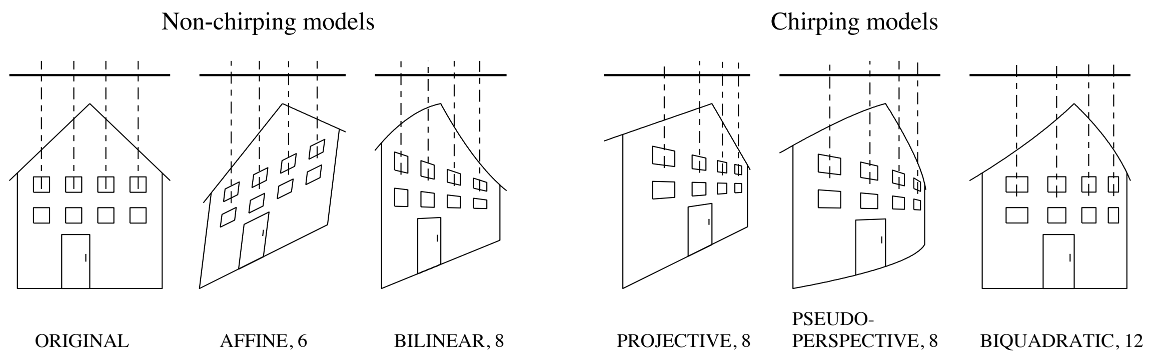

Figure 2: Pictorial effects of the six coordinate transformations of Table 1, arranged left to right by number of parameters. Note that translation leaves the ORIGINAL house unchanged, except in its location. Most importantly, only the three coordinate transformations at the right affect the periodicity of the window spacing (e.g. induce the desired "chirping" which corresponds to what we see in the real world.) Of these, only the PROJECTIVE coordinate transformation preserves straight lines. The 8-parameter PROJECTIVE coordinate transformation "exactly" describes the possible camera motions.

	Scene assumptions	**Camera assumptions**
Case 1:	arbitrary 3-D	free to zoom, rotate, pan, and tilt, fixed center of projection
Case 2:	planar	free to zoom, rotate, pan, and tilt, free to translate

Table 2: The two "no parallax" cases for a static scene.

2.3.1 Projective group in 1-D coordinates

For simplicity, we review the theory first for the projective coordinate transformation in one dimension[5]: $x' = (ax+b)/(cx+1)$, where the images are functions of one variable, x. The set of all projective coordinate transformations for which $a \neq 0$ forms a group, $\mathbf{P}$ the *projective group*. When $a \neq 0$ and $c = 0$, it is the affine group. When $a = 1$ and $c = 0$, it becomes the translation group.

Of the six coordinate transformations in the previous section, only the projective, affine, and translation operations form groups.

A group of operators together with the set of 1-D images (operands) form a *group operation*[6]. The new set of images that results from applying all possible operators from the group to a particular image from the original set is called the *orbit* of that image under the group operation [19].

The equivalent two cases of Table 2 for this hypothetical "flatland" world of 2-D objects with 1-D pictures correspond to the following. In the first case a camera is at a fixed location, and free to zoom and pan. In the second case, a camera is free to translate, zoom, and pan, but the imaged object must be flat (i.e., lie on a straight line in the plane). The resulting two (1-D) frames taken by the camera, are related by the coordinate transformation from x_1 to x_2, given by [20]:

$$\begin{aligned} x_2 &= z_2 \tan(\arctan(x_1/z_1) - \theta), \ \forall x_1 \neq o_1 \\ &= (ax_1 + b)/(cx_1 + 1), \ \forall x_1 \neq o_1 \end{aligned} \qquad (1)$$

[5]In a 2-D world, the "camera" consists of a center of projection (pinhole "lens") and a line (1-D sensor array or 1-D "film").

[6]also known as a *group action* or *G-set* [19].

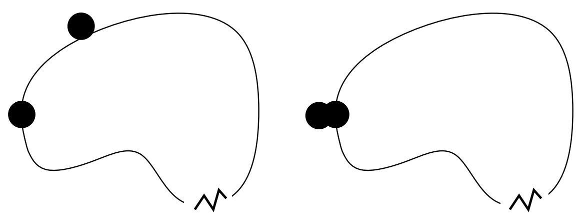

Figure 3: Video orbits. (a) The orbit of frame 1 is the set of all images that can be produced by acting on frame 1 with any element of the operator group. Assuming that frames 1 and 2 are from the same scene, frame 2 will be close to one of the possible projective coordinate transformations of frame 1. In other words, frame 2 "lies near the orbit of" frame 1. (b) By bringing frame 2 along its orbit, we can determine how closely the two orbits come together at frame 1.

where $a = z_2/z_1$, $b = -z_2\tan(\theta)$, $c = \tan(\theta)/z_1$, and $o_1 = z_1\tan(\pi/2 + \theta) = -1/c$, is the location of the singularity in the domain. We should emphasize that c, the degree of perspective, has been given the interpretation of a chirp-rate [20].

The coordinate transformations of (1) form a group operation. This result, and the proof of this group's isomorphism to the group corresponding to nonsingular projections of a flat object are given in [21].

2.3.2 Projective group in 2-D coordinates

The theory for the projective, affine, and translation groups also holds for the familiar 2-D images taken of the 3-D world. The 'video orbit' of a given 2-D frame is defined to be the set of all images that can be produced by applying operators from the 2-D projective group to the given image. Hence, we restate the coordinate transformation problem: Given a set of images that lie in the same orbit of the group, we wish to find for each image pair, that operator in the group which takes one image to the other image.

If two frames, say, f_1 and f_2, are in the same orbit, then there is an group operation $\mathbf{p}$ such that the mean-squared error (MSE) between f_1 and $f_2' = \mathbf{p} \circ f_2$ is zero. In practice, however, we find which element of the group takes one image "nearest" the other, for there will be a certain amount of parallax, noise, interpolation error, edge effects, changes in lighting, depth of focus, etc. Fig. 3 illustrates the operator $\mathbf{p}$ acting on frame f_2, to move it nearest to frame f_1. (This figure does not, however, reveal the precise shape of the orbit, which occupies an 8-D space.)

Summarizing, the 8-parameter projective group captures the exact coordinate transformation between pictures taken under the two cases of Table 2. The primary assumptions in these cases are that of no parallax, and of a static scene. Because the 8-parameter projective model is "exact," it is theoretically the right model to use for estimating the coordinate transformation. The examples which follow demonstrate that it also performs better in practice than the other proposed models. In the next section, we show a new technique for estimating its eight parameters.

3 Framework: motion parameter estimation and optical flow

Before presenting our new results, we briefly review existing methods of parameter estimation for coordinate transformations. We break existing methods into two categories: feature-based, and featureless. Of the featureless methods, we consider two subcategories: 1) methods based on minimizing MSE (generalized correlation, direct nonlinear optimization) and 2) methods based on spatiotemporal derivatives and optical flow. Note that variations such as *multiscale* have been omitted from these categories; multiscale analysis can be applied to any of them. The new algorithm we develop in this paper (with final form given in Sec. 4) is featureless, and based on multiscale spatiotemporal derivatives.

Some of the descriptions below will be presented for hypothetical 1-D images taken in a 2-D space. This simplification yields a clearer comparison of the estimation methods. The new theory and applications will be presented subsequently for 2-D images taken in a 3-D space.

3.1 Feature-based methods

Feature-based methods [22][23] assume that point correspondences in both images are available. In the projective case, given at least three correspondences between point pairs in the two 1-D images, we will find the element, $\mathbf{p} = \{a, b, c\} \in \mathbf{P}$ that maps the second image into the first. Let $x_k, k = 1, 2, 3, \ldots$ be the points in one image, and let x_k' be the corresponding points in the other image. Then: $x_k' = (ax_k + b)/(cx_k + 1)$. Re-arranging yields $ax_k + b - x_k x_k' c = x_k'$, so that a, b, and c can be found by solving $k \geq 3$ linear equations in 3 unknowns:

$$\left[\begin{array}{ccc} x_k & 1 & -x_k' x_k \end{array}\right] \left[\begin{array}{ccc} a & b & c \end{array}\right]^T = \left[\begin{array}{c} x_k' \end{array}\right] \qquad (2)$$

using least squares if there are more than three correspondence points. The extension from 1-D images to 2-D images is conceptually identical; for the affine and projective models, the minimum number of correspondence points needed in 2-D is three and four respectively.

A major difficulty with feature-based methods is finding the features. Good features are often hand-selected, or computed, possibly with some degree of human intervention [24]. A second problem with features is their sensitivity to noise and occlusion. Even if reliable features exist between frames (e.g. line markings on a playing field in a football video, see Sec. 5.4) these features may be subject to signal noise and occlusion (e.g. running football players blocking a feature). The emphasis in the rest of this paper will be on robust featureless methods.

3.2 Featureless methods based on generalized cross-correlation

Cross-correlation of two frames is a featureless method of recovering translation model parameters. Affine and projective parameters can also be recovered using generalized forms of cross-correlation.

Generalized cross-correlation is based on an inner-product formulation which establishes a similarity metric between two functions, say, g and h, where $h \approx \mathbf{p} \circ g$ is an approximately coordinate-transformed version of g, but the parameters of the coordinate transformation, $\mathbf{p}$ are unknown.[7] We can find, by exhaustive search (applying all possible operators, $\mathbf{p}$, to h), the "best" $\mathbf{p}$ as the one which maximizes the inner product:

$$\int_{-\infty}^{\infty} g(x) \frac{\mathbf{p}^{-1} \circ h(x)}{\int_{-\infty}^{\infty} \mathbf{p}^{-1} \circ h(x) dx} dx \qquad (3)$$

where we have normalized the energy of each coordinate-transformed h before making the comparison. Equivalently, instead of maximizing a similarity metric, we can minimize some distance metric, such as MSE, given by $\int_{-\infty}^{\infty} (g(x) - \mathbf{p}^{-1} \circ h(x))^2 - Dx$. Solving (3) has an advantage over finding MSE when one image is not only a coordinate-transformed version of the other, but is also an amplitude-scaled version, as generally happens when there is an automatic gain control or an automatic iris in the camera.

In 1-D, the affine model permits only dilating and translating. Given h, an affine coordinate-transformed version of g, generalized correlation amounts to estimating the parameters for dilation, a and translation, b by exhaustive search. The collection of all possible coordinate transformations, when applied to one of the images (say, h) serves

[7]In the presence of additive white Gaussian noise, this method, also known as "matched filtering", leads to a maximum likelihood estimate of the parameters [25].

to produce a family of templates to which the other image, g, can be compared. If we normalize each template so all have the same energy:

$$h_{a,b}(x) = \frac{1}{\sqrt{a}} h(ax+b)$$

then the maximum likelihood estimate corresponds to selecting the member of the family that gives the largest inner product:

$$\langle g(x), h_{a,b}(x) \rangle = \int_{-\infty}^{\infty} g(x) h_{a,b}(x) dx$$

This result is known as a *cross-wavelet transform.* A computationally efficient algorithm for the cross-wavelet transform has recently been presented [26]. (See [27] for a good review on wavelet-based estimation of affine coordinate transformations.)

Just like the cross-correlation for the translation group, and the cross-wavelet for the affine group, the 'cross-chirplet' can be used to find the parameters of a projective coordinate transformation in 1-D, searching over a 3-parameter space. The chirplet transform [28] is a generalization of the wavelet transform. The 'projective-chirplet', has the form:

$$h_{a,b,c} = h\left(\frac{ax+b}{cx+1}\right) \tag{4}$$

where h is the 'mother chirplet', analogous to the *mother wavelet* of wavelet theory. Members of this family of functions are related to one another by projective coordinate transformations.

With 2-D images, the search is over an 8-parameter space. A dense sampling of this volume is computationally prohibitive. Consequently, combinations of coarse-to-fine and iterative gradient-based search procedures are required. Adaptive variants of the chirplet transform have been previously reported in the literature [29]. However, there are still many problems with the adaptive chirplet approach; thus, we now consider featureless methods based on spatiotemporal derivatives.

3.3 Featureless methods based on spatiotemporal derivatives

3.3.1 Optical flow - "translation flow"

When the change from one image to another is small, optical flow [30] may be used. In 1-D, the traditional optical flow formulation assumes each point x in frame t is a translated version of the corresponding point in frame $t + \Delta t$, and that Δx and Δt are chosen in the ratio $\Delta x / \Delta t = u_f$, the translational flow velocity of the point in question. The image brightness $E(x,t)$ is described by:

$$E(x,t) = E(x + \Delta x, t + \Delta t), \quad \forall (x,t), \tag{5}$$

where u_f is the translational flow velocity of the point in In the case of pure translation, u_f is constant across the entire image. More generally, though, a pair of 1-D images are related by a quantity, $u_f(x)$ at each point in one of the images.

Expanding the right hand side of (5) in a Taylor series, and canceling 0th order terms gives the well-known optical flow equation: $u_f E_x + E_t + h.o.t. = 0$, where E_x and E_t are the spatial and temporal derivatives respectively, and $h.o.t.$ denotes higher order terms. Typically, the higher order terms are neglected, giving the expression for the optical flow at each point in one of the two images:

$$u_f E_x + E_t \approx 0 \tag{6}$$

3.3.2 "Affine fit" and "affine flow": a new relationship

Given the optical flow between two images, g and h, we wish to find the coordinate transformation to apply to h to make it look most like g. We now describe two approaches based on the affine model: (1) finding the optical flow at every point, and then fitting this flow with an affine model ('affine fit'), and (2) rewriting the optical flow equation in terms of an affine (not translation) motion model ('affine flow').

Wang and Adelson have proposed fitting an affine model to an optical flow field [31] of 2-D images. We briefly examine their approach with 1-D images (1-D images simplify analysis and comparison to other methods). Denote coordinates in the original image, g, by x, and in the new image, h, by x'. Suppose that h is a dilated and translated version of g, so $x' = ax + b$ for every corresponding pair (x', x). Equivalently, the affine model of velocity (normalizing $\Delta t = 1$), $u_m = x' - x$, is given by $u_m = (a-1)x + b$. We can expect a discrepancy between the flow velocity, u_f, and the model velocity, u_m, due to either errors in the flow calculation, or to errors in the affine model assumption, so we apply linear regression to get the best least-squares fit by minimizing:

$$\varepsilon_{fit} = \sum_x (u_m - u_f)^2 = \sum_x (u_m + E_t/E_x)^2 \tag{7}$$

The constants a and b that minimize ε_{fit} over the entire patch are found by differentiating (7), and setting the derivatives to zero. This results in what we call the "affine fit" equations:

$$\begin{bmatrix} \sum_x x^2, \sum_x x \\ \sum_x x, \sum_x 1 \end{bmatrix} \begin{bmatrix} a-1 \\ b \end{bmatrix} = - \begin{bmatrix} \sum_x x E_t/E_x \\ \sum_x E_t/E_x \end{bmatrix} \tag{8}$$

Alternatively, the affine coordinate transformation may be directly incorporated into the brightness change constraint equation (5). Bergen *et al.* [32] have proposed this method, which we will call 'affine flow', to distinguish it from the 'affine fit' model of Wang and Adelson (8). Let us show how 'affine flow' and 'affine fit' are related. Substituting $u_m = (ax+b) - x$ directly into (6) in place of u_f and summing the squared error:

$$\varepsilon_{flow} = \sum_x (u_m E_x + E_t)^2 \tag{9}$$

over the whole image, differentiating, and equating the result to zero, gives a linear solution for both a and b:

$$\begin{bmatrix} \sum_x x^2 E_x^2, \sum_x x E_x^2 \\ \sum_x x E_x^2, \sum_x E_x^2 \end{bmatrix} \begin{bmatrix} a-1 \\ b \end{bmatrix} = - \begin{bmatrix} \sum_x x E_x E_t \\ \sum_x E_x E_t \end{bmatrix} \tag{10}$$

To see how this result compares to the 'affine fit' we rewrite (7)

$$\varepsilon_{fit} = \sum_x (\frac{u_m E_x + E_t}{E_x})^2 \tag{11}$$

and observe, comparing (9) and (11) that 'affine flow' is equivalent to a weighted least-squares fit, where the weighting is given by E_x^2. Thus the 'affine flow' method tends to put more emphasis on areas of the image that are spatially varying than does the 'affine fit' method. Of course, one is free to separately choose the weighting for each method in such a way that 'affine fit' and 'affine flow' methods both give the same result. Both our intuition and our practical experience tends to favor the 'affine flow' weighting, but, more generally, perhaps we should ask "What is the best weighting?" (e.g. maybe there is an even better answer than the choice among these two). Lucas and Kanade [33], among others, have considered weighting issues.

Another approach to the 'affine fit' involves computation of the optical flow field using the multiscale iterative method of Lucas and Kanade, and *then* fitting to the affine model. An analogous variant of the 'affine flow' method involves multiscale iteration as well, but in this case the iteration and multiscale hierarchy are incorporated directly into the affine estimator [32]. With the addition of multiscale analysis, the 'fit' and 'flow' methods differ in additional respects beyond just the weighting. Our intuition and experience indicates that the direct multiscale 'affine flow' performs better than the 'affine fit' to the multiscale flow. Multiscale optical flow makes the assumption that blocks of the image are moving with pure translational motion , and then, paradoxically, the affine fit refutes this pure-translation assumption. However, 'fit' provides some utility over 'flow' when it is desired to segment the image into regions undergoing different motions [34], or to gain robustness by rejecting portions of the image not obeying the assumed model.

3.3.3 'Projective fit' and 'projective flow': new techniques

Analogous to the "affine fit" and "affine flow" of the previous section, we now propose two new methods: 'projective fit' and 'projective flow'. For the 1-D affine coordinate transformation, the graph of the range coordinate as a function of the domain coordinate is a straight line; for the projective coordinate transformation, the graph of the range coordinate as a function of the domain coordinate is a rectangular hyperbola [21]. The 'affine fit' case used linear regression; however, in the projective case we will use 'hyperbolic regression.' Consider the flow velocity given by (6) and the model velocity:

$$u_m = x' - x = \frac{ax+b}{cx+1} - x \tag{12}$$

and minimize the sum of the squared difference paralleling (9):

$$\varepsilon = \sum_x (\frac{ax+b}{cx+1} - x + \frac{E_t}{E_x})^2 \tag{13}$$

For 'projective-flow' ('p-flow') we use, as for affine flow, the Taylor series of u_m:

$$u_m + x = b + (a-bc)x + (bc-a)cx^2 + (a-bc)c^2x^3 + \cdots \tag{14}$$

and again use the first 3 terms, obtaining enough degrees of freedom to account for the 3 parameters being estimated. Letting $\epsilon = \sum(-h.o.t.)^2 = \sum((b + (a - bc - 1)x + (bc - a)cx^2)E_x + E_t)^2$, $\mathbf{q}_2 = (bc-a)c$, $\mathbf{q}_1 = a - bc - 1$, and $\mathbf{q}_0 = b$, and differentiating with respect to each of the 3 parameters of $\mathbf{q}$, setting the derivatives equal to zero, and verifying with the second derivatives, gives the linear system of equations for "projective flow":

$$\begin{bmatrix} \sum x^4 E_x^2 & \sum x^3 E_x^2 & \sum x^2 E_x^2 \\ \sum x^3 E_x^2 & \sum x^2 E_x^2 & \sum x E_x^2 \\ \sum x^2 E_x^2 & \sum x E_x^2 & \sum E_x^2 \end{bmatrix} \begin{bmatrix} q_2 \\ q_1 \\ q_0 \end{bmatrix} = - \begin{bmatrix} \sum x^2 E_x E_t \\ \sum x E_x E_t \\ \sum E_x E_t \end{bmatrix} \tag{15}$$

In Sec. 4 we will extend this derivation to 2-D images and show how an iterative approach may be used to compute the parameters, $\mathbf{p}$, of the exact model, by using a feedback system where the feedforward loop involves computation of the approximate parameters, $\mathbf{q}$ in the extension of (15) to 2-D.

As with the affine case, 'projective fit'and 'projective flow' (15) differ only in the weighting assumed, although 'projective fit' provides the added advantage of enabling the motion within an arbitrary subregion of the image to be easily found. In this paper we are only considering global image motion, for which we have found the projective flow to be best.

4 Multiscale 'projective flow' parameter estimation

In the previous section, two new techniques, 'p-fit' and 'p-flow' were proposed. Now we describe our algorithm for estimating the projective coordinate transformation for 2-D images using 'p-flow'. We begin with the brightness constancy constraint equation for 2-D images [30] which gives the flow velocity components in the x and y directions, analogous to (6):

$$u_f E_x + v_f E_y + E_t \approx 0 \tag{16}$$

As is well-known [30] the optical flow field in 2-D is underconstrained[8]. The model of *pure translation* at every point has two parameters, but there is only one equation (16) to solve, thus it is common practice to compute the optical flow over some neighborhood, which must be at least two pixels, but is generally taken over a small block, 3×3, 5×5, or sometimes larger (e.g. the entire image, as in this paper).

Our task is not to deal with the 2-D translation flow, but with the 2-D projective flow, estimating the eight parameters in the coordinate transformation:

$$\mathbf{x}' = \begin{bmatrix} x' \\ y' \end{bmatrix} = \frac{\mathbf{A}[x,y]^T + \mathbf{b}}{\mathbf{c}^T[x,y]^T + 1} = \frac{\mathbf{A}\mathbf{x} + \mathbf{b}}{\mathbf{c}^T\mathbf{x} + 1} \tag{17}$$

The desired eight scalar parameters are denoted by $\mathbf{p} = [\mathbf{A}, \mathbf{b}; \mathbf{c}, 1]$, $\mathbf{A} \in \mathbb{R}^{2\times2}$, $\mathbf{b} \in \mathbb{R}^{2\times1}$, and $\mathbf{c} \in \mathbb{R}^{2\times1}$.

As with the 1-D images, we make similar assumptions in expanding (17) in its own Taylor series, analogous to (14).

[8]Optical flow in 1-D did not suffer from this problem.

If we take the Taylor series up to 2nd order terms, we obtain the biquadratic model mentioned in Sec. 2.1. As mentioned in Sec. 2.1, by appropriately constraining the twelve parameters of the biquadratic model we obtain a variety of 8-parameter approximate models. In our algorithm for estimating the exact projective group parameters, we will use one of these approximate models in an intermediate step.[9] We illustrate the algorithm below using the bilinear approximate model since it has the simplest notation.[10] First we will incorporate the approximate model directly into the generalized fit or generalized flow. The Taylor series for the bilinear case gives:

$$\begin{aligned} u_m + x &= q_{x'xy}xy + q_{x'x}x + q_{x'y}y + q_{x'} \\ v_m + y &= q_{y'xy}xy + q_{y'x}x + q_{y'y}y + q_{y'} \end{aligned} \tag{18}$$

Incorporating these into the flow criteria yields a simple set of eight linear equations in eight unknowns, for "bilinear flow", appearing in (19). The summations are over the entire image (all x and y) if computing global motion (as is done in this paper), or over a windowed patch if computing local motion. This equation looks similar to the 6×6 matrix equation presented in Bergen *et al.* [32].

In order to see how well the model describes the coordinate transformation between 2 images, say, g and h, one might *warp*[11] h to g, using the estimated motion model, and then compute some quantity that indicates how different the resampled version of h is from g. The MSE between the reference image and the warped image might serve as a good measure of similarity. However, since we are really interested in how the *exact model* describes the coordinate transformation, we assess the goodness of fit by first relating the parameters of the approximate model to the exact model, and then find the MSE between the reference image and the comparison image after applying the coordinate transformation of the exact model. A method of finding the parameters of the exact model, given the approximate model, is presented in Sec 4.1.

4.1 'Four point method' for relating approximate model to exact model

Any of the approximations above, after being related to the exact projective model, tend to behave well in the neighborhood of the identity, $\mathbf{A} = \mathbf{I}, \mathbf{b} = \mathbf{0}, \mathbf{c} = \mathbf{0}$. In 1-D, we explicitly expanded the model Taylor series about the identity; here, although we do not explicitly do this, we shall assume that the terms of the Taylor series of the model correspond to those taken about the identity. In the 1-D case we solve the 3 linear equations in 3 unknowns to estimate the parameters of the approximate motion model, and then relate the terms in this Taylor series to the exact parameters, a, b, and c (which involves solving another set of 3 equations in 3 unknowns, the second set being nonlinear, although very easy to solve).

In the extension to 2-D, the estimate step is straightforward, but the relate step is more difficult, because we now have eight nonlinear equations in eight unknowns, relating the terms in the Taylor series of the approximate model to the desired exact model parameters. Instead of solving these equations directly, we now propose a simple procedure for relating the parameters of the approximate model to those of the exact model, which we call the 'four point method':

1. Select four ordered pairs (e.g. the four corners of the bounding box containing the region under analysis, or the four corners of the image if the whole image is under analysis). Here suppose, for simplicity, that these points are the corners of the unit square: $\mathbf{s} = [s_1, s_2, s_3, s_4] = [(0,0)^T, (0,1)^T, (1,0)^T, (1,1)^T]$.
2. Apply the coordinate transformation using the Taylor series for the approximate model (e.g. (18)) to these points: $\mathbf{r} = \mathbf{u}_m(\mathbf{s})$.
3. Finally, the correspondences between $\mathbf{r}$ and $\mathbf{s}$ are treated just like features. This results in four easy to solve linear equations:

$$\begin{bmatrix} x'_k \\ y'_k \end{bmatrix} = \begin{bmatrix} x_k, y_k, 1, 0, 0, 0, -x_k x'_k, -y_k x'_k \\ 0, 0, 0, x_k, y_k, 1, -x_k y'_k, -y_k y'_k \end{bmatrix} \begin{bmatrix} a_{x'x}, a_{x'y}, b_{x'}, a_{y'x}, a_{y'y}, b_{y'}, c_x, c_y \end{bmatrix}^T \tag{20}$$

   where $1 \le k \le 4$. This results in the exact eight parameters, $\mathbf{p}$.

We remind the reader that the four corners are **not** feature correspondences as used in the feature-based methods of Sec. 3.1, but, rather, are used so that the two featureless models (approximate and exact) can be related to one another.

It is important to realize the full benefit of finding the exact parameters. While the "approximate model" is sufficient for small deviations from the identity, it is not adequate to describe large changes in perspective. However, if we use it to track small changes incrementally, and each time relate these small changes to the exact model (17), then we can accumulate these small changes using the *law of composition* afforded by the group structure. This is an especially favorable contribution of the group framework. For example, with a video sequence, we can accommodate very large accumulated changes in perspective in this manner. The problems with cumulative error can be eliminated, for the most part, by constantly propagating forward the true values, computing the residual using the approximate model, and each time relating this to the exact model to obtain a goodness-of-fit estimate.

4.2 New algorithm for 'projective flow': overview

Below is an outline of the algorithm; details of each step are in subsequent sections.

[9]Use of an approximate model that doesn't capture chirping or preserve straight lines can still lead to the true projective parameters as long as the model captures at least eight degrees of freedom.

[10]The pseudo-perspective gives slightly better performance; its development is the same but with more notation.

[11]The term *warp* is appropriate here, since the approximate model does not preserve straight lines.

Equation 19

$$\begin{bmatrix}
\sum x^2y^2E_x^2, & \sum x^2yE_x^2, & \sum xy^2E_x^2, & \sum xyE_x, & \sum x^2y^2E_yE_x, & \sum x^2yE_yE_x, & \sum xy^2E_yE_x, & \sum E_yxyE_x \\
\sum x^2yE_x^2, & \sum x^2E_x^2, & \sum xyE_x^2, & \sum xE_x^2, & \sum x^2yE_yE_x, & \sum x^2E_yE_x, & \sum xyE_yE_x, & \sum E_yxE_x \\
\sum xy^2E_x^2, & \sum xyE_x^2, & \sum y^2E_x^2, & \sum yE_x^2, & \sum xy^2E_yE_x, & \sum xyE_yE_x, & \sum y^2E_yE_x, & \sum E_yyE_x \\
\sum xyE_x^2, & \sum xE_x^2, & \sum yE_x^2, & \sum E_x^2, & \sum xyE_yE_x, & \sum xE_yE_x, & \sum yE_yE_x, & \sum E_yE_x \\
\sum x^2y^2E_xE_y, & \sum x^2yE_xE_y, & \sum xy^2E_xE_y, & \sum E_xxyE_y, & \sum x^2y^2E_y^2, & \sum x^2yE_y^2, & \sum xy^2E_y^2, & \sum xyE_y^2 \\
\sum x^2yE_xE_y, & \sum x^2E_xE_y, & \sum xyE_xE_y, & \sum E_xxE_y, & \sum x^2yE_y^2, & \sum x^2E_y^2, & \sum xyE_y^2, & \sum xE_y^2 \\
\sum xy^2E_xE_y, & \sum xyE_xE_y, & \sum y^2E_xE_y, & \sum E_xyE_y, & \sum xy^2E_y^2, & \sum xyE_y^2, & \sum y^2E_y^2, & \sum yE_y^2 \\
\sum xyE_xE_y, & \sum xE_xE_y, & \sum yE_xE_y, & \sum E_xE_y, & \sum xyE_y^2, & \sum xE_y^2, & \sum yE_y^2, & \sum E_y^2
\end{bmatrix}
\begin{bmatrix} q_{x'xy} \\ q_{x'x} \\ q_{x'y} \\ q_{x'} \\ q_{y'xy} \\ q_{y'x} \\ q_{y'y} \\ q_{y'} \end{bmatrix}$$

$$= -\left[\ \sum E_txyE_x, \sum E_txE_x, \sum E_tyE_x, \sum E_tE_x, \sum E_txyE_y, \sum E_txE_y, \sum E_tyE_y, \sum E_tE_y\ \right]^T$$

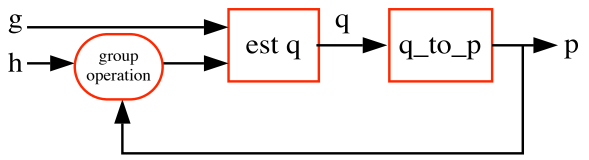

Figure 4: Method of computation of eight parameters $\mathbf{p}$ between two images from the same pyramid level, g and h. The approximate model parameters $\mathbf{q}$ are related to the exact model parameters $\mathbf{p}$ in a feedback system.

Frames from an image sequence are compared pairwise to test whether or not they lie in the same orbit:

1. A Gaussian pyramid of three or four levels is constructed for each frame in the sequence.
2. The parameters $\mathbf{p}$ are estimated at the top of the pyramid, between the two lowest-resolution images of a frame pair, g and h, using the iterative method depicted in Fig. 4.
3. The estimated $\mathbf{p}$ is applied to the next higher-resolution (finer) image in the pyramid, $\mathbf{p} \circ g$, to make the two images at that level of the pyramid nearly congruent before estimating the $\mathbf{p}$ between them.
4. The process continues down the pyramid until the highest-resolution image in the pyramid is reached.

4.3 Multiscale iterative implementation

The Taylor-series formulations we have used implicitly assume smoothness; the performance is improved if the images are blurred before estimation. To accomplish this, we do not downsample critically after low-pass filtering in the pyramid. However, after estimation, we use the original (unblurred) images when applying the final coordinate transformation.

The strategy we present differs from the multiscale iterative (affine) strategy of Bergen *et al.* in one important respect beyond simply an increase from six to eight parameters. The difference is the fact that we have two motion models, the 'exact motion model' (17) and the 'approximate motion model', namely the Taylor series approximation to the motion model itself. The approximate motion model is used to iteratively converge to the exact motion model, using the algebraic *law of composition* afforded by the exact projective group model. In this strategy, the exact parameters are determined at each level of the pyramid, and passed to the next level. The steps involved are summarized schematically in Fig. 4, and described below:

1. Initialize: Set $h_0 = h$ and set $\mathbf{p}_{0,0}$ to the identity operator.
2. Iterate ($k = 1 \ldots K$):
   (a) **ESTIMATE:** Estimate the 8 or more terms of the approximate model between two image frames, g and h_{k-1}. This results in approximate model parameters $\mathbf{q}_k$.
   (b) **RELATE:** Relate the approximate parameters $\mathbf{q}_k$ to the exact parameters using the 'four point method'. The resulting exact parameters are $\mathbf{p}_k$.
   (c) **RESAMPLE:** Apply the *law of composition* to accumulate the effect of the $\mathbf{p}_k$'s. Denote these composite parameters by $\mathbf{p}_{0,k} = \mathbf{p}_k \circ \mathbf{p}_{0,k-1}$. Then set $h_k = \mathbf{p}_{0,k} \circ h$. (This should have nearly the same effect as applying $\mathbf{p}_k$ to h_{k-1}, except that it will avoid additional interpolation and anti-aliasing errors you would get by resampling an already resampled image[6]).

Repeat until either the error between h_k and g falls below a threshold, or until some maximum number of iterations is achieved. After the first iteration, the parameters $\mathbf{q}_2$ tend to be near the identity since they account for the residual between the "perspective-corrected" image h_1 and the "true" image g. We find that only two or three iterations are usually needed for frames from nearly the same orbit.

A rectangular image assumes the shape of an arbitrary quadrilateral when it undergoes a projective coordinate transformation. In coding the algorithm, we pad the undefined portions with the quantity NaN, a standard IEEE arithmetic value, so that any calculations involving these values automatically inherit NaN without slowing down the computations. The algorithm (in Matlab on an HP 735) takes about six seconds per iteration for a pair of 320x240 images.

4.4 Exploiting commutativity for parameter estimation

There is a fundamental uncertainty [35] involved in the simultaneous estimation of parameters of a noncommutative group, akin to the Heisenberg uncertainty relation of quantum mechanics. In contrast, for a commutative[12] group (in the absence of noise), we can obtain the exact coordinate transformation.

Segman [36] considered the problem of estimating the parameters of a commutative group of coordinate transformations, in particular, the parameters of the affine group [37]. His work also deals with noncommutative groups, in particular, in the incorporation of scale in the Heisenberg group[13] [38].

Estimating the parameters of a commutative group is computationally efficient, e.g., through the use of Fourier cross-spectra [39]. We exploit this commutativity for estimating the parameters of the noncommutative 2-D projective group by first estimating the parameters that commute. For example, we improve performance if we first estimate the two parameters of translation, correct for the translation, and then proceed to estimate the eight projective parameters. We can also simultaneously estimate both the isotropic-zoom and the rotation about the optical axis by applying a log-polar coordinate transformation followed by a translation estimator. This process may also be achieved by a direct application of the Fourier-Mellin transform [40]. Similarly, if the only difference between g and h is a camera pan, then the pan may be estimated through a coordinate transformation to cylindrical coordinates, followed by a translation estimator.

In practice, we run through the following 'commutative initialization' before estimating the parameters of the projective group of coordinate transformations:

1. Assume that h is merely a translated version of g.
   (a) Estimate this translation using the method of Girod [39].
   (b) Shift h by the amount indicated by this estimate.
   (c) Compute the *MSE* between the shifted h and g, and compare to the original MSE before shifting.
   (d) If an improvement has resulted, use the shifted h from now on.
2. Assume that h is merely a rotated and isotropically zoomed version of g.
   (a) Estimate the two parameters of this coordinate transformation.
   (b) Apply these parameters to h.
   (c) If an improvement has resulted, use the coordinate-transformed (rotated and scaled) h from now on.
3. Assume that h is merely an "x-chirped" (panned) version of g, and, similarly, 'x-dechirp' h. If an improvement results, use the 'x-dechirped' h from now on. Repeat for y (tilt.)

Compensating for one step may cause a change in choice of an earlier step. Thus it might seem desirable to run through the commutative estimates iteratively. However, our experience on lots of real video indicates that a single pass usually suffices, and in particular, will catch frequent situations where there is a pure zoom, a pure pan, a pure tilt, etc, both saving the rest of the algorithm computational effort, as well as accounting for simple coordinate transformations such as when one image is an upside-down version of the other. (Any of these pure cases corresponds to a single parameter group, which is commutative.) Without the 'commutative initialization' step, these parameter estimation algorithms are prone to get caught in local optima, and thus never converge to the global optimum.

[12] A commutative (or *Abelian*) group is one in which elements of the group commute, for example, translation along the x-axis commutes with translation along the y-axis, so the 2-D translation group is commutative.

[13] While the Heisenberg group deals with translation and frequency-translation (modulation), some of the concepts could be carried over to other more relevant group structures.

5 Performance/Applications

5.1 A paradigm reversal in resolution enhancement

Much of the previous work on resolution enhancement [4][41][42] has been directed toward military applications, where one cannot get close to the subject matter; therefore, lenses of very long focal length were generally used. In this case, there was very little change in *perspective* and the motion could be adequately approximated as affine. Budgets also permitted lenses of exceptionally high quality, so that the resolving power of the lens far exceeded the resolution of the sensor array.

Sensor arrays in earlier applications generally had a small number of pixels compared to today's sensors, leaving considerable "dead space" between pixels. Consequently, using multiple frames from the image sequence to fill in gaps between pixels was perhaps the single most important consideration in combining multiple frames of video.

We argue that in the current age of consumer video, the exact opposite is generally true: subject matter generally subtends a larger angle (e.g. is either closer, or more *panoramic* in content), and the desire for low cost has led to cheap plastic lenses that have very large distortion. Moreover, sensor arrays have improved dramatically. Accurate solution of the projective model is more important than ever in these new applications.

In addition to consumer video, we believe there will be a large market in the future for small wearable wireless cameras. A prototype, the "wearable wireless webcam" (a head-mounted video camera uplinked to the Internet [43]) has provided one of the most extreme testbeds for the algorithms explored in our research, as it captures noisy transmitted video frames, grabbed by a camera attached to a human head, free to move at the will of the individual. The projective model is especially well-suited to this new application, as a person can turn their head (camera rotation about an approximately fixed center of projection) much faster than they can undergo locomotion (camera translation). The new algorithm described in this paper has consistently performed well on noisy data gathered from the headcam, even when the scene is not static and there is parallax.

5.1.1 Four ways by which "resolution" may be enhanced

1. **Sub-pixel** "filling in the gaps":
2. **Scene widening**: Increased spatial extent; stitching together images in a panorama.
3. **Saliency**: suppose we have a wide shot of a scene, and then zoom into one person's face in the scene. In order to insert the face without downsampling it, we need to upsample the wide shot, increasing the meaningful pixel count of the whole image.
4. **Perspective**: in order to seamlessly mosaic images from panning with a wide angle lens, images need to be brought into a common system of coordinates resulting in a "keystoning" effect on the previously rectangular image boundary. Thus we must hold the pixel resolution constant on the "squashed" side and upsample on the "stretched" side, resulting in increased *pixel resolution* of the entire mosaic.

The first of these four may arise from either microscopic camera movement (inducing image motion on the order of a pixel or less) or macroscopic camera movement (inducing motion on the order of many pixels). However, as movement increases, errors in registration will tend to increase, and enhancement due to (1) will be reduced, while the enhancement due to (2), (3), and (4) will increase.

Results of applying the proposed method to subpixel resolution enhancement are not presented in this paper but may be found in [21].

5.2 Increasing "resolution" in the 'pixel sense'

Figure 5 shows some frames from a typical image sequence. Figure 6 shows the same frames brought into the coordinate system of frame (c), that is, the middle frame was chosen as the *reference frame.*

Given that we have established a means of estimating the projective coordinate transformation between any pair of images, there are two basic methods we use for finding the coordinate transformations between all pairs of a longer image sequence. Because of the group structure of the projective coordinate transformations, it suffices to arbitrarily select one frame and find the coordinate transformation between every other frame and this frame. The two basic methods are:

1. **Differential parameter estimation**: The coordinate transformations between successive pairs of images, $\mathbf{p}_{0,1}$, $\mathbf{p}_{1,2}$, $\mathbf{p}_{2,3}$, ..., estimated.
2. **Cumulative parameter estimation**: The coordinate transformation between each image and the reference image is estimated directly. Without loss of generality, select frame zero (E_0) as the reference frame and denote these coordinate transformations as $\mathbf{p}_{0,1}$, $\mathbf{p}_{0,2}$, $\mathbf{p}_{0,3}$, ...

Theoretically, the two methods are equivalent:

$$\begin{aligned} E_0 &= p_{0,1} \circ p_{1,2} \circ \ldots \circ p_{n-1,n} E_n \quad \text{differential method} \\ E_0 &= p_{0,n} E_n \quad \text{cumulative method} \end{aligned} \tag{21}$$

However, in practice, the two methods differ for two reasons:

1. **cumulative error**: In practice, the estimated coordinate transformations between pairs of images register them only approximately, due to violations of the assumptions (e.g. objects moving in the scene, center of projection not fixed, camera swings around to bright window and automatic iris closes, etc.). When a large number of estimated parameters are composed, cumulative error sets in.
2. **finite spatial extent of image plane**: Theoretically, the images extend infinitely in all directions, but, in practice, images are cropped to a rectangular bounding box. Therefore, a given pair of images (especially if they are far from adjacent in the orbit) may not overlap at all; hence it is not possible to estimate the parameters of the coordinate transformation using those two frames.

The frames of Fig 5 were brought into register using the differential parameter estimation, and "cemented" together seamlessly on a common canvas. "Cementing" involves piecing the frames together, for example, by median, mean, or trimmed mean, or combining on a subpixel grid [21]. (Trimmed mean was used here, but the particular method made little visible difference.) Fig 7 shows this result ("projective/projective"), with a comparison to two non-projective cases. The first comparison is to "affine/affine" where affine parameters were estimated (also multiscale) and used for the coordinate transformation. The second comparison, "affine/projective," uses the six affine parameters found by estimating the eight projective parameters and ignoring the two "chirp" parameters $\mathbf{c}$ (which capture the essence of tilt and pan). These six parameters $\mathbf{A}$, $\mathbf{b}$ are more accurate than those obtained using the affine estimation, as the affine estimation tries to fit its shear parameters to the camera pan and tilt. In other words, the affine estimation does worse than the six affine parameters within the projective estimation. The affine coordinate transform is finally applied, giving the image shown. Note that the coordinate-transformed frames in the affine case are parallelograms.

5.3 Submosaics and the support matrix

Two situations have so far been dealt with:

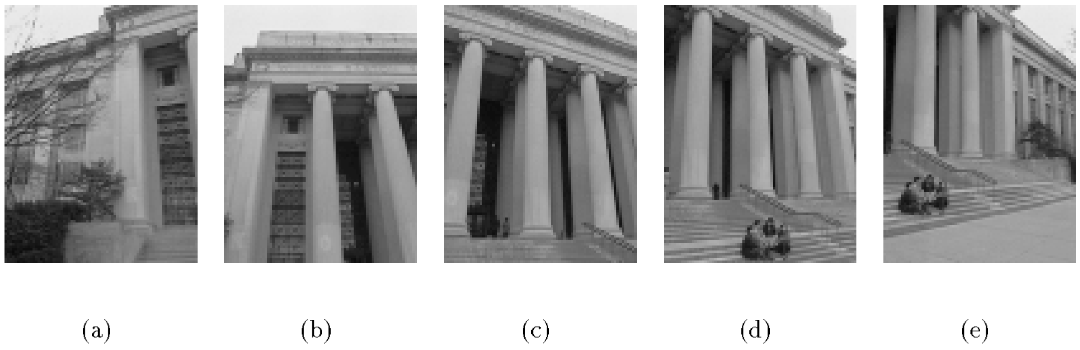

Figure 5: Frames from original image orbit, sent from the wearable amateur television transmitter ("netcam"[15]). The entire sequence, consisting of 20 color frames, is available (see note at end of bibliography), together with examples of applying the proposed algorithm to this data.

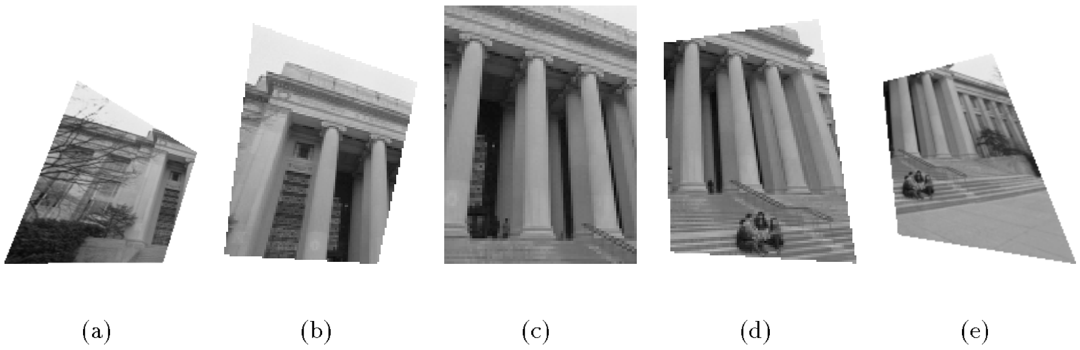

Figure 6: Frames from original image video orbit after a coordinate transformation to move them along the orbit to the reference frame (c). The coordinate-transformed images are alike except for the region over which they are defined. Note that the regions are not parallelograms; thus, methods based on the affine model fail.

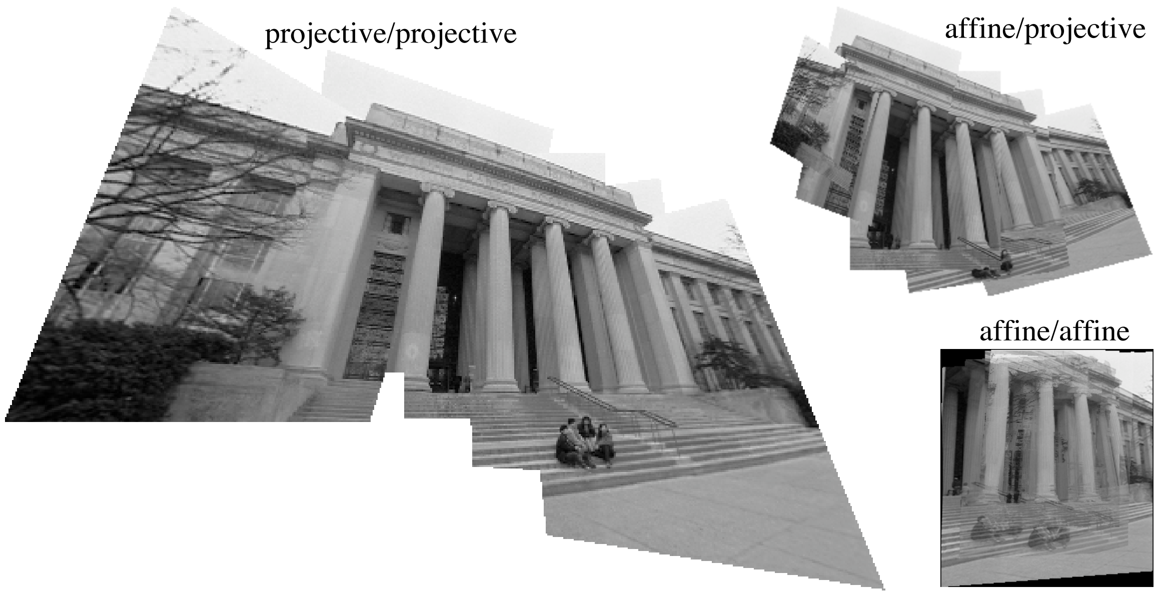

Figure 7: Frames of Fig 6 "cemented" together on single image "canvas", with comparison of affine and projective models. Note the good registration and nice appearance of the projective/projective image despite the noise in the amateur television receiver, wind-blown trees, and the fact that the rotation of the camera was not actually about its center of projection. To see this image in color, look at [44], where additional examples (e.g. some where the algorithm still worked despite "crowd noise" where many people were entering and leaving the building) also appear. Note also that the affine model fails to properly estimate the motion parameters (affine/affine), and even if the "exact" projective model is used to **estimate** the affine parameters, there is no affine coordinate transformation that will properly register all of the image frames.

1. The camera movement is small, so that any pair of frames chosen from the video orbit have a substantial amount of overlap when expressed in a common coordinate system. (Use differential parameter estimation.)
2. The camera movement is monotonic, so that any errors that accumulate along the registered sequence are not particularly noticeable. (Use cumulative parameter estimation.)

In the example of Fig 7, any cumulative errors are not particularly noticeable because the camera motion is progressive, that is, it does not reverse direction, or loop around on itself. Now let us look at an example where the camera motion loops back on itself and small errors, due to violations of the assumptions (fixed camera location and static scene), accumulate.

Consider the image sequence shown in Fig 8. The image mosaic arising from bringing these 16 image frames into the coordinates of the first frame exhibited somewhat poor registration due to cumulative error; we use this case to illustrate the importance of submosaics.

The 'differential support matrix[16]', for which the entry $\mathfrak{q}_{m,n}$ tells us how much frame n overlaps with frame m when expressed in the coordinates of frame m, for the sequence of Fig 8 appears in Fig 9.

Examining the support matrix, and the mean-squared error estimates, the local maxima of the support matrix correspond to the local minima of the mean-squared error estimates, suggesting the submosaics[17]: $\{7, 8, 9, 10, 6, 5\}$, $\{1, 2, 3, 4\}$, and $\{15, 14, 13, 12\}$. It is important to note that when the error is low, if the support is also low, the error estimate might not be valid. For example if the two images overlap in only one pixel, then even if the error estimate is zero (e.g. perhaps that pixel has a value of 255 in both images) the alignment is not likely good.

The selected submosaics appear in Fig 10. Estimating the coordinate transformation between these submosaics, and putting them together into a common frame of reference results in a nice image mosaic (Fig 10) about 1200 pixels across, where the image is sharp despite the fact that the person in the picture was moving slightly and the camera operator was also moving (violating the assumptions of both static scene and fixed center of projection).

5.4 Flat subject matter and alternate coordinates

Many sports such as football or soccer are played on a nearly flat field that forms a rigid planar patch over which the analysis may be conducted. After each of the frames

[16]The 'differential support matrix' is not necessarily symmetric, while the 'cumulative support matrix' for which the entry $bfq_{m,n}$ tells us how much frame n overlaps with frame m when expressed in the coordinates of frame 0 (reference frame) is symmetric.

[17]Researchers at Sarnoff also consider the use of submosaics, and refer to them as *tiles* [41][42]

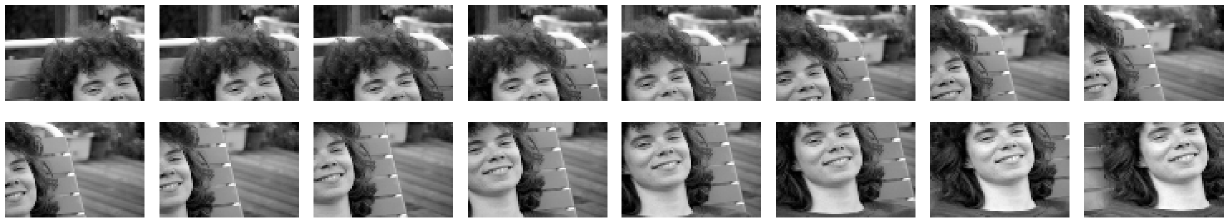

Figure 8: The Hewlett Packard "Claire" image sequence, which violates the assumptions of the model (the camera location was not fixed, and the scene was not completely static). Images appear in TV raster-scan order.

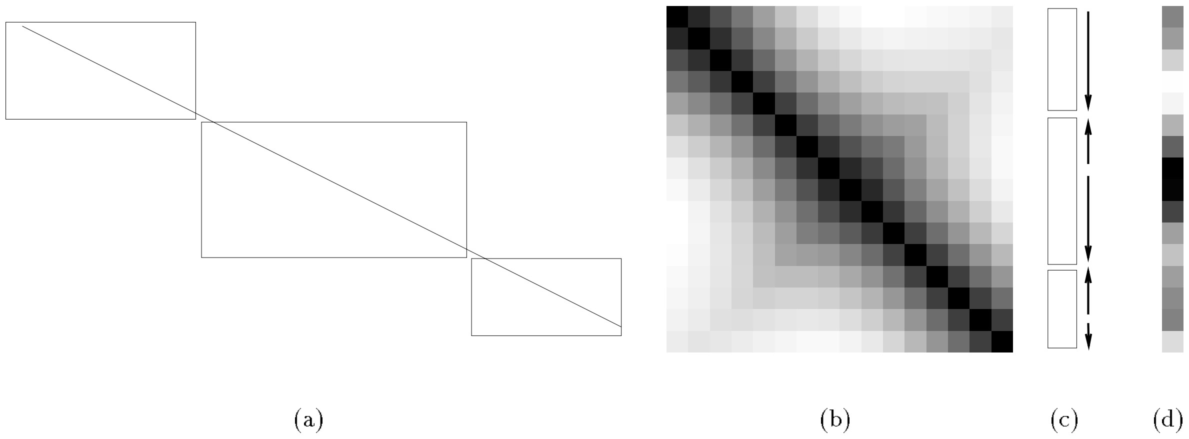

Figure 9: Support matrix and mean-squared registration error defined by image sequence in Fig 8 and the estimated coordinate transformations between images. (a) entries in table. The diagonals are one since every frame is fully supported in itself. The entries just above (or below) the diagonal give the amount of pairwise support. For example, frames 0 and 1 share high mutual support (.91). Frames 7, 8, and 9 also share high mutual support (again .91). (b) corresponding *density plot* (more dense ink indicates higer values). (c) mean-square registration error (d) corresponding density plot

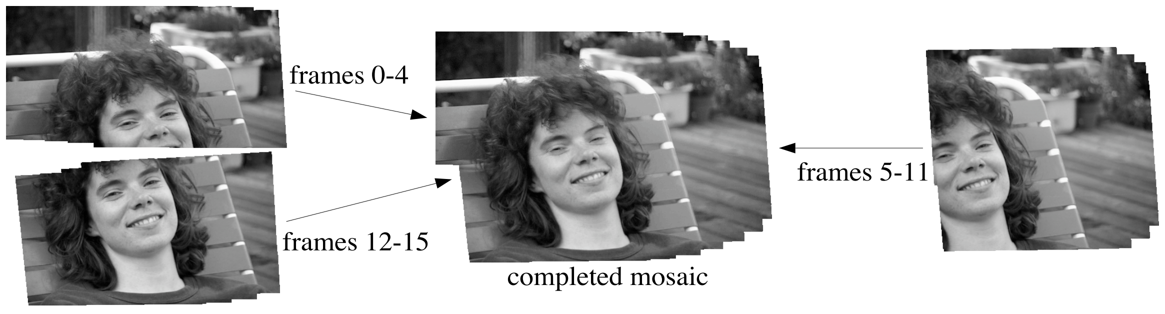

Figure 10: Submosaics are each made from subsets of the images that share high quantities of mutual support and low estimates of mutual error, and then combined to form the final mosiac.

undergoes the appropriate coordinate transformation to bring it into the same coordinate system as the reference frame, the sequence can be played back showing only the players (and the image boundaries) moving. Markings on the field (such as numbers and lines) remain at a fixed location, which makes subsequent analysis and summary of the video content easier. This data makes a good test case for the algorithms because the video was noisy and the players caused the assumption of static scene to be violated.

Despite the players moving in the video, the proposed method successfully registers all of the images in the orbit, mapping them into a single high-resolution image mosaic of the entire playing field. Figure 11(a) shows 16 frames of video from a football game combined into a single image mosaic, expressed in the coordinates of the first image in the sequence. The choice of coordinate system was arbitrary, and any of the images could have been chosen as the reference frame. In fact, a coordinate system other than one chosen from the input images could also be used. In particular, a coordinate system where *parallel lines never meet*, and periodic structures are "dechirped" (Fig 11(b)) lends itself well to machine vision and player-tracking algorithms[45]. Even if the entire playing field was never visible in any one image, collectively, the video from an entire game will likely reveal every square yard of playing surface at one time or another, hence enabling us to make a mosaic of the entire playing surface.

6 Conclusions

We presented new connections between different motion estimation approaches, in particular, a relation between "affine fit" and "affine flow." This led us to propose two new techniques, "projective fit" and "projective flow" which estimate the projective (homographic) coordinate transformation between pairs of images, taken with a camera that is free to pan, tilt, rotate about its optical axis, and zoom. A new multiscale iterative algorithm for projective flow was presented and applied to mosaicing. The algorithm solves for the 8 parameters of the "exact" model (the projective group of coordinate transformations), is fully automatic, and converges quickly. The use of the algorithm with submosaics, useful when the camera motion loops back on itself, has also been demonstrated.

The proposed method was found to work well on image data collected from both good-quality and poor-quality video under a wide variety of conditions (sunny, cloudy, day, night). It has been tested with a head-mounted wireless video camera, and performs successfully even in the presence of noise, interference, scene motion (such as people walking through the scene) and parallax (such as the wearer's head moving freely.)

7 Acknowledgement

The authors would like to express their thanks to many individuals for their suggestions and encouragement. In particular, thanks to Lee Campbell for help with the correction of barrel distortion (enabling really cheap plastic lenses to be used), Shawn Becker, John Wang, Nassir Navab, Ujjaval Desai, Chris Graczyk, Walter Bender, Fang Liu, and Constantine Sapuntzakis of MIT, and Alex Drukarev and Jeanne Wiseman at HP Labs, Palo Alto. Thanks to Paul and Claire Hubel of HP Labs for the use of the "Claire" image sequence. Some of the software to implement the 'p-chirp' models was developed in collaboration with Shawn Becker.

References

[1] S. B. J.L. Barron, D.J. Fleet, "Systems and experiment performance of optical flow techniques," *International journal of computer vision*, pp. 43–77, 1994.

[2] A. Tekalp, M. Ozkan, and M. Sezan, "High-resolution image reconstruction from lower-resolution image sequences and space-varying image restoration," in *Proc. of the Int. Conf. on Acoust., Speech and Sig. Proc.*, (San Francisco, CA), pp. III–169, IEEE, Mar. 23-26, 1992.

[3] Q. Zheng and R. Chellappa, "A Computational Vision Approach to Image Registration ," *IEEE Transactions Image Processing*, July 1993. pages 311-325.

[4] M. Irani and S. Peleg, "Improving Resolution by Image Registration," *CVGIP*, vol. 53, pp. 231–239, May 1991.

[5] L. Teodosio and W. Bender, "Salient video stills: Content and context preserved," *Proc. ACM Multimedia Conf.*, 1993.

[6] G. Wolberg, *Digital Image Warping*. 10662 Los Vaqueros Circle, Los Alamitos, CA: IEEE Computer Society Press, 1990. IEEE Computer Society Press Monograph.

[7] G. Adiv, "Determining 3D Motion and structure from optical flow generated by several moving objects," *IEEE Trans. Pattern Anal. Machine Intell.*, pp. 304–401, July 1985.

[8] N. Navab and S. Mann, "Recovery of relative affine structure using the motion flow field of a rigid planar patch.," *Mustererkennung 1994, Tagungsband.*, 1994.

[9] R. Y. Tsai and T. S. Huang, "Estimating three-dimensional motion parameters of a rigid planar patch," *tassp*, vol. ASSP-29, pp. 1147–1152, Dec. 1981.

[10] O. D. Faugeras and F. Lustman, "Motion and structure from motion in a piecewise planar environment," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 2, no. 3, pp. 485–508, 1988.

[11] A. Shashua and N. Navab, "Relative Affine: Theory and Application to 3D Reconstruction From Perspective Views.," *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, June 1994. 1994.

[12] H. Sawhney, "Simplifying motion and structure analysis using planar parallax and image warping," *CVPR*, 1994.

[13] R. Kumar, P. Anandan, and K. Hanna, "Shape recovery from multiple views: a parallax based approach," *ARPA image understanding workshop*, Nov 1984.

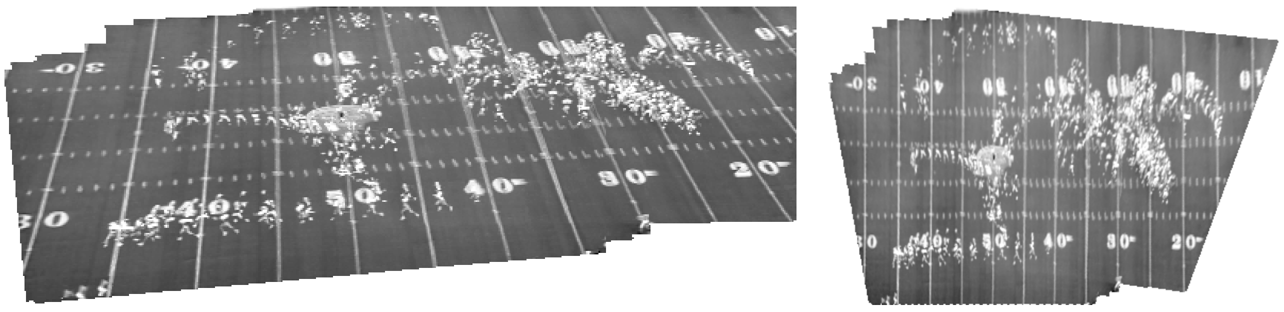
(a) (b)

Figure 11: Image mosaic made from 16 video frames taken from a television broadcast sporting event. Note the "Edgertonian" appearance, as each player traces out a stroboscopic-like path. The proposed method works robustly, despite the movement of players on the field. (a) Images are expressed in the coordinates of the first frame. (b) Images are expressed in a new useful coordinate system corresponding to none of the original frames. Note the slight distortion, due to the fact that football fields are not perfectly flat, but, rather, are raised slightly in the center.

[14] S. Mann, "Compositing multiple pictures of the same scene," in *Proceedings of the 46th Annual IS&T Conference*, (Cambridge, Massachusetts), The Society of Imaging Science and Technology, May 9-14 1993.

[15] R. Szeliski and J. Coughlan, "Hierarchical spline-based image registration," *CVPR*, 1994.

[16] L. Campbell and A. Bobick, "Correcting for radial lens distortion: A simple implementation," TR 322, M.I.T. Media Lab Perceptual Computing Section, Cambridge, Ma, Apr 1995.

[17] C. W. Wyckoff, "An experimental extended response film," *S.P.I.E. NEWSLETTER*, JUNE-JULY 1962.

[18] S. Mann and R. Picard, "Being 'undigital' with digital cameras: Extending dynamic range by combining differently exposed pictures," Tech. Rep. 323, M.I.T. Media Lab Perceptual Computing Section, Boston, Massachusetts, 1994. Also appears, IS&T's 46th annual conference, May 1995.

[19] M. Artin, *Algebra*. Prentice Hall, 1991.

[20] S. Mann, "Wavelets and chirplets: Time–frequency perspectives,
with applications," in *Advances in Machine Vision, Strategies and Applications* (P. Archibald, ed.), World Scientific, 1992.

[21] S. Mann and R. W. Picard, "Virtual bellows: constructing high-quality images from video," in *Proceedings of the IEEE first international conference on image processing*, (Austin, Texas), Nov. 13-16 1994.

[22] R. Y. Tsai and T. S. Huang, "Multiframe image restoration and registration," *ACM*, 1984.

[23] T. S. Huang and A. Netravali, "Motion and structure from feature correspondences: a review," *Proc. IEEE*, Feb 1984.

[24] N. Navab and A. Shashua, "Algebraic Description of Relative Affine Structure: Connections to Euclidean, Affine and Projective Structure.," *MIT Media Lab Memo No. 270*, 1994.

[25] H. L. Van Trees, *Detection, Estimation, and Modulation Theory (Part I)*. John Wiley and Sons, 1968.

[26] R. Young, "Wavelet theory and its applications," 1993.

[27] L. G. Weiss, "Wavelets and wideband correlation processing," *IEEE Signal Processing Magazine*, pp. 13–32, 1993.

[28] S. Mann and S. Haykin, "The chirplet transform — a generalization of Gabor's logon transform," *Vision Interface '91*, June 3-7 1991.

[29] S. Mann and S. Haykin, "Adaptive "Chirplet" Transform: an adaptive generalization of the wavelet transform," *Optical Engineering*, vol. 31, pp. 1243–1256, June 1992.

[30] B. Horn and B. Schunk, "Determining Optical Flow," *Artificial Intelligence*, 1981.

[31] J. Y. Wang and E. H. Adelson, "Spatio-Temopral Segmentation of Video Data ," in *SPIE Image and Video Processing II*, (San Jose, California), pp. 120–128, February 7-9 1994.

[32] J. Bergen, P. Burt, R. Hingorini, and S. Peleg, "Computing two motions from three frames," in *Proc. Third Int'l Conf. Comput. Vision*, (Osaka, Japan), pp. 27–32, December 1990.

[33] Lucas and T. Kanade, "An iterative image-registration technique with an application to stereo vision ," in *Image Understanding Workshop*, pp. 121–130, 1981.

[34] J. Y. A. Wang and E. H. Adelson, "Representing moving images with layers," *Image Processing Spec. Iss: Image Seq. Compression*, vol. 12, September 1994.

[35] R. Wilson and G. H. Granlund, "The Uncertainty Principle in Image Processing ," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, November 1984.

[36] J. Segman, J. Rubinstein, and Y. Y. Zeevi, "The canonical coordinates method for pattern deformation: Theoretical and computational considerations," *pami*, vol. 14, pp. 1171–1183, Dec. 1992.

[37] J. Segman, "Fourier cross correlation and invariance transformations for an optimal recognition of func-

tions deformed by affine groups," *josa*, vol. 9, pp. 895–902, June 1992.

[38] J. Segman and W. Schempp, *Two methods of incorporating scale in the Heisenberg group.* 1993. JMIV special issue on wavelets.

[39] B. Girod and D. Kuo, "Direct estimation of displacement histograms," *OSA Meeting on IMAGE UNDERSTANDING AND MACHINE VISION*, June 1989.

[40] Y. Sheng, C. Lejeune, and H. H. Arsenault, "Frequency-domain Fourier-Mellin descriptors for invariant pattern recognition," *Optical Engineering*, May 1988.

[41] P. J. Burt and P. Anandan, "Image stabilization by registration to a reference mosaic," *ARPA image understanding workshop*, Nov 1984.

[42] M. Hansen, P. Anandan, K. Dana, G. van der Wal, and P. Burt, "Real-time scene stabilization and mosaic construction," *ARPA image understanding workshop*, Nov 1984.

[43] S. Mann, "'See the world through my eyes,' a wearable wireless camera," 1995. http://www-white.media.mit.edu/˜steve/netcam.html.

[44] S. Mann, "Video orbits of the projective group," 1995. http://www-white.media.mit.edu/˜steve/orbits/orbits.html.

[45] S. Intille, "Computers watching football," 1995. http://www-white.media.mit.edu/vismod/demos/football/football.html.

Additional technical notes, examples, computer code, and sequences are available from our world wide web pages, http://www-white.media.mit.edu/vismod/vismod.html
and
http://www-white.media.mit.edu/˜steve/orbits/orbits.html
.