Real-Time Tracking of Image Regions
with Changes in Geometry and Illumination

Gregory D. Hager
Dept. of Computer Science
Yale University
New Haven, CT 06520-8258

Peter N. Belhumeur
Dept. of Electrical Engineering
Yale University
New Haven, CT 06520-8267

Abstract

Historically, SSD or correlation-based visual tracking algorithms have been sensitive to changes in illumination and shading across the target region. This paper describes methods for implementing SSD tracking that is both insensitive to illumination variations and computationally efficient. We first describe a vector-space formulation of the tracking problem, showing how to recover geometric deformations. We then show that the same vector space formulation can be used to account for changes in illumination. We combine geometry and illumination into an algorithm that tracks large image regions on live video sequences using no more computation than would be required to track with no accommodation for illumination changes. We present experimental results which compare the performance of SSD tracking with and without illumination compensation.

1   Introduction

Visual feature tracking has emerged as an important component of systems using vision as feedback for continuous control, surveillance, or visual reconstruction [1, 6, 8, 11, 13, 19]. The central issue in visual feature tracking is the temporal correspondence problem: the problem of matching regions of images taken at two closely-spaced time instants. A popular means for establishing temporal correspondence on unstructured images is to choose that which minimizes the sum of squared differences of image intensities, often referred to as SSD matching. SSD-based methods have been employed in a variety of contexts including stereo matching [12], optical flow computation [2], hand-eye coordination [13], and visual motion analysis [16].

In general, any tracking algorithm that operates in the "real world" has to deal with three major complications: local geometric distortions of the target, lighting changes and other photometric effects, and occlusion. SSD-based tracking methods tend to concentrate on handling geometry—in fact most implementations of SSD tracking only solve for a rigid translation of the target region. For inter-frame calculations such as those required for motion analysis, this is typically adequate, but generally not for applications where the correspondence for a finite size image patch over a long time span is needed. As noted by Shi and Tomasi [16], in such cases rotation, scaling, shear and other image distortions often have a significant effect on feature appearance and must be accounted for to achieve reliable matching. For example, Black and Yacoob [5] describe an algorithm for computing structured optical flow for recognizing facial expressions using motion models that include affine deformations and simple polynomial distortions. Rehg and Witkin [14] describe a similar algorithm for tracking arbitrarily deforming objects. In both cases, the algorithms track by integrating inter-frame changes, a procedure that is prone to cumulative error. More recent work considers tracking while the target undergoes changes of view by using a subspace of images and an iterative robust regression algorithm [4].

All of the SSD methods cited above assume that the changes in the target region are entirely due to geometry. Hence, changes in shading and illumination can easily influence the solutions for translation or object geometry, leading to estimation bias and, eventually, mistracking. Solutions to illumination problems have been largely confined to simple accommodations for brightness and contrast changes. In this paper, we address the problem of tracking objects under both

geometric and illumination changes. In particular, we show how general illumination models can be incorporated into SSD motion estimation with no extra online computational cost. We also exhibit a closed-loop formulation for the tracking problem and describe methods for accelerating the SSD computation so that it operates at, or near, frame rate.

2 The Method

Methods for implementing SSD matching are well-known and can be found in a variety of publications, e.g. [12, 2, 16, 18]. In this section, we first review the SSD method for estimating image translation, describe a generalization to arbitrary motion fields, and then introduce methods for handling changes in illumination.

2.1 Geometry

To begin we define our notation. Let $I(\mathbf{x}, t)$ be the image at time t. Let $I(\mathbf{x}, 0)$ be the *reference image*. Let $\mathbf{x} = (x, y)^T$ denote a point in an image. Let the set $\mathcal{R}$ be a grid of points in the reference image; we refer to $\mathcal{R}$ as the *target region* in the reference image. Let the center point of the target region have coordinates $\mathbf{x} = (0, 0)^T$.

If the target region only translates within the image, then for any time interval τ, there exists a $\mathbf{u} = (u, v)^T$ such that $I(\mathbf{x}, 0) = I(\mathbf{x} + \mathbf{u}, \tau)$ for all $\mathbf{x} \in \mathcal{R}$. Hence, the displacement of the target region can be determined by minimizing

$$O(\mathbf{u}) = \sum_{\mathbf{x} \in \mathcal{R}} (I(\mathbf{x} + \mathbf{u}, \tau) - I(\mathbf{x}, 0))^2. \quad (1)$$

Given that τ is small, a well-established method for solving for the optimum $\mathbf{u}$ is to linearize $I(\mathbf{x}, \tau)$ by expanding it in a Taylor series as follows

$$I(\mathbf{x} + \mathbf{u}, \tau) = I(\mathbf{x}, 0) + I_x u + I_y v + I_t \tau + h.o.t. \quad (2)$$

where $I_x = \partial I(\mathbf{x}, 0) / \partial x$, $I_y = \partial I(\mathbf{x}, 0) / \partial y$, $I_t = \partial I(\mathbf{x}, 0) / \partial t$, and $h.o.t.$ denotes the higher-order terms. Note that discrete approximations to I_x and I_y are computed from the reference image alone, while a discrete approximation to I_t is computed from both the reference image and the current image $I(\mathbf{x}, \tau)$.

Substituting into (1) and ignoring the higher-order terms, we get

$$O(\mathbf{u}) = \sum_{\mathbf{x} \in \mathcal{R}} (I_x u + I_y v + I_t \tau)^2. \quad (3)$$

Because (3) is quadratic function of u and v, we can find their optimum values by solving the pair of equa-

tions $\nabla O(\mathbf{u}) = \mathbf{0}$ to get

$$\begin{bmatrix} u \\ v \end{bmatrix} = -\tau \left(\sum_{\mathbf{x} \in \mathcal{R}} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \right)^{-1} \sum_{\mathbf{x} \in \mathcal{R}} \begin{bmatrix} I_x I_t \\ I_y I_t \end{bmatrix}. \quad (4)$$

SSD-based tracking algorithms repeatedly solve this linear system and then integrate the computed values of $\mathbf{u}$ to provide an estimate of the location of the target region.

It is interesting to consider what the solution to this system represents. The right-most vector in (4) has two components: the first given by the cross correlation of I_x with I_t and the second given by the cross correlation of I_y with I_t. To get the translation vector $\mathbf{u}$ we simply apply to the right-most vector a change of coordinates which depends only on the reference image, not on the current image $I(\mathbf{x}, \tau)$. Thus, we can think of I_x and I_y as "motion templates" which, through a cross correlation process with I_t, are used determine the translational motion.

The same observation applies to other types of motion. For example, suppose the target region undergoes a rotation in the image plane about its center point. We can compute this rotation θ by minimizing

$$O(\theta) = \sum_{\mathbf{x} \in \mathcal{R}} (I(R(\theta)\mathbf{x}, \tau) - I(\mathbf{x}, 0))^2 \quad (5)$$

where $R(\theta)$ is a 2×2 rotation matrix. As we did in (2), we can write down the Taylor series expansion of the current image $I(R(\theta)\mathbf{x}, \tau)$ as follows

$$I(R(\theta)\mathbf{x}, \tau) = I(\mathbf{x}, 0) + I_\theta \theta + I_t \tau + h.o.t. \quad (6)$$

where $I_\theta = \frac{\partial I(\mathbf{x}, 0)}{\partial \theta} = -I_x y + I_y x$. If the rotation is small, we can ignore the higher-order terms and substitute this back into (5) to get

$$O(\theta) = \sum_{\mathbf{x} \in \mathcal{R}} (I_\theta \theta + I_t \tau)^2. \quad (7)$$

We can find the optimum θ from the equation $\frac{dO(\theta)}{d\theta} = 0$ to get

$$\theta = -\tau \frac{\sum_{\mathbf{x} \in \mathcal{R}} I_\theta I_t}{\sum_{\mathbf{x} \in \mathcal{R}} I_\theta^2}. \quad (8)$$

It is again interesting to reflect on what the solution represents. The rotation θ is the cross correlation of I_θ with I_t, normalized by a number which again depends only on the reference image, not on the current image. We can think of I_θ as a "motion template" created by the dot product of $\nabla I = (I_x, I_y)^T$ with the motion field $\frac{dR(\theta)}{d\theta}\mathbf{x}$. As before this motion template
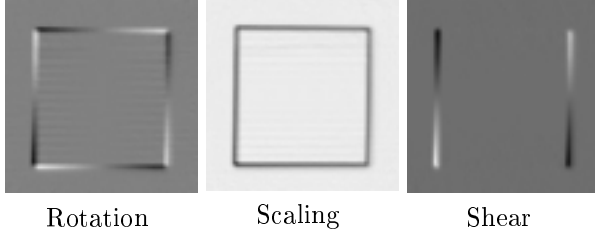
| Rotation | Scaling | Shear |

Figure 1: The motion template images for various motions of a black square on a white background.

I_θ, through a cross correlation process with I_t, is used determine the motion.

More generally, if we have a motion $\mathbf{f}(\mathbf{x}, \boldsymbol{\mu})$ parameterized by $\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots, \mu_n)^T$, with $\mathbf{f}(\mathbf{x}, \mathbf{0}) = \mathbf{x}$, we can compute the motion vector $\boldsymbol{\mu}$ by minimizing

$$O(\boldsymbol{\mu}) = \sum_{\mathbf{x} \in \mathcal{R}} (I(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}), \tau) - I(\mathbf{x}, 0))^2. \quad (9)$$

We can simplify the notation by defining the vector $\mathbf{I}(\boldsymbol{\mu}, t)$ as follows

$$\mathbf{I}(\boldsymbol{\mu}, t) = \begin{bmatrix} I(\mathbf{f}(\mathbf{x}_1, \boldsymbol{\mu}), t) \\ I(\mathbf{f}(\mathbf{x}_2, \boldsymbol{\mu}), t) \\ \ldots \\ I(\mathbf{f}(\mathbf{x}_n, \boldsymbol{\mu}), t) \end{bmatrix} \quad (10)$$

where $\mathbf{x}_i \in \mathcal{R}$. We can then rewrite (9) as

$$O(\boldsymbol{\mu}) = \|\mathbf{I}(\boldsymbol{\mu}, \tau) - \mathbf{I}(\mathbf{0}, 0)\|^2. \quad (11)$$

Let us further define $\mathbf{I}_{\boldsymbol{\mu}_i}(t)$ and $\mathbf{I_t}(t)$ as

$$\mathbf{I}_{\boldsymbol{\mu}_i}(t) = \begin{bmatrix} I_{\mu_i}(\mathbf{x}_1, t) \\ I_{\mu_i}(\mathbf{x}_2, t) \\ \vdots \\ I_{\mu_i}(\mathbf{x}_N, t) \end{bmatrix}, \; \mathbf{I_t}(t) = \begin{bmatrix} \tau I_t(\mathbf{x}_1, t) \\ \tau I_t(\mathbf{x}_2, t) \\ \vdots \\ \tau I_t(\mathbf{x}_N, t) \end{bmatrix} \quad (12)$$

where $I_{\mu_i}(\mathbf{x}, t) = \frac{\partial I(\mathbf{x}, t)}{\partial \mu_i}$ and $I_t(\mathbf{x}, t) = \frac{\partial I(\mathbf{x}, t)}{\partial t}$. We again call the $\mathbf{I}_{\boldsymbol{\mu}_i}(t)$ the "motion templates." (Figure 1 shows motion templates for various image motions for an image of a black square on a white background.) Let us also define the matrix

$$\mathbf{M}(t) = [\mathbf{I}_{\boldsymbol{\mu}_1}(t) | \mathbf{I}_{\boldsymbol{\mu}_2}(t) | \ldots | \mathbf{I}_{\boldsymbol{\mu}_n}(t)]. \quad (13)$$

Finally, for shorthand let $\mathbf{I_t} = \mathbf{I_t}(0)$ and $\mathbf{M} = \mathbf{M}(0)$.

Writing the current image in a Taylor series expansion, we get

$$\mathbf{I}(\boldsymbol{\mu}, \tau) = \mathbf{I}(\mathbf{0}, 0) + \mathbf{M}\boldsymbol{\mu} + \mathbf{I_t}\tau + h.o.t. \quad (14)$$

Solving (11) yields

$$\boldsymbol{\mu} = -(\mathbf{M}^T\mathbf{M})^{-1} \mathbf{M}^T \mathbf{I_t} \quad (15)$$

provided, of course, that the matrix $\mathbf{M}^T\mathbf{M}$ is invertible. When the matrix $\mathbf{M}^T\mathbf{M}$ is not invertible, then we are faced with a generalization of the aperture problem, i.e. the motion $\boldsymbol{\mu}$ can not be determined uniquely.

2.2 Illumination

The systems described above are inherently sensitive to changes in illumination. An illumination change in one part of the image produces temporal changes that would be interpreted by the machinery of the previous section as motion. This is not surprising, as we are effectively computing a structured optical flow, and optical flow methods are well-known to be sensitive to illumination changes [10]. Thus, shadowing or shading changes cause a bias, or, in the worst case, the complete loss of the target.

Recently, it has been shown that often a relatively small number of "basis" images can be used to account for large changes in illumination [15, 9]. Briefly, the reason for this is as follows. Consider a point p on a Lambertian surface and a collimated light source characterized by a vector $\mathbf{s} \in \mathbb{R}^3$, such that the direction of $\mathbf{s}$ gives the direction of the light rays and $\|\mathbf{s}\|$ gives the intensity of the light source. The irradiance at the point p is given by

$$E = a\,\mathbf{n} \cdot \mathbf{s} \quad (16)$$

where $\mathbf{n}$ is the unit inwards normal vector to the surface at p and a the non-negative absorption coefficient (albedo) of the surface at the point p [10]. This shows that the irradiance at the point p, and hence the gray level seen by a camera, is linear on $\mathbf{s} \in \mathbb{R}^3$.

Therefore, given three images (taken from the same viewpoint) of a Lambertian surface seen under three (known and linearly independent) light source directions, we can recover the albedo and surface normal; this is the well-known method of photometric stereo [20, 17]. Alternatively, one can reconstruct the image of the surface under a novel lighting direction by a linear combination of the three original images [15]. In other words, if the surface is purely Lambertian and there is no shadowing, then all images under varying illumination lie within a 3-D linear subspace of $\mathbb{R}^N$, the space of all possible images (where N is the number of pixels in the images).

A complication comes when handling shadowing: all images are no longer guaranteed to lie in a linear subspace [3]. Nevertheless, as done in [9], we can still use a linear model as an approximation. Naturally, we need more than three images and a higher than

three dimensional linear subspace if we hope to provide good approximation to these effects. However, a small set of basis images can account for much of the shading changes that occur on patches of non-specular surfaces.

Returning to the problem of SSD tracking, suppose now that we have a basis of image vectors $\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_m$ where the *ith* element of each of the basis vectors corresponds to the image location $\mathbf{x}_i \in \mathcal{R}$. Let us choose the first basis vector to be the target region in the reference image, i.e. $\mathbf{B}_1 = \mathbf{I}(\mathbf{0}, 0)$. To model brightness changes, let us choose the second basis vector to be a column of ones, i.e. $\mathbf{B}_2 = (1, 1, \ldots, 1)^T$. Let us choose the remaining basis vectors by performing SVD (singular value decomposition) on a set of training images of the target, taken under varying illumination. We denote the collection of basis vectors by the matrix $\mathbf{B} = [\mathbf{B}_1 | \mathbf{B}_2 | \ldots | \mathbf{B}_m]$. Finally, let us renormalize this matrix so that each of its columns are orthonormal.

If $\mathbf{I}(\mathbf{0}, 0)$ and $\mathbf{I}(\mathbf{0}, \tau)$ are registered geometrically, the remaining difference between them is due to illumination. Thus, the illumination of the target region in the current image $\mathbf{I}(\mathbf{0}, \tau)$ can be approximated by the target region in the reference image plus a linear comination of the basis vectors $\mathbf{B}$, i.e.

$$\mathbf{I}(\mathbf{0}, \tau) \simeq \mathbf{I}(\mathbf{0}, 0) + \mathbf{B}\boldsymbol{\lambda} \qquad (17)$$

where the vector $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_m)^T$. Note that because the target region in the reference image and an image of ones are included in the basis $\mathbf{B}$, we implicitly handle both variation due to contrast changes and variation due to brightness changes. The remaining basis vectors are used to handle subtler variation – variation that depends both on the geometry of the target object and on the nature of the light sources.

2.3 Combining Geometry and Illumination

Using the results of the previous two sections, we want to account for both illumination and geometry changes in one global optimization step, solved via linear methods. Our optimality criterion becomes

$$O(\boldsymbol{\mu}, \boldsymbol{\lambda}) = \|\mathbf{I}(\boldsymbol{\mu}, \tau) + \mathbf{B}\boldsymbol{\lambda} - \mathbf{I}(\mathbf{0}, 0)\|^2. \qquad (18)$$

We now approximate the current image $\mathbf{I}(\boldsymbol{\mu}, \tau)$ as

$$\mathbf{I}(\boldsymbol{\mu}, \tau) \simeq \mathbf{I}(\mathbf{0}, 0) + \mathbf{B}\boldsymbol{\lambda} + \mathbf{M}\boldsymbol{\mu}. \qquad (19)$$

Solving $\nabla O(\boldsymbol{\mu}, \boldsymbol{\lambda}) = \mathbf{0}$, yields

$$\begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{M}^T\mathbf{M} & \mathbf{M}^T\mathbf{B} \\ \mathbf{B}^T\mathbf{M} & \mathbf{B}^T\mathbf{B} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{M}^T \\ \mathbf{B}^T \end{bmatrix} \mathbf{I_t}. \qquad (20)$$

In general, we are only interested in the motion parameters. Reworking (20) we get

$$\boldsymbol{\mu} = -(\mathbf{M}^T(\mathbf{1} - \mathbf{B}\mathbf{B}^T)\mathbf{M})^{-1}\mathbf{M}^T(\mathbf{1} - \mathbf{B}\mathbf{B}^T)\,\mathbf{I_t}. \qquad (21)$$

If the columns of $\mathbf{B}$ are not orthonormal, then a similar but slightly longer expression for $\boldsymbol{\mu}$ results.

The above set of equations only involves the motion parameters $\boldsymbol{\mu}$. The dimension of the product of matrices multiplying $\mathbf{I_t}$ depends only on the number of motion fields to be computed. Hence, we have shown how to compute image motion while accounting for variations in illumination *using no more online computation than would be required to compute pure motion.*

2.4 Producing a Tracking System

The goal of visual tracking is to maintain a "constant fix" on a given target region. In our case, we define "constant fix" to mean that at any time $t > 0$, we have computed a globally optimal set of motion parameters $\boldsymbol{\mu}(t)$ such that

$$\boldsymbol{\mu}(t) = \arg\min_{\boldsymbol{\mu}} \|\mathbf{I}(\boldsymbol{\mu}, t) + \mathbf{B}\boldsymbol{\lambda} - \mathbf{I}(\mathbf{0}, 0)\|^2. \qquad (22)$$

The machinery of the previous two sections suggests that we can find the motion parameters at time $t + \tau$ by assuming we have the correct parameters at time t, i.e. $\mathbf{I}(\boldsymbol{\mu}(t), t) = \mathbf{I}(\mathbf{0}, 0)$. Using (21), we arrive at the discrete time closed-loop system

$$\boldsymbol{\mu}(t + \tau) = \boldsymbol{\mu}(t) + \boldsymbol{\Lambda}(t)\mathbf{I_t}(t) \qquad (23)$$

where

$$\mathbf{I_t}(t) = \mathbf{I}(\boldsymbol{\mu}(t), t + \tau) - \mathbf{I}(\mathbf{0}, 0)$$

and where

$$\boldsymbol{\Lambda}(t) = -(\mathbf{M}(t)^T(\mathbf{1} - \mathbf{B}\mathbf{B}^T)\mathbf{M}(t))^{-1}\mathbf{M}(t)^T(\mathbf{1} - \mathbf{B}\mathbf{B}^T).$$

It is interesting to note that the result is in the form of a proportional feedback system. $\mathbf{I_t}(t)$ is an error term which drives the system. The values in $\boldsymbol{\mu}$ form the state vector of the system and parameterize the change of coordinates $\mathbf{f}$ used to rectify the current image. This change of coordinates is typically implemented using *image warping*—hence, the the plant to be controlled is a computational warping operator. Given some knowledge of plant dynamics (i.e. the motion of the patch), more sophisticated feedback system using PID methods or more sophisticated optimal control methods could be used. In particular, a smoother/predictor such as a Kalman filter could be

used to add feedforward to the system based on prior knowledge of target dynamics.

This general solution to the tracking problem is slow to execute as it requires explicit linearization of the system at every time point, which in turn involves calculation of the motion templates and solving a large linear system at each step. This can be simplified if $\mathbf{M}(t)$ is constant. In particular, suppose that $\mathbf{f}$ is linear in $\boldsymbol{\mu}$. Then the Jacobian of $\mathbf{f}$ does not involve $\boldsymbol{\mu}$, the motion templates are constant, and $\boldsymbol{\Lambda}(t)$ is constant and can be computed offline. Examples include affine transformations and polynomials in $\mathbf{x}$. It is also possible to show that a similar result holds for rigid body transformations such as rotations, even though they are not linear in their parameters.

To summarize an efficient tracking algorithm for deformations that are linear in their parameters consists of the following steps: **offline,** acquire images to compute an illumination basis, acquire the target region in the reference image, compute the motion templates, and compute $\boldsymbol{\Lambda}$; **online,** use the current motion parameter estimates to acquire and warp the target region in the current image, compute the difference between the warped target region and the target region in the reference image, and finally multiply this value by the solution matrix computed offline, adding the result to the current parameter estimate.

3   Experimental Results

We have implemented the methods described above within the X Vision environment [7]. The implemented system accommodates affine distortions and non-orthonormal illumination bases. Timings of the system indicate that it can perform frame rate (30 Hz) tracking of image regions of up to 100×100 pixels at half resolution undergoing affine distortion on a 120Mhz Pentium processor or SGI Indy. Higher performance is achieved for smaller regions, lower resolutions, or fewer parameters. For example, tracking the same region by computing just translation at one-fourth resolution take 4 milliseconds.

In this section, we present experiments with the system focusing on the effects of warping and illumination. All experiments in this section were performed on live video sequences by an SGI Indy equipped with a 175Mhz R4400 SC processor and VINO image acquisition system[1].

The target we track is a human face. We have found that at normal viewing distances and viewing angles,

---

[1] Because of additional data collection overhead, the tracking is actually much slower than the figures stated above.

affine warping performs well at accounting for the geometry changes of a face, but at the same time the face is non-planar, hence exhibits interesting illumination effects. To accommodate illumination changes in a face, we use an illumination basis of five orthogonal image vectors. This basis was computed offline by acquiring several images of the face under various lighting conditions. A singular value decomposition (SVD) was applied to the resulting image vectors and the vectors with the maximum singular values were chosen to be included in the basis. The basis is shown in Figure 2.

Geometry To get a sense of the accuracy with which geometric changes can be computed, we first test the tracking algorithm in a sequence when there are no large changes in the illumination of the target. In this test, we simultaneously executed an SSD algorithm which incorporated an illumination basis, and one which did not. Under these circumstances, we would expect both algorithms to compute solutions that are extremely close to one another unless the illumination basis significantly affects the sensitivity of the computation. Figure 3 shows excerpts from the run, and also shows the residuals of the linearized SSD computation at each time step. The black frames denote the region selected as the best match by the algorithm using no illumination basis, and the white frames correspond to the best match computed by the algorithm incorporated illumination. As is apparent from the figures, the residuals and computed state variables of both algorithms are extremely similar for the entire run — so close that in many cases one frame is obscured by the other.

The middle row of images shows the region contained within the frame after warping by the estimated motion parameters. If the parameters were ideal, each image would be identical. Despite the fact that the face is non-planar, resulting for example in a stretching of the nose as the face is turned, the warping is quite effective.

Illumination Because it deviates from a planar surface, a face can exhibit strong shading and illumination effects. However, a small number of basis images of a face gathered under different illuminations is sufficient to accurately account for most gross shading and illumination effects [9]. In a second set of experiments, we used a fixed geometry and varied the illumination of the face. Figure 4 shows the effects of

Figure 2: The illumination basis for the face.

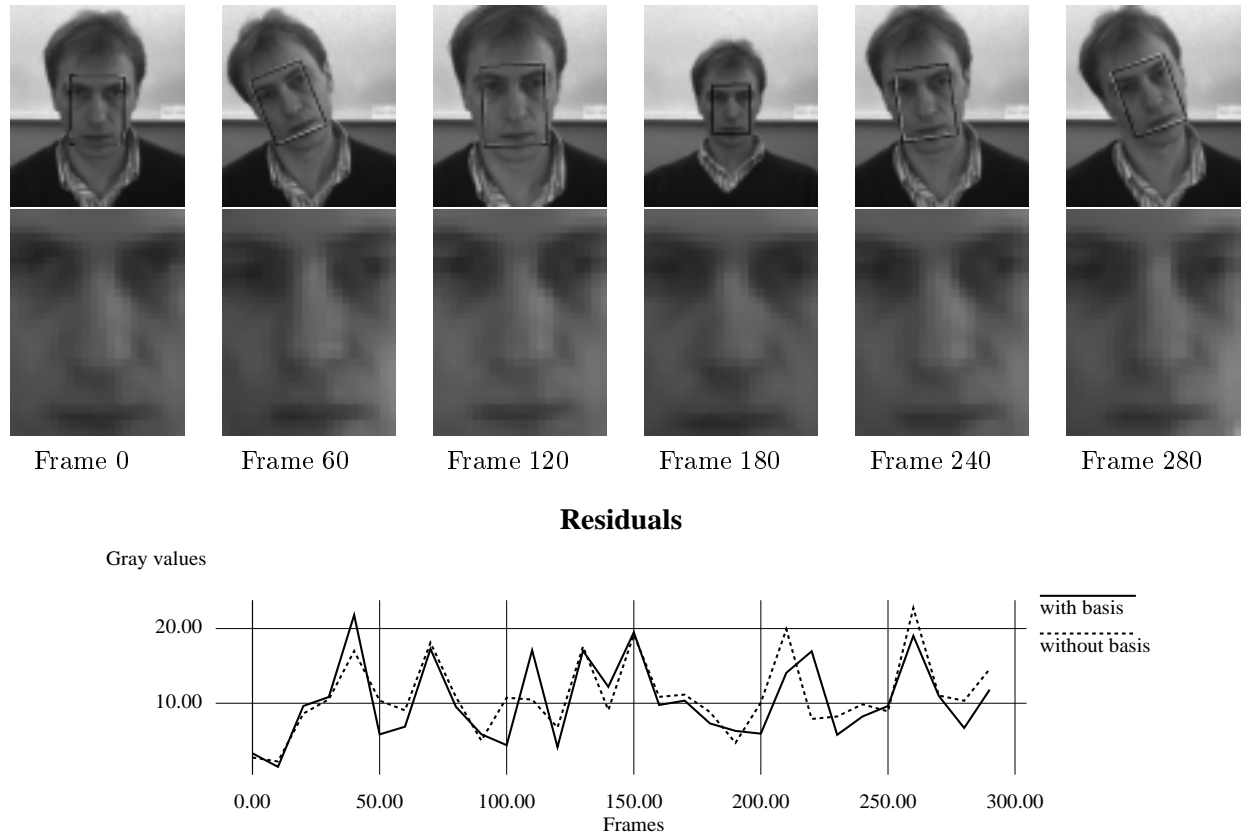| Frame 0 | Frame 60 | Frame 120 | Frame 180 | Frame 240 | Frame 280 |

Residuals

Figure 3: Above, excerpts from a sequence of tracked images. The white frames represent the region tracked by the SSD algorithm using an illumination basis, and black frames represent the regions tracked by an algorithm which does not. In some cases the estimates are so close that only one box is visible. Middle row, the region within the frame warped by the current motion estimate. Below, the residuals, in gray-scale units per pixel of the algorithms as a function of time.
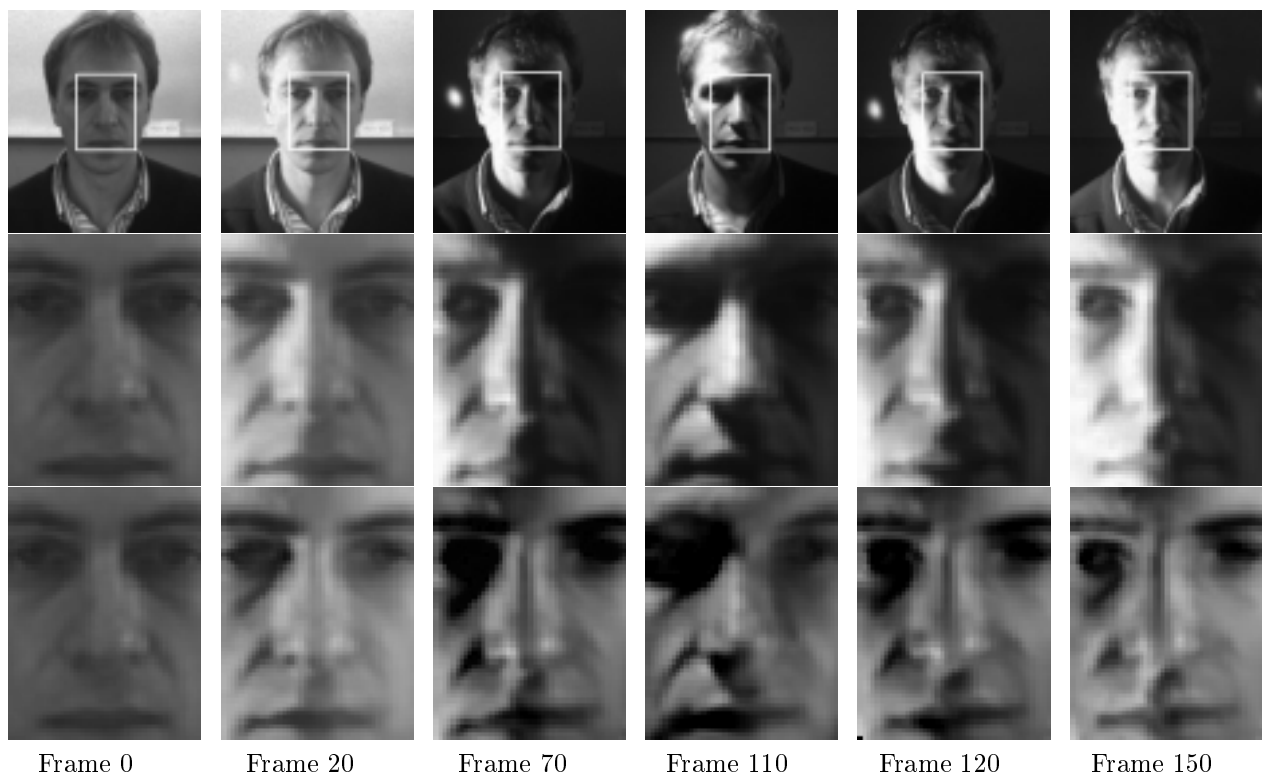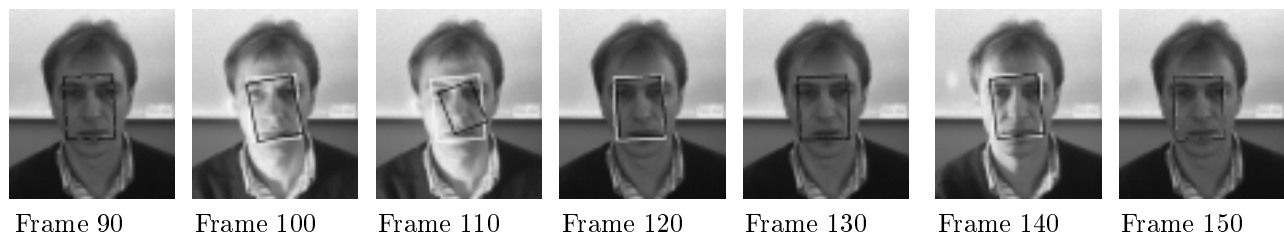
Figure 4: The first row of images shows excerpts of a tracking sequence. The second row is a magnified view of the region in the white box, and the third row are the images of the second row after adjustment for illumination.
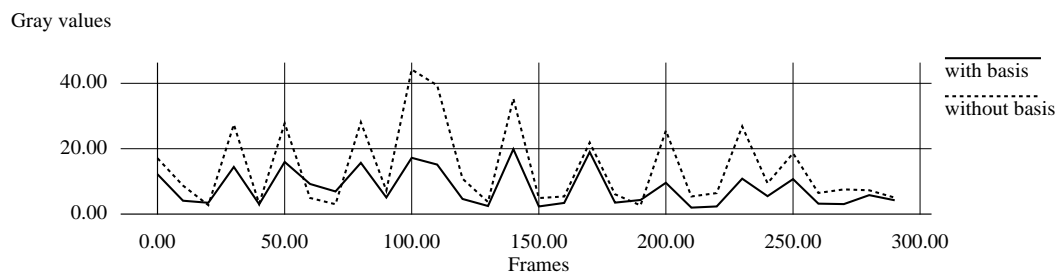

Figure 5: Tracking with illumination changes. Above, an excerpt near the middle of a tracking sequence. Note that the algorithm which does not use illumination (the black frames) completely looses the target until the original lighting is restored. Below, the residuals, in scale units per pixel, of SSD tracking with and without an illumination basis as a light is turned on and off.

illumination compensation for the illumination situations depicted in the first row. As with warping, if the compensation were perfect, the images of the bottom row would appear to be identical up to brightness and contrast. In particular, note how the strong shading effects of frames 70 through 150 have been "erased" by the illumination basis.

Combining Illumination and Geometry Finally, we present a set of experiments illustrating the interaction of geometry and illumination. In these experiments, translation, scale and orientation were computed, again using two algorithms—one which incorporated an illumination basis and one which did not. As the algorithms were tracking, a light was periodically switched on and off. The results appear in Figure 5. In the residual graph, we see that the illumination basis clearly "accounts" for the shading on the face quite well, leading to a much lower fluctuation of the residuals. The sequence of images shows an excerpt near the middle of the sequence where the algorithm which could not compensate for illumination completely lost the target for several frames, only regaining it after the original lighting was restored. Since the target was effectively motionless during this period, this can be completely attributed to biases due to illumination effects. Similar sequences with larger target motions often cause the purely geometric algorithm to loose the target completely.

4   Discussion

We have shown a straightforward solution to the problem of tracking regions undergoing geometric distortion and changing illumination. The method is simple and straightforward, and builds on an already popular method for performing visual feature tracking.

We are actively continuing to evaluate the performance of these methods, and to extend their theoretical underpinnings. One area that still needs attention is the question of determining an illumination basis. In the work described above, we used a basis computed offline. In order to make the method applicable to on-line real-time applications, we need to either develop the basis online, or develop a "generic" basis that handles most dominant shading effects. As in [4], we are also exploring is the use of using basis images to handle changes of view or aspect not well addressed by warping.

One problem that we did not address in this paper is occlusion. In particular, the methods described here can be quite sensitive to partial occlusion. For example, a small light patch entering the image strongly affects the image normalization steps and can lead to poor tracking performance or mistracking. Recent experiments using robust regression methods suggest that these minor intrusions can be effectively detected and ignored.

References

[1] P.K. Allen, B. Yoshimi, and A. Timcenko. Hand-eye coordination for robotics tracking and grasping. In K. Hashimoto, editor, *Visual Servoing*, pages 33–70. World Scientific, 1994.

[2] P. Anandan. A computational framework and an algorithm for the measurement of structure from motion. *Int. J. Computer Vision*, 2:283–310, 1989.

[3] P. N. Belhumeur and D. J. Kriegman. What is the set of images of and object under all possible lighting conditions? In *IEEE Conf. Computer Vision and Pattern Recognition*, 1996.

[4] M.J. Black and A.D. Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. In *Proc. European Conf. on Computer Vision*, 1996.

[5] M.J. Black and Y. Yacoob. Tracking and recognizing rigid and non-rigid facial motions using local parametric models of image motion. In *Proceedings of the ICCV*, pages 374–381, 1995.

[6] E.D. Dickmanns and V. Graefe. Dynamic monocular machine vision. *Machine Vision and Applications*, 1:223–240, 1988.

[7] G. D. Hager. The "X-vision" system: A general purpose substrate for real-time vision-based robotics. In *Proceedings of the Workshop on Vision for Robots*, pages 56–63, 1995. Also available as Yale CS-RR-1078.

[8] G. D. Hager, W-C. Chang, and A. S. Morse. Robot hand-eye coordination based on stereo vision. *IEEE Control Systems Magazine*, 15(1):30–39, February 1995.

[9] Peter Hallinan. A low-dimensional representation of human faces for arbitrary lighting conditions. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, pages 995–999, 1994.

[10] B.K.P. Horn. *Computer Vision*. MIT Press, Cambridge, Mass., 1986.

[11] E. Huber and D. Kortenkamp. Using stereo vision to pursue moving agents with a mobile robot. In *Proc. 1995 IEEE Conf. on Rob. and Autom.*, pages 2340–2346, Nagoya, Japan, May 1995.

[12] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. Int. Joint Conf. Artificial Intelligence*, pages 674–679, 1981.

[13] N. Papanikolopoulos, P. Khosla, and T. Kanade. Visual tracking of a moving target by a camera mounted on a robot: A combination of control and vision. *IEEE Trans. on Robotics and Automation*, 9(1), 1993.

[14] J.M. Rehg and A.P. Witkin. Visual tracking with deformation models. In *Proc. IEEE Int. Conf. Robot. and Automat.*, pages 844–850, 1991.

[15] Amnon Shashua. *Geometry and Photometry in 3D Visual Recognition*. PhD thesis, MIT, 1992.

[16] J. Shi and C. Tomasi. Good features to track. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 593–600. IEEE Computer Society Press, 1994.

[17] W.M. Silver. *Determining Shape and Reflectance Using Multiple Images*. PhD thesis, MIT, Cambridge, MA, 1980.

[18] C.E. Smith, S.A. Brandt, and N.P. Papnikolopoulos. Controlled active exploration of uncalibrated environments. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 792–795. IEEE Computer Society Press, 1994.

[19] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *Int. J. Computer Vision*, 9(2):137–154, 1992.

[20] R.J. Woodham. Analysing images of curved surfaces. *Artificial Intelligence*, 17:117–140, 1981.