Submitted to *IEEE Trans. on PAMI*

Efficient Region Tracking With Parametric Models of Geometry and Illumination

Gregory D. Hager
Department of Computer Science
Yale Univ., P.O. Box 208285
New Haven, CT, 06520

Phone: (203) 432-6432,   Fax: (203) 432-0593
E-mail: hager@cs.yale.edu

Peter N. Belhumeur
Department of Electrical Engineering
Yale Univ., P.O. Box 208267
New Haven, CT, 06520

Phone: (203) 432-4249,   Fax: (203) 432-7481
E-mail: belhumeur@yale.edu

Abstract

As an object moves through the field of view of a camera, the images of the object may change dramatically. This is not simply due to the translation of the object across the image plane. Rather, complications arise due to the fact that the object undergoes changes in pose relative to viewing camera, changes in illumination relative to light sources, and may even be partially or fully occluded. Thus to successfully track an object, complications arising from varying pose, illumination, and partial occlusion must be accounted for. In this paper, we develop an efficient, general framework for object tracking – one which addresses each of these complications. We first develop a computationally efficient method for handling the geometric distortions produced by changes in pose. We then combine geometry and illumination into an algorithm that tracks large image regions using no more computation than would be required to track with no accommodation for illumination changes. Finally, we augment these methods with techniques from robust statistics and treat occluded regions on the object as statistical outliers. Throughout, we present experimental results performed on live video sequences demonstrating the effectiveness and efficiency of our methods.

Keywords: Visual Tracking, Illumination, Motion Estimation, Robust Statistics.

1 Introduction

Visual tracking has emerged as an important component of systems in several application areas including vision-based control [1, 32, 38, 15], human-computer interfaces [10, 14, 20], surveillance [30, 29, 19], agricultural automation [27, 41], medical imaging [12, 4, 45] and visual reconstruction [11, 42, 48]. The central challenge in visual tracking is to determine the image position of a target region (or features) of an object as it moves through a camera's field of view. This is done by solving what is known as the temporal correspondence problem: the problem of matching the target region in successive frames of a sequence of images taken at closely-spaced time intervals. The correspondence problem for visual tracking has, of course, much in common with the correspondence problems which arise in stereopsis and optical flow. It differs, however, in that the goal is not to determine the exact correspondence for every image location in a pair of images, but rather to determine, in a gross sense, the movement of an entire target region over a long sequence of images. What makes tracking difficult is the extreme variability often present in the images of an object over time. This variability arises from three principle sources: variation in object pose, variation in illumination, and partial or full occlusion of the target. When ignored, any one of these three sources of variability is enough to cause a tracking algorithm to lose its target.

In this paper, we develop a general framework for region tracking which includes models for image changes due to motion, illumination, and partial occlusion. In the case of motion, all points in the target region are presumed to be part of the same object allowing us the luxury – at least for most applications – of assuming that these points move coherently in space. This permits us to develop low-order parametric models for the image motion of points within a target region—models that can be used to predict the movement of the points and track the target through an image sequence. In the case of illumination, we exploit the observations of [25, 17, 5] to model image variation due to changing illumination by low-dimensional linear subspaces. The motion and illumination models are then woven together in an efficient algorithm which establishes temporal correspondence of the target region by simultaneously determining motion and illumination parameters. These parameters not only shift and deform image coordinates, but also adjust brightness values within the target region to provide the best match to a fixed reference image. Finally, in the case of partial occlusion, we apply results from robust statistics [16] to show that this matching algorithm is easily extended to include automatic rejection of outlier pixels in a computationally efficient manner.

The approach to matching described in this paper is based on comparing the so-called sum-of-squared differences (SSD) between two regions, an idea that has been explored in a variety of contexts including stereo matching [35], optical flow computation [2], hand-eye coordination [38], and visual motion analysis [44]. Much of the previous work using SSD matching for tracking has modeled the motion of the target region as pure translation in the image plane [48, 38], which implicitly assumes that the underlying object is translating parallel to the image plane and is being viewed orthographically. For inter-frame calculations such as those required for optical flow or motion analysis, pure translation is typically adequate. However, for tracking applications in which the correspondence for a finite size image patch must be computed over a long time span, the pure translation assumption is soon violated [44]. In such cases, both geometric image distortions such as rotation, scaling, shear, and illumination changes introduce significant changes in the appearance of the target region and, hence, must be accounted for in order to achieve reliable matching.

Attempts have been made to include more elaborate models for image change in region tracking algorithms, but with sizable increases in the computational effort required to establish correspondence. For example, Rehg and Witkin [40] describe energy-based algorithms for tracking deforming image regions, and Rehg and Kanade [39] consider articulated objects undergoing self-occlusion. More recently, Black and Yacoob [8] describe an algorithm for recognizing facial expressions using motion models which include both affine and simple polynomial deformations of the face and its features. Black and Jepson [7] develop a robust algorithm for tracking a target undergoing changes in pose or appearance by combining a simple parametric motion model with an image subspace method [37]. These algorithms require from several seconds to several minutes per frame to compute, and most do not address the problems of changes in appearance due to illumination.

In contrast, we develop a mathematical framework for the region tracking problem that naturally incorporates models for geometric distortions and varying illumination. Using this framework, we show that the computations needed to perform temporal matching can be factored to greatly improve algorithm efficiency. The result is a family of region-tracking algorithms which can easily track large image regions (for example the face of a user at a workstation, at a 30 Hz frame rate) using no special hardware other than a standard digitizer.

To date, most tracking algorithms achieving frame-rate performance track only a sparse collection of features (or contours). For example, Blake et al. [9] and Isaard and Blake [33] describe a variety of novel methods for incorporating both spatial and temporal constraints on feature evolution for snake-like contour tracking. Lowe [34] and Gennery [21] describe edge-based tracking methods using rigid three-dimensional geometric models. Earlier work by Ayache [3] and Crowley [13] use incrementally constructed rigid models to constrain image matching.

In practice, feature-based and region-based methods can be viewed as complementary techniques. In edge-rich environments such as a manufacturing floor, working with sparse features such as edges has the advantage of computational simplicity — only a small area of the image contributes to the tracking process, and the operations performed in that region are usually very simple. Furthermore, edge-based methods use local derivatives and, hence, tend to be insensitive to global changes in the intensity and/or composition of the incident illumination. However, in less structured situations strong edges are often sparsely distributed in an image, and are difficult to detect and match robustly without a strong predictive model [33]. In such cases, the fact that region-based methods make direct and complete use of all available image intensity information eliminates the need to identify and model a special set of features to track. By incorporating illumination models and robust estimation methods and by making the correspondence algorithm efficient, the robustness and performance of our region tracking algorithms closely rivals that achieved by edge-based methods.

The remainder of this article is organized as follows. Section 2 establishes a framework for posing the problem of region tracking for parametric motion models and describes conditions under which an efficient tracking algorithm can be developed. Section 3 then shows how models of illumination can be incorporated with no loss of computational efficiency. Section 4 details modifications for handling partial target occlusion via robust estimation techniques. Section 5 presents experimental results from an implementation of the algorithms. Finally, Section 6 presents a short discussion of performance improving extensions to our tracking algorithm.

2 Tracking Moving Objects

In this section, we describe a framework for the efficient tracking of a target region through an image sequence. We first write down a general parametric model for the set of allowable image motions and deformations of the target region. We then pose the tracking problem as the problem of finding the best (in a least squares sense) set of parameter values describing the motions and deformations of the target through the sequence. Finally, we describe how the best set of parameters can be efficiently computed.

2.1 On Recovering Structured Motion

We first consider the problem of describing the motion of a target region of an object through a sequence of images. Points on the surface of the object, including those in the target region, are projected down into the image plane. As the object moves through space, the projected points move in the image plane. If the 3-D structure of the object is known in advance, then we could *exactly* determine the set of possible motions of the points in the images. In general, this information is not known in advance. Therefore, we approximate the set of possible motions by a parametric model for image motions.

Let $I(\mathbf{x}, t)$ denote the brightness value at the location $\mathbf{x} = (x, y)^T$ in an image acquired at time t and let $\nabla_{\mathbf{x}} I(\mathbf{x}, t)$ denote the spatial gradient at that location and time. The symbol t_0 denotes an identified "initial" time and we refer to the image at time t_0 as the *reference image.* Let the set $\mathcal{R} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ be a set of N image locations which define a *target region.* We refer to the brightness values of the target region in the reference image as the *reference template.*

Over time, the relative motion between the target object and the camera causes the image of the target to shift and to deform. Let us model the image motion of the target region of the object by a parametric *motion model* $\mathbf{f}(\mathbf{x}; \boldsymbol{\mu})$ parameterized by $\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots, \mu_n)^T$, with $\mathbf{f}(\mathbf{x}; \mathbf{0}) = \mathbf{x}$ and $N > n$. We assume that $\mathbf{f}$ is differentiable in both $\boldsymbol{\mu}$ and $\mathbf{x}$. We call $\boldsymbol{\mu}$ the *motion parameter vector.* We consider recovering the motion parameter vector for each image in the tracking sequence as the equivalent to "tracking the object." We write $\boldsymbol{\mu}^*(t)$ to denote the ground truth values of these parameters at time t, and $\boldsymbol{\mu}(t)$ to denote the corresponding estimate. The argument t will be suppressed when it is obvious from its context.

Suppose that a reference template is acquired at time t_0 and that initially $\boldsymbol{\mu}^*(t_0) = \boldsymbol{\mu}(t_0) = \mathbf{0}$. Let us assume for now that the only changes in subsequent images of the target are completely described by $\mathbf{f}$, i.e. there are no changes in the illumination of the target. It follows that for any time $t > t_0$ there is a parameter vector $\boldsymbol{\mu}^*(t)$ such that

$$I(\mathbf{x}, t_0) = I(\mathbf{f}(\mathbf{x}; \boldsymbol{\mu}^*(t)), t) \text{ for all } \mathbf{x} \in \mathcal{R}. \tag{1}$$

This a generalization of the so-called *image constancy assumption* [28]. The motion parameter vector of the target region can be estimated at time t by minimizing the following least squares objective function

$$O(\boldsymbol{\mu}) = \sum_{\mathbf{x} \in \mathcal{R}} (I(\mathbf{f}(\mathbf{x}; \boldsymbol{\mu}), t) - I(\mathbf{x}, t_0))^2. \tag{2}$$

For later developments, it is convenient to rewrite this optimization problem in vector notation. To this end, let us consider images of the target region as vectors in an N dimensional space. The image of the target region at time t, under the change of coordinates $\mathbf{f}$ with parameters $\boldsymbol{\mu}$, is written as

$$\mathbf{I}(\boldsymbol{\mu}, t) = \begin{bmatrix} I(\mathbf{f}(\mathbf{x}_1, \boldsymbol{\mu}), t) \\ I(\mathbf{f}(\mathbf{x}_2, \boldsymbol{\mu}), t) \\ \cdots \\ I(\mathbf{f}(\mathbf{x}_N, \boldsymbol{\mu}), t) \end{bmatrix}. \tag{3}$$

This vector is subsequently referred to as the *rectified image* at time t with parameters $\boldsymbol{\mu}$. We also make use of the partial derivatives of $\mathbf{I}$ with respect to the components of $\boldsymbol{\mu}$ and the time parameter t. These are written as

$$\mathbf{I}_{\mu_i}(\boldsymbol{\mu}, t) = \frac{\partial \mathbf{I}}{\partial \mu_i} = \begin{bmatrix} I_{\mu_i}(\mathbf{f}(\mathbf{x}_1, \boldsymbol{\mu}), t) \\ I_{\mu_i}(\mathbf{f}(\mathbf{x}_2, \boldsymbol{\mu}), t) \\ \vdots \\ I_{\mu_i}(\mathbf{f}(\mathbf{x}_N, \boldsymbol{\mu}), t) \end{bmatrix} \tag{4}$$

and

$$\mathbf{I}_t(\boldsymbol{\mu}, t) = \frac{\partial \mathbf{I}}{\partial t} = \begin{bmatrix} I_t(\mathbf{f}(\mathbf{x}_1, \boldsymbol{\mu}), t) \\ I_t(\mathbf{f}(\mathbf{x}_2, \boldsymbol{\mu}), t) \\ \vdots \\ I_t(\mathbf{f}(\mathbf{x}_N, \boldsymbol{\mu}), t) \end{bmatrix} \tag{5}$$

where $1 \leq i \leq n$.

Using this vector notation, the image constancy assumption (1) can be rewritten as

$$\mathbf{I}(\boldsymbol{\mu}^*(t), t) = \mathbf{I}(\mathbf{0}, t_0)$$

and (2) becomes

$$O(\boldsymbol{\mu}) = \|\mathbf{I}(\boldsymbol{\mu}, t) - \mathbf{I}(\mathbf{0}, t_0)\|^2. \tag{6}$$

In general, (6) is a non-convex objective function. Thus, in the absence of a good starting point, this problem will usually require some type of costly global optimization procedure to solve [6].

In the case of visual tracking, the continuity of motion provides such a starting point. Suppose that, at some arbitrary time $t > t_0$, the geometry of the target region is described by $\boldsymbol{\mu}(t)$. We recast the tracking problem as one of determining a vector of offsets, $\delta\boldsymbol{\mu}$ such that $\boldsymbol{\mu}(t+\tau) = \boldsymbol{\mu}(t) + \delta\boldsymbol{\mu}$ from an image acquired at $t + \tau$. Incorporating this modification into (6), we redefine the objective function as a function on $\delta\boldsymbol{\mu}$

$$O(\delta\boldsymbol{\mu}) = \|\mathbf{I}(\boldsymbol{\mu}(t) + \delta\boldsymbol{\mu}, t + \tau) - \mathbf{I}(\mathbf{0}, t_0)\|^2. \tag{7}$$

If the magnitude of the components of $\delta\boldsymbol{\mu}$ are small, then it is possible to apply continuous optimization procedures to a linearized version of the problem [7, 28, 35, 47, 44]. The linearization is carried out by expanding $\mathbf{I}(\boldsymbol{\mu} + \delta\boldsymbol{\mu}, t + \tau)$ in a Taylor series about $\boldsymbol{\mu}$ and t,

$$\mathbf{I}(\boldsymbol{\mu} + \delta\boldsymbol{\mu}, t + \tau) = \mathbf{I}(\boldsymbol{\mu}, t) + \mathbf{M}(\boldsymbol{\mu}, t)\ \delta\boldsymbol{\mu} + \tau\mathbf{I}_t(\boldsymbol{\mu}, t) + h.o.t, \tag{8}$$

where $h.o.t$ denotes higher order terms of the expansion, and $\mathbf{M}$ is the *Jacobian matrix* of $\mathbf{I}$ with respect to $\boldsymbol{\mu}$, i.e. the $N \times n$ matrix of partial derivatives which can be written in column form as

$$\mathbf{M}(\boldsymbol{\mu}, t) = [\mathbf{I}_{\mu_1}(\boldsymbol{\mu}, t) | \mathbf{I}_{\mu_2}(\boldsymbol{\mu}, t) | \ldots | \mathbf{I}_{\mu_n}(\boldsymbol{\mu}, t)]. \tag{9}$$

While the notation above explicitly indicates that the values of the partial derivatives are a function of the evaluation point $(\boldsymbol{\mu}, t)$, these arguments will be suppressed when obvious from their context.

By substituting (8) into (7) and ignoring the higher order terms, we have

$$O(\delta\boldsymbol{\mu}) \approx \|\mathbf{I}(\boldsymbol{\mu}, t) + \mathbf{M}\ \delta\boldsymbol{\mu} + \tau\mathbf{I}_t - \mathbf{I}(\mathbf{0}, t_0)\|^2. \tag{10}$$

With the additional approximation $\tau \mathbf{I}_t(\boldsymbol{\mu}, t) \approx \mathbf{I}(\boldsymbol{\mu}, t+\tau) - \mathbf{I}(\boldsymbol{\mu}, t)$, (10) becomes

$$O(\delta\boldsymbol{\mu}) \approx \|\mathbf{M}\ \delta\boldsymbol{\mu} + \mathbf{I}(\boldsymbol{\mu}, t+\tau) - \mathbf{I}(\mathbf{0}, t_0)\|^2. \tag{11}$$

Solving the set of equations $\nabla O = \mathbf{0}$ yields the solution

$$\delta\boldsymbol{\mu} = -(\mathbf{M}^T\mathbf{M})^{-1}\ \mathbf{M}^T\ \left[\mathbf{I}(\boldsymbol{\mu}, t+\tau) - \mathbf{I}(\mathbf{0}, t_0)\right], \tag{12}$$

provided the matrix $\mathbf{M}$ is full rank. When this is not the case, we are faced with a generalization of the aperture problem, i.e. the target region does not have sufficient structure to determine all of the elements of $\boldsymbol{\mu}$ uniquely.

In subsequent developments, it will be convenient to define the *error vector* $\mathbf{e}(t+\tau) = \mathbf{I}(\boldsymbol{\mu}(t), t+\tau) - \mathbf{I}(\mathbf{0}, t_0)$. Incorporating this definition into (12), we see that the solution of (6) at time $t+\tau$ given a solution at time t is

$$\boldsymbol{\mu}(t+\tau) = \boldsymbol{\mu}(t) - (\mathbf{M}^T\mathbf{M})^{-1}\ \mathbf{M}^T\ \mathbf{e}(t+\tau) \tag{13}$$

where $\mathbf{M}$ is evaluated at $(\boldsymbol{\mu}, t)$.

2.2 An Efficient Tracking Algorithm

From (13), we see that to track the target region through the image sequence, we must compute the Jacobian matrix $\mathbf{M}(\boldsymbol{\mu}, t)$. Each element of this matrix is given by

$$\begin{aligned} m_{ij} &= I_{\mu_j}(\mathbf{f}(\mathbf{x}_i; \boldsymbol{\mu}), t) \\ &= \nabla_{\mathbf{f}} I(\mathbf{f}(\mathbf{x}_i; \boldsymbol{\mu}), t)^T \mathbf{f}_{\mu_j}(\mathbf{x}_i; \boldsymbol{\mu}), \end{aligned} \tag{14}$$

where $\nabla_{\mathbf{f}} I$ is the gradient of I with respect to the components of the vector $\mathbf{f}$. Recall that the Jacobian matrix of the transformation $\mathbf{f}$ regarded as a function of $\boldsymbol{\mu}$ is the $2 \times n$ matrix

$$\mathbf{f}_{\boldsymbol{\mu}}(\mathbf{x}; \boldsymbol{\mu}) = \left[\frac{\partial \mathbf{f}(\mathbf{x}; \boldsymbol{\mu})}{\partial \mu_1} \,\middle|\, \frac{\partial \mathbf{f}(\mathbf{x}; \boldsymbol{\mu})}{\partial \mu_2} \,\middle|\, \dots \,\middle|\, \frac{\partial \mathbf{f}(\mathbf{x}; \boldsymbol{\mu})}{\partial \mu_n} \right]. \tag{15}$$

By making use of (15), $\mathbf{M}$ can be written compactly in row form as

$$\mathbf{M}(\boldsymbol{\mu}, t) = \begin{bmatrix} \nabla_{\mathbf{f}} I(\mathbf{f}(\mathbf{x}_1; \boldsymbol{\mu}), t)^T\ \mathbf{f}_{\boldsymbol{\mu}}(\mathbf{x}_1; \boldsymbol{\mu}) \\ \nabla_{\mathbf{f}} I(\mathbf{f}(\mathbf{x}_2; \boldsymbol{\mu}), t)^T\ \mathbf{f}_{\boldsymbol{\mu}}(\mathbf{x}_2; \boldsymbol{\mu}) \\ \vdots \\ \nabla_{\mathbf{f}} I(\mathbf{f}(\mathbf{x}_N; \boldsymbol{\mu}), t)^T\ \mathbf{f}_{\boldsymbol{\mu}}(\mathbf{x}_N; \boldsymbol{\mu}) \end{bmatrix}. \tag{16}$$

Because $\mathbf{M}$ depends on time-varying quantities, it may appear that it must be completely recomputed at each time step—a computationally expensive procedure involving the calculation of the image gradient vector, the calculation of a $2 \times n$ Jacobian matrix, and n 2×1 vector inner products for each of the N pixels of the target region. However, we now show that it is possible to reduce this computation by both eliminating the need to recompute image gradients and by factoring $\mathbf{M}$.

First, we eliminate the need to compute image gradients. To do so, let us assume that our estimate is *exact*, i.e. $\boldsymbol{\mu}(t) = \boldsymbol{\mu}^*(t)$. By differentiating both sides of (1) we obtain

$$\nabla_{\mathbf{x}} I(\mathbf{x}, t_0) = \mathbf{f}_{\mathbf{x}}(\mathbf{x}; \boldsymbol{\mu})^T \ \nabla_{\mathbf{f}} I(\mathbf{f}(\mathbf{x}; \boldsymbol{\mu}), t) \tag{17}$$

where $\mathbf{f}_{\mathbf{x}}$ is the 2×2 Jacobian matrix of $\mathbf{f}$ treated as a function of $\mathbf{x} = (x, y)^T$,

$$\mathbf{f}_{\mathbf{x}}(\mathbf{x}; \boldsymbol{\mu}) = \left[\frac{\partial \mathbf{f}(\mathbf{x}; \boldsymbol{\mu})}{\partial x} \ \Big| \ \frac{\partial \mathbf{f}(\mathbf{x}; \boldsymbol{\mu})}{\partial y} \right]. \tag{18}$$

Combining (17) with (16), we see that $\mathbf{M}$ can be written as

$$\mathbf{M}(\boldsymbol{\mu}) = \begin{bmatrix} \nabla_{\mathbf{x}} I(\mathbf{x}_1, t_0)^T \ \mathbf{f}_{\mathbf{x}}(\mathbf{x}_1; \boldsymbol{\mu})^{-1} \ \mathbf{f}_{\boldsymbol{\mu}}(\mathbf{x}_1; \boldsymbol{\mu}) \\ \nabla_{\mathbf{x}} I(\mathbf{x}_2, t_0)^T \ \mathbf{f}_{\mathbf{x}}(\mathbf{x}_2; \boldsymbol{\mu})^{-1} \ \mathbf{f}_{\boldsymbol{\mu}}(\mathbf{x}_2; \boldsymbol{\mu}) \\ \vdots \\ \nabla_{\mathbf{x}} I(\mathbf{x}_N, t_0)^T \ \mathbf{f}_{\mathbf{x}}(\mathbf{x}_N; \boldsymbol{\mu})^{-1} \ \mathbf{f}_{\boldsymbol{\mu}}(\mathbf{x}_N; \boldsymbol{\mu}) \end{bmatrix}. \tag{19}$$

It follows that for *any choice* of image deformations, the image spatial gradients need only be calculated once on the reference template. This is not surprising given that the target at time $t > t_0$ is only a distortion of the target at time t_0, and so its image gradients are also a distortion of those at t_0. This transformation also allows us to drop the time argument of $\mathbf{M}$ and regard it solely as a function of $\boldsymbol{\mu}$.

The remaining non-constant factor in $\mathbf{M}$ is a consequence of the fact that, in general, $\mathbf{f}_{\mathbf{x}}$ and $\mathbf{f}_{\boldsymbol{\mu}}$ involve components of $\boldsymbol{\mu}$ and, hence, implicitly vary with time. However, suppose that we choose $\mathbf{f}$ so that $\mathbf{f}_{\mathbf{x}}^{-1} \ \mathbf{f}_{\boldsymbol{\mu}}$ can be factored into the product of a $2 \times k$ matrix $\boldsymbol{\Gamma}$ which depends only on image coordinates, and a $k \times n$ matrix $\boldsymbol{\Sigma}$ which depends only on $\boldsymbol{\mu}$ as

$$\mathbf{f}_{\mathbf{x}}(\mathbf{x}; \boldsymbol{\mu})^{-1} \ \mathbf{f}_{\boldsymbol{\mu}}(\mathbf{x}; \boldsymbol{\mu}) = \boldsymbol{\Gamma}(\mathbf{x}) \ \boldsymbol{\Sigma}(\boldsymbol{\mu}). \tag{20}$$

For example, as discussed in more detail below, one family of such factorizations results when $\mathbf{f}$ is a linear function of the image coordinate vector $\mathbf{x}$.

Combining (19) with (20), we have

$$\mathbf{M}(\boldsymbol{\mu}) = \begin{bmatrix} \nabla_{\mathbf{x}} I(\mathbf{x}_1, t_0)^T \, \boldsymbol{\Gamma}(\mathbf{x}_1) \\ \nabla_{\mathbf{x}} I(\mathbf{x}_2, t_0)^T \, \boldsymbol{\Gamma}(\mathbf{x}_2) \\ \vdots \\ \nabla_{\mathbf{x}} I(\mathbf{x}_N, t_0)^T \, \boldsymbol{\Gamma}(\mathbf{x}_N) \end{bmatrix} \boldsymbol{\Sigma}(\boldsymbol{\mu}) = \mathbf{M_0} \, \boldsymbol{\Sigma}(\boldsymbol{\mu}). \tag{21}$$

As a result, we have shown that $\mathbf{M}$ can be written as a product of an *constant* $N \times k$ matrix $\mathbf{M_0}$ and a time-varying $k \times n$ matrix $\boldsymbol{\Sigma}$.

We can now exploit this factoring to define an efficient tracking algorithm which operates as follows:

offline:

- Define the target region.
- Acquire and store the reference template.
- Compute and store $\mathbf{M_0}$ and $\boldsymbol{\Lambda} = \mathbf{M_0}^T \mathbf{M_0}$.

online:

- Use the most recent motion parameter estimate $\boldsymbol{\mu}(t)$ to rectify the target region in the current image.
- Compute $\mathbf{e}(t + \tau)$ by taking the difference between the rectified image and the reference template.
- Solve the system $\boldsymbol{\Sigma}^T \boldsymbol{\Lambda} \boldsymbol{\Sigma} \; \delta\boldsymbol{\mu} = \boldsymbol{\Sigma}^T \; \mathbf{M_0}^T \; \mathbf{e}(t + \tau)$ for $\delta\boldsymbol{\mu}$, where $\boldsymbol{\Sigma}$ is evaluated at $\boldsymbol{\mu}(t)$.
- Compute $\boldsymbol{\mu}(t + \tau) = \boldsymbol{\mu}(t) + \delta\boldsymbol{\mu}$.

The online computation performed by this algorithm is quite small, and consists of two $n \times k$ matrix multiplies, k N-vector inner products, n k-vector inner products, and an $n \times n$ linear system solution, where k and n are typically far smaller than N.

We note that the computation can be further reduced if $\boldsymbol{\Sigma}$ is invertible. In this case, the solution to the linear system can be expressed as

$$\delta\boldsymbol{\mu} = -\boldsymbol{\Sigma}^{-T}(\mathbf{M_0}^T\mathbf{M_0})^{-1}\mathbf{M_0}^T\mathbf{e}(t+\tau), \tag{22}$$

where $\boldsymbol{\Sigma}^{-T} = (\boldsymbol{\Sigma}^{-1})^T$ is evaluated at $\boldsymbol{\mu}(t)$. The factor $(\mathbf{M_0}^T\mathbf{M_0})^{-1}\mathbf{M_0}^T$ can be computed offline, so the online computation is reduced to n N-vector inner products and n n-vector inner products.

2.3 Some Examples

2.3.1 Linear Models

Let us assume that $\mathbf{f}(\mathbf{x};\boldsymbol{\mu})$ is linear in $\mathbf{x}$. Then we have

$$\mathbf{f}(\mathbf{x};\boldsymbol{\mu}) = \mathbf{A}(\boldsymbol{\mu})\mathbf{x} + \mathbf{u}(\boldsymbol{\mu}) \tag{23}$$

and, hence, $\mathbf{f_x} = \mathbf{A}$. It follows that $\mathbf{f}_\mathbf{x}^{-1}\mathbf{f}_{\boldsymbol{\mu}}$ is linear in the components of $\mathbf{x}$ and the factoring defined in (20) applies. We now present three examples illustrating these concepts.

Pure Translation: In the case of pure translation, the allowed image motions are parameterized by the vector $\mathbf{u} = (u, v)$ giving

$$\mathbf{f}(\mathbf{x};\mathbf{u}) = \mathbf{x} + \mathbf{u}. \tag{24}$$

It follows immediately that $\mathbf{f_x}$ and $\mathbf{f}_{\boldsymbol{\mu}}$ are both the 2×2 identity matrix, and therefore

$$\mathbf{M}_0 = [\mathbf{I}_x(t_0)\ |\mathbf{I}_y(t_0)] \tag{25}$$

and $\boldsymbol{\Sigma}$ is the 2×2 identity matrix.

The resulting linear system is nonsingular if the image gradients in the template region are not all collinear, in which case the solution at each time step is just

$$\delta\boldsymbol{\mu} = -(\mathbf{M}_0^T\mathbf{M}_0)^{-1}\mathbf{M}_0^T\mathbf{e}(t+\tau). \tag{26}$$

Note that in this case $\boldsymbol{\Lambda} = -(\mathbf{M}_0^T\mathbf{M}_0)^{-1}\mathbf{M}_0^T$, a constant matrix which can be computed offline.

Translation, Rotation and Scale: Objects which are viewed under scaled orthography and which do not undergo out-of-plane rotation can be modeled using a four parameter model consisting an image-plane rotation through an angle θ, a scaling by s, and a translation by $\mathbf{u}$. The change of coordinates is given by

$$\mathbf{f}(\mathbf{x};\mathbf{u},\theta,s) = s\mathbf{R}(\theta)\mathbf{x} + \mathbf{u} \tag{27}$$

where $\mathbf{R}(\theta)$ is a 2×2 rotation matrix. After some minor algebraic manipulations, we obtain

$$\boldsymbol{\Gamma}(\mathbf{x}) = \begin{bmatrix} 1 & 0 & -y & x \\ 0 & 1 & x & y \end{bmatrix} \tag{28}$$

and

$$\boldsymbol{\Sigma}(\theta,s) = \begin{bmatrix} \frac{1}{s}\mathbf{R}(-\theta) & 0 & 0 \\ \mathbf{0} & 1 & 0 \\ \mathbf{0} & 0 & \frac{1}{s} \end{bmatrix}. \tag{29}$$

From this $\mathbf{M}_0$ can be computed using (21) and, since $\boldsymbol{\Sigma}$ is invertible, the solution to the linear system becomes

$$\delta\boldsymbol{\mu} = -\boldsymbol{\Sigma}^{-T}(\mathbf{M_0}^T\mathbf{M_0})^{-1}\mathbf{M_0}^T\mathbf{e}(t+\tau) \tag{30}$$

This result can be explained as follows. The matrix $\mathbf{M_0}$ is the linearization of the system about $\theta = 0$ and $s = 1$. At time t the target has orientation $\theta(t)$ and $s(t)$. Image rectification effectively rotates the target by $-\theta$ and scales by $\frac{1}{s}$ so the displacements of the target are computed *in the original target coordinate system.* $\boldsymbol{\Sigma}^{-T}$ then applies a change of coordinates to rotate and scale the computed displacements from the original target coordinate system back to the actual target coordinates.

Affine Motion: The image distortions of planar objects viewed under orthographic projection are described by a six-parameter linear change of coordinates. Suppose that we define

$$\boldsymbol{\mu} = (u, v, a, b, c, d)^t \tag{31}$$

$$\mathbf{f}(\mathbf{x};\boldsymbol{\mu}) = \begin{bmatrix} a & c \\ b & d \end{bmatrix}\mathbf{x} + \begin{bmatrix} u \\ v \end{bmatrix} = \mathbf{A}\mathbf{x} + \mathbf{u} \tag{32}$$

After some minor algebraic manipulations, we obtain

$$\boldsymbol{\Gamma}(\mathbf{x}) = \begin{bmatrix} 1 & 0 & x & 0 & y & 0 \\ 0 & 1 & 0 & x & 0 & y \end{bmatrix} \tag{33}$$

and

$$\mathbf{\Sigma}(\boldsymbol{\mu}) = \begin{bmatrix} \mathbf{A}^{-1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{A}^{-1} \end{bmatrix}. \tag{34}$$

Note that $\mathbf{\Sigma}$ is once again invertible which allows for additional computational savings as before.

2.3.2 Nonlinear Motion Models

The separability property needed for factoring does not hold for any type of nonlinear motion. However, consider a motion model of the form

$$\mathbf{f}(\mathbf{x}; u, v, a) = \mathbf{x} + \begin{bmatrix} u \\ v + 1/2ax^2 \end{bmatrix} \tag{35}$$

where $\mathbf{x} = (x, y)^T$. Intuitively, this model performs a quadratic distortion of the image according to the equation $y = 1/2ax^2$. For example, a polynomial model of this form was used in [8] to model the motions of lips and eyebrows on a face. Again, after several algebraic steps we arrive at

$$\mathbf{\Gamma}(\mathbf{x}) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & x & x^2/2 \end{bmatrix} \tag{36}$$

and

$$\mathbf{\Sigma}(\boldsymbol{\mu}) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -a & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{37}$$

Note this general result holds for any distortion which can be expressed exclusively as either $y = f(x)$ or $x = g(y)$. However, adding more freedom to the motion model, for example combining affine and polynomial distortion, often makes factoring impossible. One possibility in such cases is to use a *cascaded model* in which the image is first rectified using an affine distortion model, and then the resulting rectified image is further rectified for polynomial distortion.

2.4 On the Structure of Image Change

The Jacobian matrix $\mathbf{M}$ plays a central role in the algorithms described above, so it is informative to digress briefly on its structure. If we consider the rectified image as a continuous time-varying quantity, then its total derivative with respect to time is

$$\frac{d\mathbf{I}}{dt} = \mathbf{M}\frac{d\boldsymbol{\mu}}{dt} + \mathbf{I}_t \quad \text{or} \quad \dot{\mathbf{I}} = \mathbf{M}\dot{\boldsymbol{\mu}} + \mathbf{I}_t. \tag{38}$$

Note that this is simply a differential form of (8). Due to the image constancy assumption (1), it follows that $\dot{\mathbf{I}} = \mathbf{0}$ when $\boldsymbol{\mu} = \boldsymbol{\mu}^*$. This is, of course, a parameterized version of Horn's optical flow constraint equation [28].

In this form, it is clear that the role of $\mathbf{M}$ is to relate variations in motion parameters to variations in brightness values in the target region. The solution given in (13) effectively reverses this relationship and provides a method for interpreting observed *changes* in brightness as motion. In this sense, we can think of the algorithm as performing correlation on temporal changes (as opposed to spatial structure) to compute motion.

To better understand the structure of $\mathbf{M}$, recall that in column form, it can be written in terms of the partial derivatives of the rectified image:

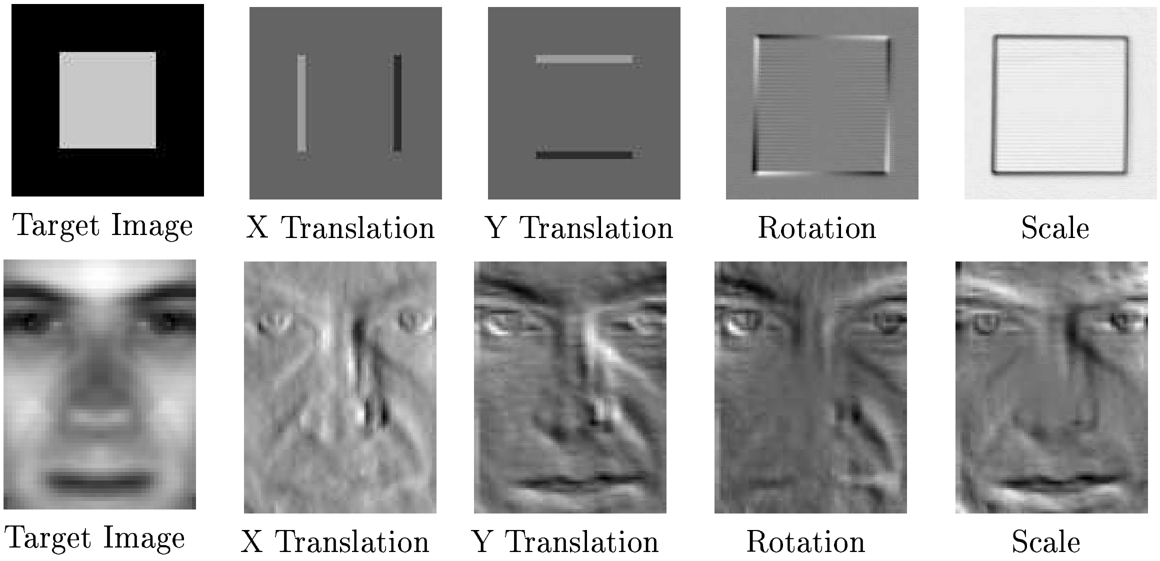
$$\mathbf{M} = [\mathbf{I}_{\mu_1} | \mathbf{I}_{\mu_2} | \ldots | \mathbf{I}_{\mu_n}]. \tag{39}$$

Thus, the model states that the temporal variation in image brightness in the target region is a weighted combination of the vectors $\mathbf{I}_{\mu_i}$. We can think of each of these columns (which have an entry for every pixel in the target region) as a "motion template" which directly represents the changes in brightness induced by the motion represented by the corresponding motion parameter. For example, in the top row of Figure 1, we have shown these templates for several canonical motions of an image of a black square on a white background. Below, we show the corresponding templates for a human face.

The development in this section has assumed that we start with a given parametric motion model from which these templates are derived. Based on that model, the structure of each entry of $\mathbf{M}$ is given by (15) which states that

$$m_{i,j} = \nabla_{\mathbf{f}} I \cdot \mathbf{f}_{\mu_j} \Big|_{\mathbf{x}=\mathbf{x}_i}. \tag{40}$$

The image gradient $\nabla_{\mathbf{f}} I$ defines, at each point in the image, the direction of strongest intensity change. The vector $\mathbf{f}_{\mu_j}$ evaluated at $\mathbf{x}_i$ is the instantaneous direction and magnitude of motion of that image location captured by the parameter μ_j. The collection of the latter for all pixels in the region represents the *motion field* defined by the motion parameter μ_j.



Target Image X Translation Y Translation Rotation Scale

Target Image X Translation Y Translation Rotation Scale

Figure 1: Above, the reference template for a bright square on a dark background the motion template for four canonical motions. Below, the same motion templates for a human face.

Thus, the change in the brightness of the image location $\mathbf{x}_i$ due to the motion parameter μ_j is the projection of the image gradient onto the motion vector.

This suggests how our techniques can be used to perform structured motion estimation without an explicit parametric motion model. First, if the changes in images due to motion can be observed directly (for example, by computing the differences of images taken before and after small reference motions are performed), then these can be used as the motion templates which comprise $\mathbf{M}$. Second, if a one or more *motion fields* can be observed (for example, by tracking a set of fiducial points in a series of training images), then projecting each element of the motion field onto the corresponding image gradient yields motion templates for those motion fields. The linear estimation process described above can be used to interpret time-varying images in terms of those basis motions.

3 Illumination-Insensitive Tracking

The systems described above are inherently sensitive to changes in illumination of the target region. This is not surprising, as the incremental estimation step is effectively computing a structured optical flow, and optical flow methods are well-known to be sensitive to illumination changes [28]. Thus, shadowing or shading changes of the target object over time lead to bias, or, in the worst case, complete loss of the target.

Recently, it has been shown that a relatively small number of "basis" images can often be used to account for large changes in illumination [5, 17, 22, 24, 43]. 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

$$E = a\,\mathbf{n} \cdot \mathbf{s} \tag{41}$$

where $\mathbf{n}$ is the unit inwards normal vector to the surface at p and a is the non-negative absorption coefficient (albedo) of the surface at the point p [28]. 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, in the absence of self-shadowing, given three images of a Lambertian surface from the same viewpoint taken under three known, linearly independent light source directions, the albedo and surface normal can be recovered; this is the well-known method of photometric stereo [50, 46]. Alternatively, one can reconstruct the image of the surface under a novel lighting direction by a linear combination of the three original images [43]. 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 [5]. Nevertheless, as done in [24], we can still use a linear model as an approximation: a small set of basis images can account for much of the shading changes that occur on patches of non-specular surfaces. Naturally, we need more than three images (we use between 8 and 15) and a higher than three dimensional linear subspace (we use 4 or 5) if we hope to provide good approximation to these effects.

Returning to the problem of region 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 template image, i.e. $\mathbf{B}_1 = \mathbf{I}(\mathbf{0}, t_0)$. To model brightness changes, let us choose the second basis vector to be a column of ones, i.e. $\mathbf{B}_2 = (1, 1, \ldots, 1)^T$.[1] Let us choose the remaining basis vectors by

[1] In practice, choosing a value close to the mean of the brightness of the image produces a more numerically

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 $\mathbf{B} = [\mathbf{B}_1|\mathbf{B}_2|\dots|\mathbf{B}_m]$.

Suppose now that $\boldsymbol{\mu}(t) = \boldsymbol{\mu}^*(t)$ so that the template image and the current target region are registered geometrically at time t. The remaining difference between them is due to illumination. From the above discussion, it follows that interframe changes in the current target region can be approximated by the template image plus a linear combination of the basis vectors $\mathbf{B}$, i.e.

$$\mathbf{I}(\boldsymbol{\mu} + \delta\boldsymbol{\mu}, t + \tau) = \mathbf{I}(\boldsymbol{\mu}, t) + \mathbf{M}\delta\boldsymbol{\mu} + \mathbf{I}_t\tau + \mathbf{B}\boldsymbol{\lambda} + h.o.t \tag{42}$$

where the vector $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \dots, \lambda_m)^T$. Note that because the template 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 more subtle variation – variation that depends both on the geometry of the target object and on the nature of the light sources.

Using the vector-space formulation for motion recovery established in the previous section, it is clear that illumination and geometry can be recovered in one global optimization step solved via linear methods. Incorporating illumination into (7) we have the following modified optimization:

$$O(\delta\boldsymbol{\mu}, \boldsymbol{\lambda}) = \|\mathbf{I}(\boldsymbol{\mu}(t) + \delta\boldsymbol{\mu}, t + \tau) + \mathbf{B}\boldsymbol{\lambda} - \mathbf{I}(\mathbf{0}, t_0)\|^2. \tag{43}$$

Substituting (42) into (43) and performing the same simplifications and approximations as before, we arrive at

$$O(\delta\boldsymbol{\mu}, \boldsymbol{\lambda}) = \|\mathbf{M}\delta\boldsymbol{\mu} + \mathbf{B}\boldsymbol{\lambda} + \mathbf{I}(\boldsymbol{\mu}(t), t + \tau) - \mathbf{I}(\mathbf{0}, t_0)\|^2. \tag{44}$$

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

$$\begin{bmatrix} \delta\boldsymbol{\mu} \\ \boldsymbol{\lambda} \end{bmatrix} = - \begin{bmatrix} \mathbf{M}^T\mathbf{M} & \mathbf{M}^T\mathbf{B} \\ \mathbf{B}^T\mathbf{M} & \mathbf{B}^T\mathbf{B} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{M}^T \\ \mathbf{B}^T \end{bmatrix} \mathbf{e}(t + \tau). \tag{45}$$

In most tracking applications, we are only interested in the motion parameters. We can eliminate explicit computation of these parameters by first optimizing over $\boldsymbol{\lambda}$ in (44). Upon

stable linear system.

substituting the resulting solution back into (44) and then solving for $\delta\boldsymbol{\mu}$ we arrive at

$$\delta\boldsymbol{\mu} = -(\mathbf{M}^T(\mathbf{1} - \mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T)\mathbf{M})^{-1}\mathbf{M}^T(\mathbf{1} - \mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T)\,\mathbf{e}(t+\tau). \tag{46}$$

Note that if the columns of $\mathbf{B}$ are orthogonal vectors, $\mathbf{B}^T\mathbf{B}$ is the identity matrix.

It is easy to show that in both equations, factoring $\mathbf{M}$ into time-invariant and time-varying components as described above leads to significant computational savings. Since the illumination basis is time-invariant, the dimensionality of the time-varying portion of the computation depends only on the number of motion fields to be computed, not on the illumination model. 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.*

4 Making Tracking Resistant to Occlusion

As a system tracks objects over a large space, it is not uncommon that other objects “intrude” into the picture. For example, the system may be in the process of tracking a target region which is the side of a building when, due to observer motion, a parked car begins to occlude a portion of that region. Similarly the target object may rotate, causing the tracked region to “slide off” and pick up a portion of the background. Such intrusions will bias the motion parameter estimates and, in the long term can potentially cause mistracking. In this section, we describe how to avoid such problems. For the sake of simplicity, we develop a solution for the case where we are only recovering motion parameters; the modifications for combined motion and illumination models are straightforward.

A common approach to this problem is to assume that occlusions create large image differences which can be viewed as “outliers” by the estimation process [7]. The error metric is then modified to reduce sensitivity to “outliers” by solving a robust optimization problem of the form

$$O_R(\boldsymbol{\mu}) = \sum_{\mathbf{x}\in\mathcal{R}} \rho(I(f(\mathbf{x};\boldsymbol{\mu}),t) - I(\mathbf{x},t_0)) \tag{47}$$

where ρ is one of a variety of “robust” regression metrics [31].

It is well-known that optimization of (47) is closely related to another approach to robust estimation—iteratively reweighted least squares (IRLS). We have chosen to implement the optimization using a somewhat unusual form of IRLS due to Dutter and Huber [16]. In order to formulate the algorithm, we introduce the notation of an "inner iteration" which is performed one or more times at each time step. We will use a superscript to denote this iteration.

Let $\delta\boldsymbol{\mu}^i$ denote the value of $\delta\boldsymbol{\mu}$ computed by the ith inner iteration with $\delta\boldsymbol{\mu}^0 = \mathbf{0}$. Define the vector of residuals in the ith iteration $\mathbf{r}^i$ as

$$\mathbf{r}^i = \mathbf{e}(t+\tau) - \mathbf{M}(\boldsymbol{\mu})\delta\boldsymbol{\mu}^i. \tag{48}$$

We introduce a diagonal weighting matrix $\mathbf{W}^i = \mathbf{W}(\mathbf{r}^i)$ which has entries

$$w^i_{k,k} = \eta(r^i_k) = \rho'(r^i_k)/r^i_k. \tag{49}$$

The inner iteration cycle at time $t+\tau$ is consists of performing an estimation step by solving the linear system

$$\boldsymbol{\Sigma}^T\boldsymbol{\Lambda}\boldsymbol{\Sigma}\ \delta\boldsymbol{\mu}^{i+1} = \boldsymbol{\Sigma}^T\mathbf{M_0}^T\ \mathbf{W}^i\ \mathbf{r}^i \tag{50}$$

where $\boldsymbol{\Sigma}$ is evaluated at $\boldsymbol{\mu}(t))$ and $\mathbf{r}^i$ and $\mathbf{W}^i$ are given by (48) and (49), respectively. This process is repeated for k iterations.

This form of IRLS is particularly efficient for our problem. It does not require recomputation of $\boldsymbol{\Lambda}$ or $\boldsymbol{\Sigma}$ and, since the weighting matrix is diagonal, does not add significantly to the overall computation time needed to solve the linear system. In addition, the error vector $\mathbf{e}$ is fixed over all inner iterations, so these iterations do not require the additional overhead of acquiring and warping images.

As discussed in [16], on linear problems this procedure is guaranteed to converge to a unique global minimum for a large variety of choices of ρ. In this article, ρ is taken to be a so-called "windsorizing" function [31] which is of the form

$$\rho(r) = \begin{cases} r^2/2 & \text{if } |r| \le \tau \\ c|r| - c^2/2 & \text{if } |r| > \tau \end{cases} \tag{51}$$

where r is normalized to have unit variance. The parameter τ is a user-defined threshold which places a limit on the variations of the residuals before they are considered outliers. This function has the advantage of guaranteeing global convergence of the IRLS method while being cheap to compute. The updating function for matrix entries is

$$\eta(r) = \begin{cases} 1 & \text{if } |r| \le \tau \\ c/|r| & \text{if } |r| > \tau \end{cases}. \tag{52}$$

As stated, the weighting matrix is computed anew at each iteration, a process which can require several inner iterations. However, given that tracking is a continuous process, it is natural to start with an initial weighting matrix that is closely related to that computed at the end of the previous estimation step. In doing so, two issues arise. First, the fact that the linear system we are solving is a local linearization of a nonlinear system means that, in cases when inter-frame motion is large, the effect of higher-order terms of the Taylor series expansion will cause areas of the image to masquerade as outliers. Second, if we assume that areas of the image with low weights correspond to intruders, it makes sense to add a "buffer zone" around those areas for the next iteration to pro-actively cancel the effects of intruder motion.

Both of these problems can be dealt with by noting that the diagonal elements of $\mathbf{W}$ themselves form an image where "dark areas" (those locations with low value) are areas of occlusion or intrusion, while "bright areas" (those with value 1) are the expected target. Let $Q(\mathbf{x})$ to be the pixel values in the eight-neighborhood of the image coordinate $\mathbf{x}$ plus the value at $\mathbf{x}$ itself. We use two common morphological operators [26]

$$\mathrm{close}(\mathbf{x}) = \max_{v \in Q(\mathbf{x})} v \tag{53}$$

and

$$\mathrm{open}(\mathbf{x}) = \min_{v \in Q(\mathbf{x})} v. \tag{54}$$

When applied to a weighting matrix image, close has the effect of removing small areas of outlier pixels, while open increases their size. Between frames of the sequence we propagate the weighting matrix forward after applying one step of close to remove small areas of outliers followed by two or three steps of open to buffer detected intruders.

5 Implementation and Experiments

This section illustrates the performance of the tracking algorithm under a variety of circumstances, noting particularly the effects of image warping, illumination compensation, and outlier detection. All experiments were performed on live video sequences by an SGI Indy equipped with a 175Mhz R4400 SC processor and VINO image acquisition system.

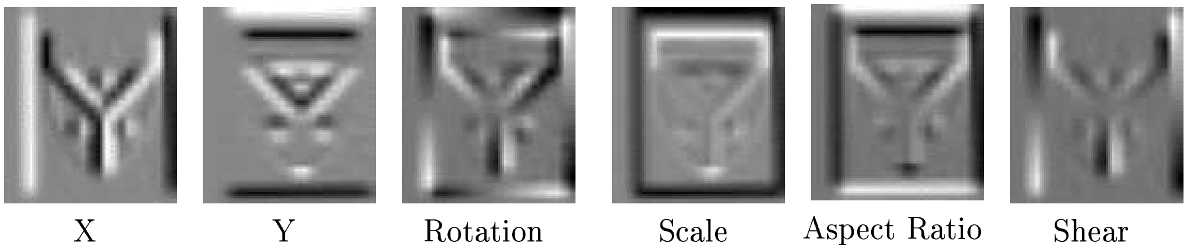


Figure 2: The columns of the motion Jacobian matrix for the planar target and their geometric interpretations.

5.1 Implementation

We have implemented the methods described above within the X Vision environment [23]. The implemented system incorporates all of the linear motion models described in Section 2, non-orthonormal illumination bases as described in Section 3, and outlier rejection using the algorithm described in Section 4.

The image warping required to support the algorithm is implemented by factoring linear transformations into a rotation matrix and a positive-definite upper-diagonal matrix. This factoring allows image warping to be implemented in two stages. In the first stage, an image region surrounding the target is acquired and rotated using a variant on standard Bresenham line-drawing algorithms [18]. The acquired image is then scaled and sheared using a bilinear interpolation. The resolution of the region is then reduced by averaging neighboring pixels. Spatial and temporal derivatives are computed by applying Prewitt operators on the reduced scale images. More details on this level of the implementation can be found in [23].

Timings of the algorithm[2] indicate that it can perform frame rate (30 Hz) tracking of image regions of up to 100 × 100 pixels at one-half resolution undergoing affine distortions and illumination changes. Similar performance has been achieved on a 120Mhz Pentium processor and 70 Mhz Sun SparcStation. Higher performance is achieved for smaller regions, lower resolutions, or fewer parameters. For example, tracking the same size region while computing just translation at one-fourth resolution takes just 4 milliseconds per cycle.

5.2 Planar Tracking

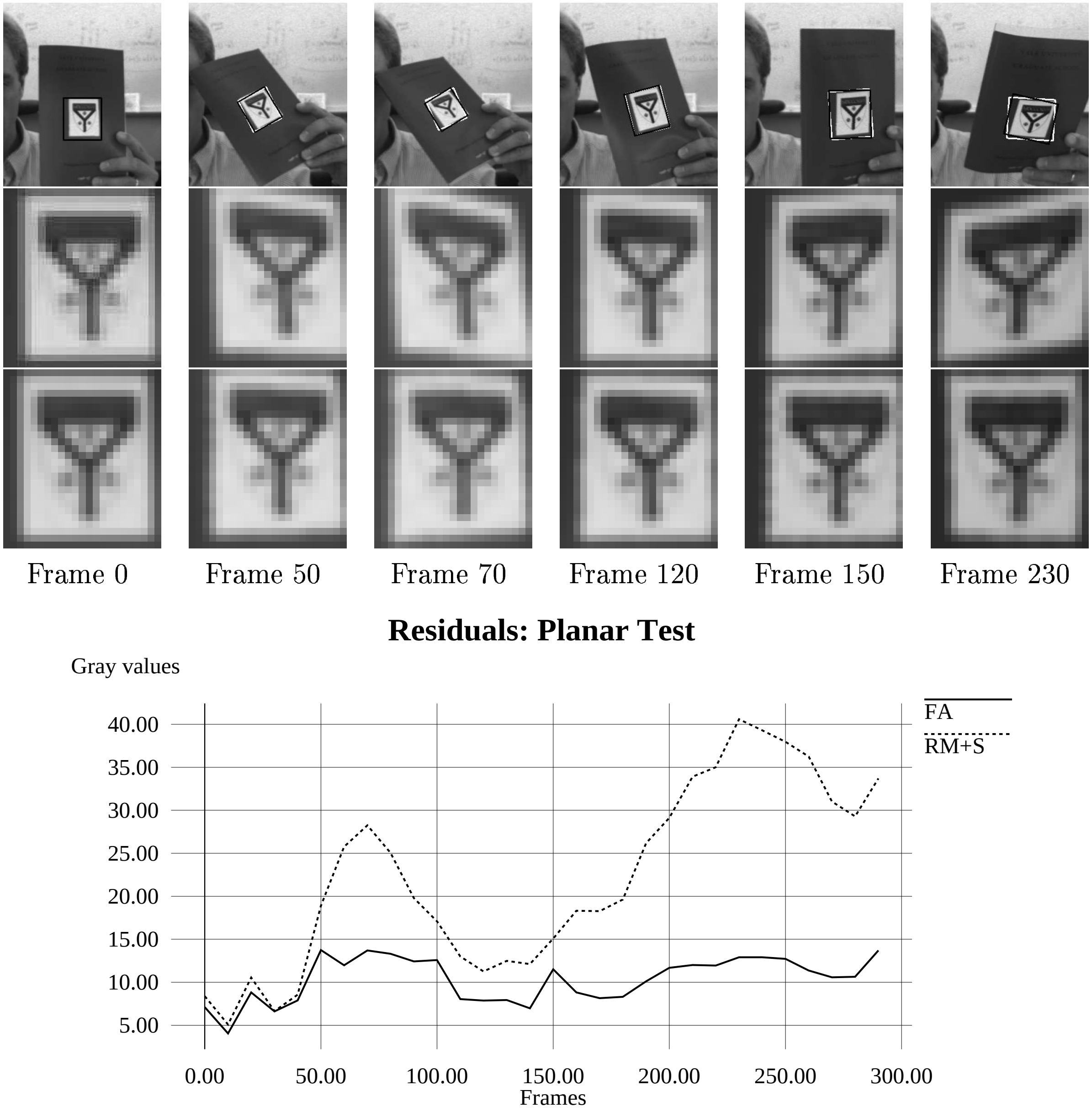
As a baseline, we first consider tracking a non-specular planar object—the cover of a book. Affine warping augmented with brightness and contrast compensation is the best possible linear approximation to this case (it is exact for an orthographic camera model and purely Lambertian surface). As a point of comparison, recent work by Black and Jepson [7] used the rigid motion plus scaling model for SSD-based region tracking. Their reduced model is more efficient and may be more stable since fewer parameters must be computed, but it does ignore the effects of changing aspect ratio and shear.

We tested both the rigid motion plus scale (RM+S) and full affine (FA) motion models on the same live video sequence of the book cover in motion. Figure 2 shows the set of motion templates (the columns of the motion matrix) for an 81×72 region of a book cover tracked at one third resolution. Figure 3 shows the results of tracking. The upper series of images shows several images of the object with the region tracked indicated with a black frame (the RM+S algorithm) and a white frame (the FA algorithm). The middle row of images shows the output of the warping operator from the RM+S algorithm. If the computed parameters were error-free, these images would be identical. However, because of the inability to correct for aspect ratio and skew, the best fit leads to a skewed image. The bottom row shows the output of the warping operator for the FA algorithm. Here we see that the full affine warping is much better at accommodating the full range of image distortions. The graph at the bottom of the figure shows the least squares residual (in squared gray-values per pixel). Here, the difference between the two geometric models is clearly evident.

5.3 Human Face Tracking

There has been a great deal of recent interest in face tracking in the computer vision literature [8, 14, 36]. Although faces can produce images with significant variation due to illumination, empirical results suggest that 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 [24]. At the same time, the depth variations exhibited by facial features are small enough to be well-approximated by an affine warping model. The following

[2] Because of additional data collection overhead, the tracking performance in the experiments presented here is slower than the stated figures.



Frame 0 Frame 50 Frame 70 Frame 120 Frame 150 Frame 230

Figure 3: Top, several images of a planar region and the corresponding warped image computed by a tracker computing position, orientation and scale (RM+S), and one computing a full affine deformation (FA). The image at the left is the initial reference image. Bottom, the graph of the SSD residuals for both algorithms.

experiments demonstrate the ability of our algorithm to track a face as it undergoes changes in pose and illumination, and under partial occlusion. Throughout, we assume the subject is roughly looking toward the camera, so we use the rigid motion plus scaling (RM+S) motion model. Figure 1 on page 14 shows the columns of the motion matrix for this model.

5.3.1 Geometry

We first performed a test to determine the accuracy of the computed motion parameters for the face and to investigate the effect of the illumination basis on the sensitivity of those estimates. During this test, we simultaneously executed two tracking algorithms: one using the rigid motion plus scale model (RM+S) and one which additionally included an illumination model for the face (RM+S+I). The algorithms were executed on a sequence which did not contain large changes in the illumination of the target. The top row of Figure 4 shows images excerpted from the video sequence. In each image, the black frames denote the region selected as the best match by RM+S and the white frames correspond to the best match computed by RM+S+I. For this test, we would expect both algorithms to be quite accurate and to exhibit similar performance unless the illumination basis significantly affected the sensitivity of the computation. As is apparent from the figures, the computed motion parameters of both algorithms are extremely similar for the entire run — so close that in many cases one frame is obscured by the other.

In order to demonstrate the absolute accuracy of the tracking solution, below each live image in Figure 4 we have included the corresponding rectified image computed by RM+S+I. The rectified image at time 0 is the reference template. If the motion of the target fit the RM+S motion model, and the computed parameters were exact, then we would expect each subsequent rectified image to be identical to the reference template. Despite the fact that the face is non-planar and we are using a reduced motion model, we see that the algorithm is quite effective at computing an accurate geometric match.

Finally, the graph in Figure 4 shows the residuals of the linearized SSD computation at each time step. As is apparent from the figures, the residuals of both algorithms are also extremely similar for the entire run. From this experiment we conclude that, in the absence of illumination changes, the performance of both algorithms is quite similar — including illumination models does not appear to reduce accuracy.

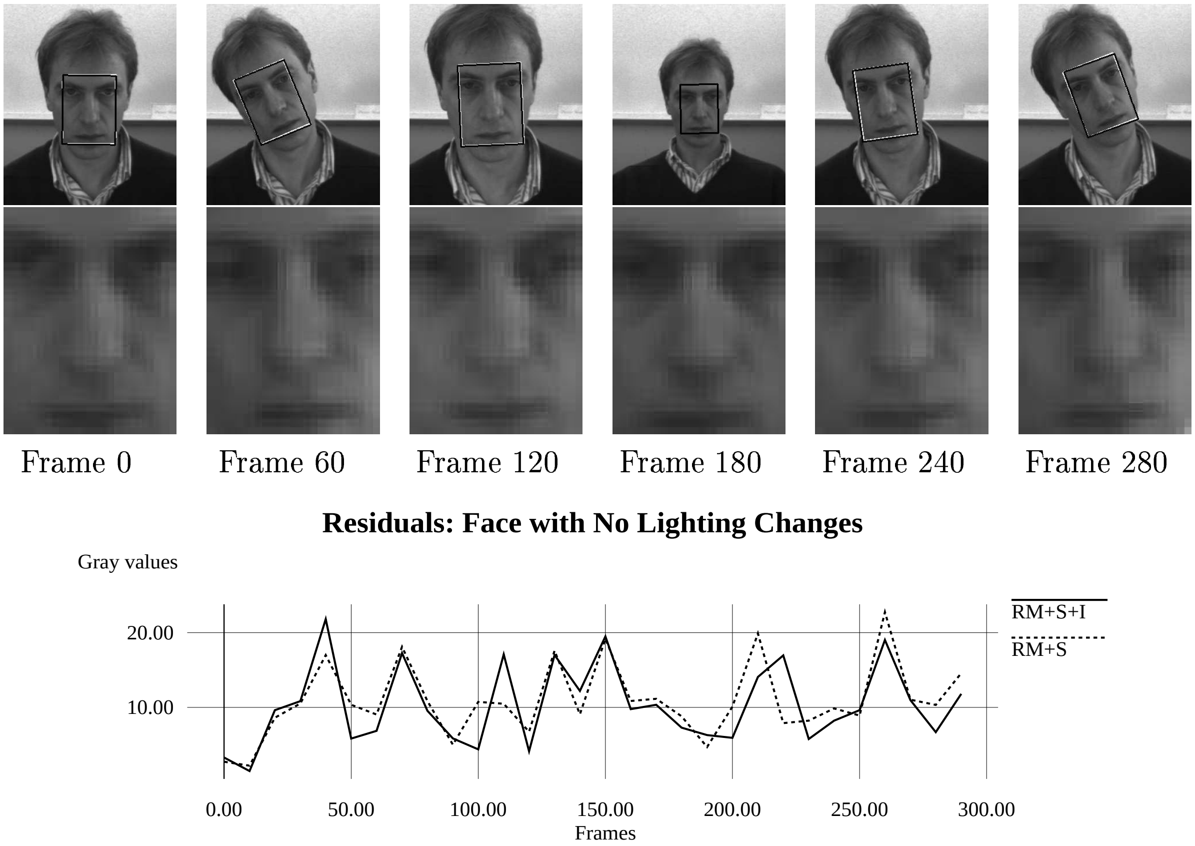


Figure 4: Top row, excerpts from a sequence of tracked images of a face. The black frames represent the region tracked by an SSD algorithm using no illumination model (RM+S) and the white frames represent the regions tracked by an algorithm which includes an illumination model (RM+S+I). In some cases the estimates are so close that only one box is visible. Middle row, the region within the frame warped by the current motion estimate. Bottom row, the residuals of the algorithms expressed in gray-scale units per pixel as a function of time.

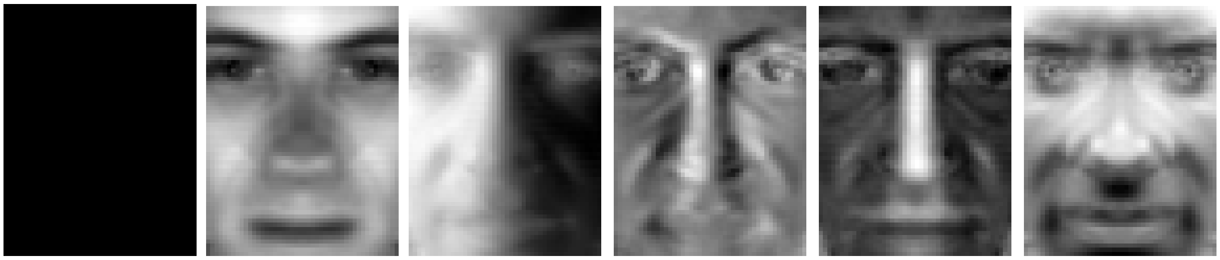


Figure 5: The illumination basis for the face (**B**). The left two images are included to compensate for brightness and contrast, respectively, while the remaining four images compensate for changes in lighting direction.

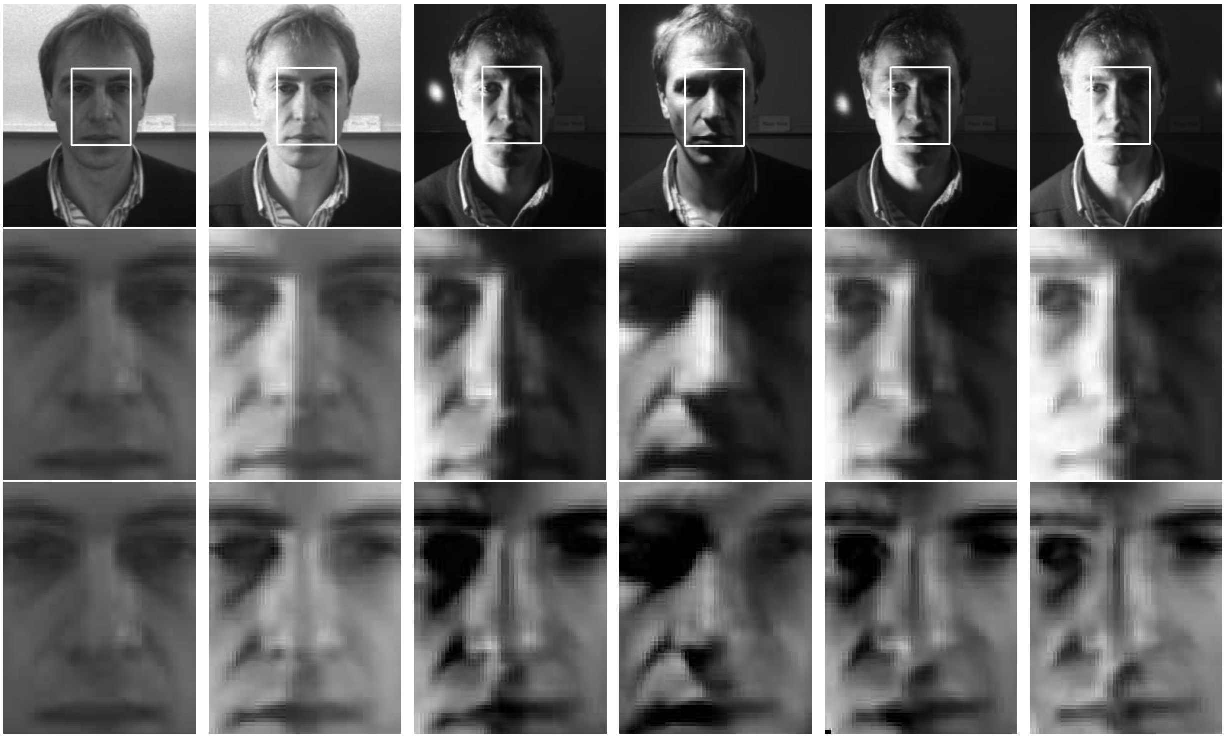
5.3.2 Illumination

In a second set of experiments, we kept the face nearly motionless and varied the illumination. We used an illumination basis of four orthogonal image vectors. This basis was computed offline by acquiring ten 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 illumination basis is shown in Figure 5.

Figure 6 shows the effects of 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 "corrected" by the illumination basis.

5.3.3 Combining Illumination and Geometry

Next, we present a set of experiments illustrating the interaction of geometry and illumination. In these experiments we again executed two algorithms again labeled RM+S and RM+S+I. As the algorithms were operating, a light was periodically switched on and off and the face moved slightly. The results appear in Figure 7. 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 RM+S algorithm (which could not compensate for il-



Frame 0 Frame 20 Frame 70 Frame 110 Frame 120 Frame 150

Figure 6: The first row of images shows excerpts of a tracking sequence. The second row is a magnified view of the region in the white frame. The third row contains the images in the second row after adjustment for illumination using the illumination basis shown in Figure 5 (for sake of comparison we have not adjusted for brightness and contrast).

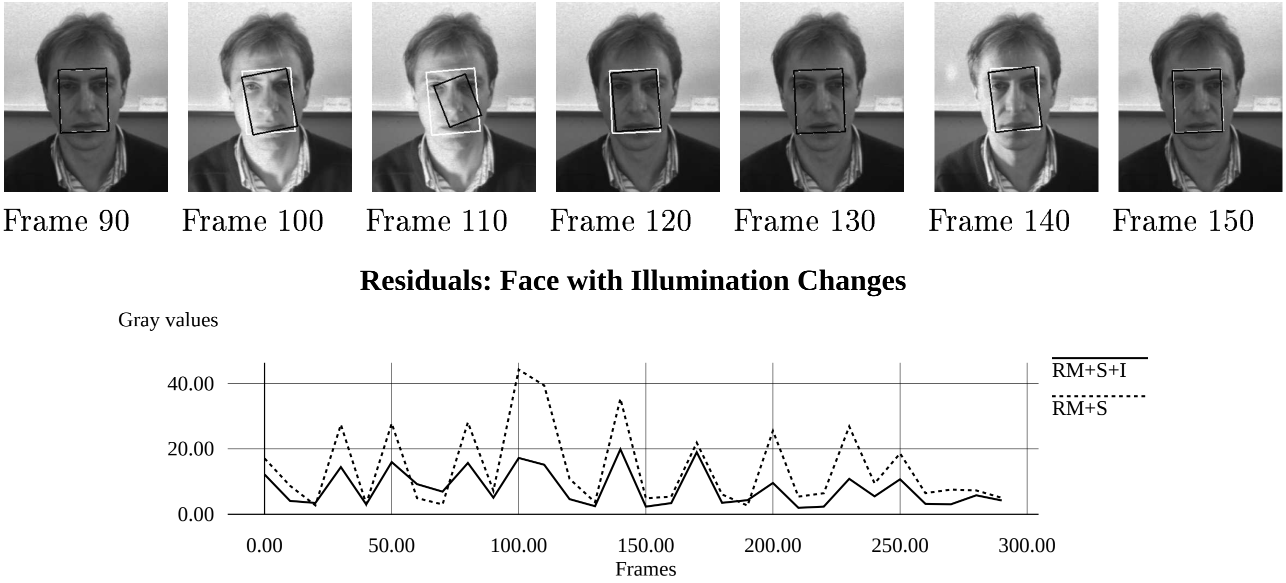

Frame 90 Frame 100 Frame 110 Frame 120 Frame 130 Frame 140 Frame 150

Figure 7: Top, an excerpt from a tracking sequence containing changes in both geometry and illumination. The black frame corresponds to the algorithm without illumination (RM+S) and the write frame corresponds to the algorithm with an illumination basis (RM+S+I). Note that the algorithm which does not use illumination completely looses the target until the original lighting is restored. Bottom, the residuals, in gray scale units per pixel, of the two algorithms as a light is turned on and off.

lumination changes) 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.

5.3.4 Tracking With Outliers

Finally, we illustrate the performance of the method when the image of the target becomes partially occluded. We again track a face. The motion and illumination basis are the same as before. In the weighting matrix calculations the pixel variance was set to 5 and the outlier threshold was set to 5 variance units.

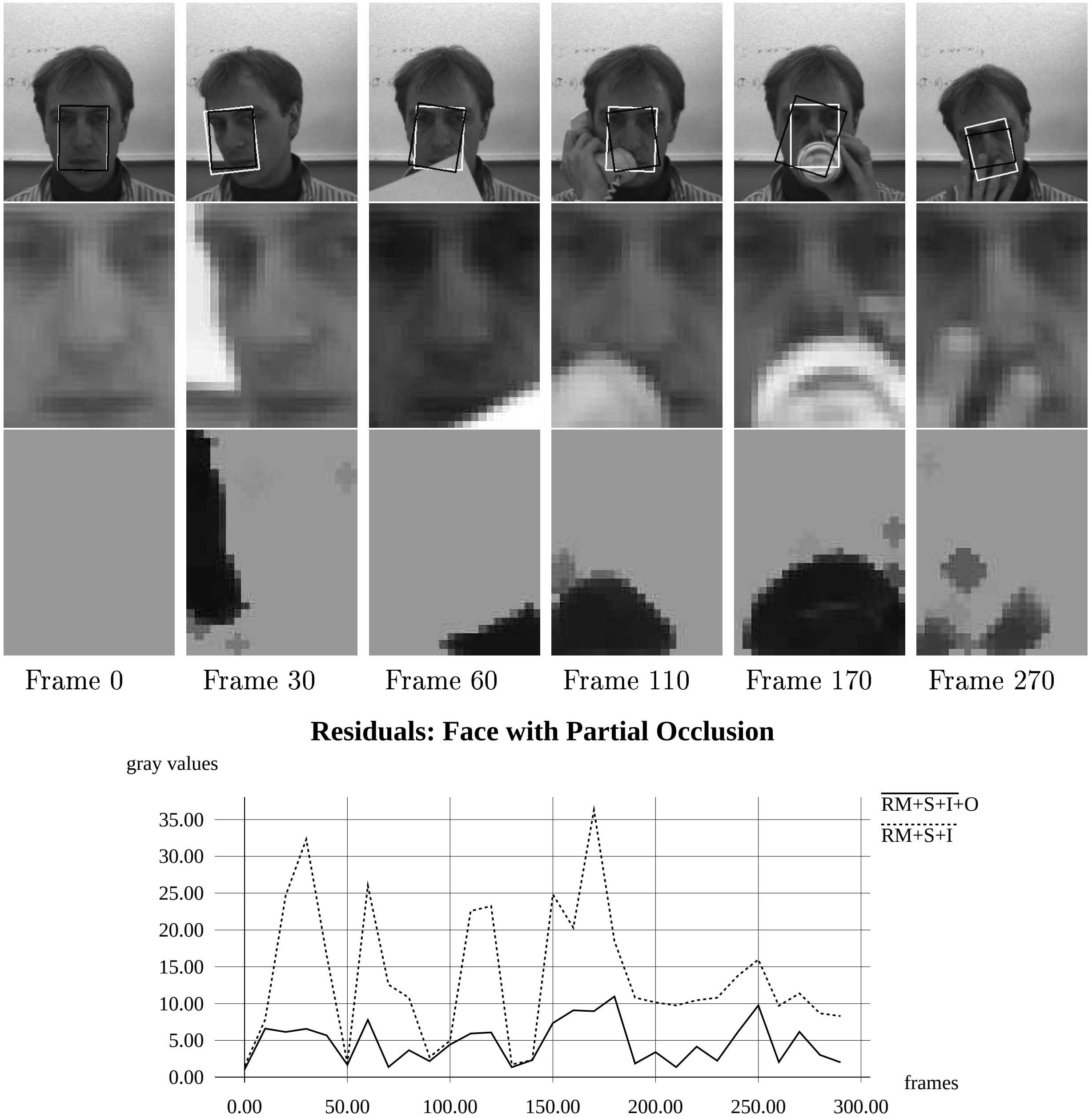
The sequence is an “office” sequence which includes several “intrusions” including the background, a piece of paper, a telephone, a soda can, and a hand. As before we executed two versions of the tracker, the non-robust algorithm from the previous experiment (RM+S+I) and a robust version (RM+S+I+O). Figure 8 shows the results. The upper series of images shows the region acquired by both algorithms (the black frame corresponds to RM+S+I, the white to RM+S+I+O). As is clear from the sequence, the non-robust algorithm is disturbed significantly by the occlusion, whereas the robust algorithm is much more stable. In fact, a slight motion of the head while the soda can is in the image caused the non-robust algorithm to mistrack completely. The middle series of images shows the output of the warping operation for the robust algorithm. The lower row of images depicts the weighting values attached to each pixel in the warped image. Dark areas correspond to "outliers." Note that, although the occluded region is clearly identified by the algorithm, there are some small regions away from the occlusion which received a slightly reduced weight. This is due to the fact that the robust metric used introduces some small bias into the computed parameters. In areas where the spatial gradient is large (e.g. near the eyes and mouth), this introduces some false rejection of pixels.

It is also important to note that the dynamical performance of the tracker is significantly reduced by including outliers. Large, fast motions tend to cause the algorithm to "turn off" areas of the image where there are large gradients, slowing convergence. At the same time, performing outlier rejection is more computationally intensive as it requires explicit computation of both the motion and illumination parameter to calculate the residual values.

6 Discussion and Conclusions

We have shown a straightforward and efficient solution to the problem of tracking regions undergoing geometric distortion, changing illumination, and partial occlusion. The method is simple, yet robust, and it builds on an already popular method for solving spatial and temporal correspondence problems.

Although the focus in this article has been on parameter estimation techniques for tracking using image rectification, the same estimation methods can be used for directly controlling devices. For example, instead of computing a parameter estimate $\boldsymbol{\mu}$, the incremental solutions $\delta\boldsymbol{\mu}$ can be used to control the position and orientation of a camera so to stabilize the target image by active motion. Hybrid combinations of camera control and image warping are also possible.



Frame 0 Frame 30 Frame 60 Frame 110 Frame 170 Frame 270

Residuals: Face with Partial Occlusion

Figure 8: The first row of images shows excerpts of a tracking sequence with occurrences of partial occlusion. The black frame corresponds to the algorithm without outlier rejection (RM+S+I) and the write frame corresponds to the algorithm with outlier rejection (RM+S+I+O). The second row is a magnified view of the region in the white frame. The third row contains the corresponding outlier images where darker areas mark outliers. The graph at the bottom compares the residual values for both algorithms.

One possible objection to the methods is the requirement that the change from frame to frame is small, limiting the speed at which objects can move. Luckily, there are several means for improving the dynamical performance of the algorithms. One possibility is to include a model for the motion of the underlying object and to incorporate prediction into the tracking algorithm. Likewise, if a model of the noise characteristics of images is available, the updating method can modified to incorporate this model. In fact, the linear form of the solution makes it straightforward to incorporate the estimation algorithm into a Kalman filter or similar iterative estimation procedure.

Performance can also be improved by operating the tracking algorithm at multiple levels of resolution. One possibility, as is used by many authors [7, 44], is to perform a complete coarse to fine progression of estimation steps on each image in the sequence. Another possibility, which we have used successfully in prior work [23], is to dynamically adapt resolution based on the motion of the target. That is, when the target moves quickly estimation is performed at a coarse resolution, and when it moves slowly the algorithm changes to a higher resolution. The advantage of this approach is that it not only increases the range over which the linearized problem is valid, but it also reduces the computation time required on each image when motion is fast.

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 problem of determining an illumination basis online, i.e. while tracking the object. Initial experiments in this direction have shown that online determination of the illumination basis can be achieved, although we have not included such results in this paper. As in [7], we are also exploring the use of basis images to handle changes of view or aspect not well addressed by warping.

We are also looking at the problem of extending the method to utilize shape information on the target when such information is available. In particular, it is well known [49] that under orthographic projection, the image deformations of a surface due to motion can be described with a linear motion model. This suggests that our methods can be extended to handle such models. Furthermore, as with the illumination basis, it may be possible to estimate the deformation models online, thereby making it possible to efficiently track arbitrary objects under changes in illumination, pose, and partial occlusion.

Acknowledgments

This research was supported by ARPA grant N00014-93-1-1235, Army DURIP grant DAAH04-95-1-0058, National Science Foundation grant IRI-9420982, Army Research Office grant DAAH04-95-1-0494, and by funds provided by Yale University. The second author would like to thank David Mumford, Alan Yuille, David Kriegman, and Peter Hallinan for discussions contributing the ideas in this paper.

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] N. Ayache and O.D. Faugeras. Maintaining representations of the environment of a mobile robot. *IEEE Trans. on Robotics and Automation*, 5(6):804–819, December 1989.

[4] E. Bardinet, L. Cohen, and N. Ayache. Tracking medical 3D data with a deformable parametric model. In *Proc. European Conf. on Computer Vision*, pages I:317–328, 1996.

[5] P. N. Belhumeur and D. J. Kriegman. What is the set of images of an object under all possible lighting conditions. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, 1996. In press.

[6] M. Betke and N. Makris. Fast object recognition in noisy images using simulated annealing. In *Proceedings of the ICCV*, pages 523–530, 1995.

[7] 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.

[8] 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.

[9] A. Blake, R. Curwen, and A. Zisserman. A framework for spatio-temporal control in the tracking of visual contour. *Int. J. Computer Vision*, 11(2):127–145, 1993.

[10] A. F. Bobick and A. D. Wilson. A state-based technique for the summarization of recognition of gesture. In *Proceedings of the ICCV*, pages 382–388, 1995.

[11] E. Boyer. Object models from contour sequences. In *Proc. European Conf. on Computer Vision*, pages II:109–118, 1996.

[12] J.J. Crisco, K. Hentel, S.W. Wolfe, and J.S. Duncan. Two-dimensional rigid-body kinematics using image registration. *J. Biomechanics*, 1994. In press.

[13] J. L. Crowley, P. Stelmaszyk, T. Skordas, and P. Puget. Measurement and integration of 3-D structures by tracking edge lines. *Int'l Journal of Computer Vision*, 8(1):29–52, 1992.

[14] T. Darrell, B. Moghaddam, and A.P. Pentland. Active face tracking and pose estimation in an interactive room. In *Proc. IEEE Conf. Comp. Vision and Patt. Recog.*, pages 67–72, 1996.

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

[16] R. Dutter and P.J. Huber. Numerical methods for the nonlinear robust regression problem. *J. Statist. Comput. Simulation*, 13(2):79–113, 1981.

[17] R. Epstein, P. Hallinan, and A.L. Yuille. 5 ± 2 Eigenimages suffice: An empirical investigation of low-dimensional lighting models. Technical Report 94-11, Harvard University, 1994.

[18] J.D. Foley, A. van Dam, S.K. Feiner, and J.F. Hughes. *Computer Graphics.* Addison Wesley, 1993.

[19] T. Frank, M. Haag, H. Kollnig, and H.-H. Nagel. Tracking of occluded vehicles in traffic scenes. In *Proc. European Conf. on Computer Vision*, pages II:485–494, 1996.

[20] D.M. Gavrila and L.S. Davis. Tracking humans in action: A 3D model-based approach. In *Proc. Image Understanding Workshop*, pages 737–746, 1996.

[21] D. B. Gennery. Visual tracking of known three-dimensional objects. *Int'l Journal of Computer Vision*, 7(3):243–270, 1992.

[22] G. D. Hager and P.N. Belhumeur. Real-time tracking of image regions with changes in geometry and illumination. In *Proc. IEEE Conf. Comp. Vision and Patt. Recog.*, pages 403–410. IEEE Computer Society Press, 1996.

[23] G. D. Hager and K. Toyama. XVision: A portable substrate for real-time vision applications. *Computer Vision and Image Understanding*, 1996. In Press.

[24] 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.

[25] Peter Hallinan. *A Deformable Model for Face Recognition Under Arbitrary Lighting Conditions.* PhD thesis, Harvard University, 1995.

[26] R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision.* Addison Wesley, 1993.

[27] R. C. Harrell, D. C. Slaughter, and P. D. Adsit. A fruit-tracking system for robotic harvesting. *Machine Vision and Applications*, 2:69–80, 1989.

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

[29] R. Howarth and H. Buxton. Visual surveillance monitoring and watching. In *Proc. European Conf. on Computer Vision*, pages II:321–334, 1996.

[30] Eric Huber and David 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.

[31] P.J. Huber. *Robust Statistics.* John Wiley & Sons, New York, NY, 1981.

[32] S. Hutchinson, G.D. Hager, and P. Corke. A tutorial introduction to visual servo control. *IEEE Trans. on Robotics and Automation*, 12(5), 1996.

[33] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *ECCV96*, pages I:343–356, 1996.

[34] D. G. Lowe. Robust model-based motion tracking through the integration of search and estimation. *Int'l Journal of Computer Vision*, 8(2):113–122, 1992.

[35] 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.

[36] S. McKenna, S. Gong, and J.J. Collins. Face tracking and pose representation. In *British Maching Vision Conference*, 1996.

[37] H. Murase and S. Nayar. Visual learning and recognition of 3-D objects from appearence. *Int. J. Computer Vision*, 14(5–24), 1995.

[38] 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.

[39] J.M. Rehg and T. Kanade. Visual tracking of high DOF articulated structures: An application to human hand tracking. In *Computer Vision – ECCV '94*, volume B, pages 35–46, 1994.

[40] J.M. Rehg and A.P. Witkin. Visual tracking with deformation models. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 844–850, 1991.

[41] D. Reynard, A. Wildenberg, A. Blake, and J. Marchant. Learning dynamics of complex motions from image sequences. In *Proc. European Conf. on Computer Vision*, pages I:357–368, 1996.

[42] L.S. Shapiro. *Affine Analysis of Image Sequences.* Cambridge Univ. Press, 1995.

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

[44] J. Shi and C. Tomasi. Good features to track. In *Proc. IEEE Conf. Comp. Vision and Patt. Recog.*, pages 593–600. IEEE Computer Society Press, 1994.

[45] P. Shi, G. Robinson, T. Constable, A. Sinusas, and J. Duncan. A model-based integrated approach to track myocardial deformation using displacement and velocity constraints. In *Proc. Internal Conf. on Computer Vision*, pages 687–692, 1995.

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

[47] R. Szeliski. Image mosaicing for tele-reality applications. In *Proceedings of the Workshop on Applications of Computer Vision*, pages 44–53, 1994.

[48] 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.

[49] S. Ullman and R. Basri. Recognition by a linear combination of models. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 13:992–1006, 1991.

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