Proceedings of the 1998 IEEE
International Conference on Robotics & Automation
Leuven, Belgium • May 1998

Weighting Observations: The use of Kinematic Models in Object Tracking

Kevin Nickels
knickels@uiuc.edu

Seth Hutchinson
seth@uiuc.edu

Dept. of Electrical and Computer Engineering and The Beckman Institute
University of Illinois at Urbana-Champaign
Urbana, IL 61801

Abstract

We describe a model-based object tracking system that updates the configuration parameters of an object model based upon information gathered from a sequence of monocular images. Realistic object and imaging models are used to determine the expected visibility of object features, and to determine the expected appearance of all visible features. We formulate the tracking problem as one of parameter estimation from partially observed data, and apply the Extended Kalman Filtering (EKF) algorithm. The models are also used to determine what point feature movement reveals about the configuration parameters of the object. This information is used by the EKF to update estimates for parameters, and for the uncertainty in the current estimates, based on observations of point features in monocular images.

1 Introduction

In this paper we describe a model-based object tracking system that updates the configuration parameters (state) of an object model based upon information gathered from a sequence of monocular images of a robotic arm. We formulate the tracking problem as one of parameter estimation from partially observed data, and apply the EKF algorithm to this problem.

The use of complex explicit kinematic models in object tracking has increased in recent years [5] [7]. When tracking a known three dimensional object, the use of an explicit object model enables an ongoing assessment of the usefulness of each tracked feature in computing each internal degree of freedom of an object. Explicit models also allow feature measurements to decrease the uncertainty in the estimation of internal degrees of freedom of an object. This reduced uncertainty can then be used to aid in disambiguating other measurements. Our work uses a robotic arm exercising three internal degrees of freedom. This work demonstrates that object tracking with explicit models of objects with complex geometry is feasible, even for objects with many internal degrees of freedom.

Previous work has assumed a constant set of visible features [10], or the use of features with simple primitives (edges or corners) [5]. Since we use realistic object and imaging models to compute the expected visibility of features, we can accommodate a set of features distributed about the object such that several will be visible in most configurations of the object, yet concentrate our search efforts during feature tracking on those features expected to be visible in the current (estimated) configuration of the object. Since we use object and imaging models to compute the expected appearance of features, this work can track features with more complex geometry, and therefore less ambiguity, than lines or edges. For example one finger of a gripper, or the edge of a gear linkage, could be used as basic features to track. This is in contrast to tracking edges or corners, then building up these tracking results into a complex feature afterwards.

The remainder of the paper is organized as follows. In Section 2 we describe our system for using kinematic and imaging models for object tracking. We begin in Section 2.1 by introducing our formalization for the tracking problem, and the notation used throughout the paper. In Section 2.2 we describe in more detail the methods used to calculate the expected feature appearance and visibility. We review in Section 2.3 some methods for feature tracking, and describe in Section 2.4 our method for extracting an estimate of the measurement error as well as the most likely location for a feature. All these pieces are consolidated in Section 2.5, where we present the relevant equations from the Extended Kalman Filtering literature, and describe our use of the filter. Finally, in Section 3 we present some experimental results for the case of a three degree of freedom robotic arm.

0-7803-4300-x-5/98 $10.00

2 System Overview

In this section, we present an overview of the tracking system. First, we present our formalization of the tracking problem. We next describe how our use of object kinematic and appearance models enables us to compute the expected appearance of each visible feature. We then describe the feature tracking process, where a comparison of portions of the actual input image to the expected feature appearance is made. We describe how an estimate of the uncertainty involved in the extraction of each feature is made. Finally, we present the relevant equations from Extended Kalman Filtering, and describe how our system uses this framework to update estimates for all internal degrees of freedom of an object.

2.1 Formalization and Notation

We wish to estimate the joint angles, $\mathbf{x}_k$ of a robotic arm, which we shall call the *state vector* of the system, at each time step k. In this paper, $\mathbf{x}_k \in \Re^3$. We will be modeling the $\mathbf{x}_k$ as a random vector and we associate a *covariance matrix*, P_k, with the state vector. We cannot observe the joint angles directly, but we can define a vector valued function $\mathbf{h}_k$ mapping joint space into observation space. We denote our observations by $\mathbf{z}_k$. Any bias about likely motion in joint space is defined in the system model $\mathbf{f}_k$, a vector valued function mapping the a-posteri state estimate at each time step into the a-priori state estimate at the next time step.

Certain parameters will be conditioned on the number of observations made. We denote this with a subscript. Thus we use the notation $\hat{\mathbf{x}}_{k|k-1}$ to represent the optimal estimate for the vector $\mathbf{x}_k$, given $\mathbf{z}_0 \ldots \mathbf{z}_{k-1}$. Similarly, $P_{k|k}$ denotes the covariance of $\mathbf{x}_k$ given $\mathbf{z}_0 \ldots \mathbf{z}_k$. We often use P_k interchangeably with $P_{k|k}$. This notation is standard in the Kalman Filtering literature.

In summary, we assume a fairly standard nonlinear system model,

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k) + G'_k(\mathbf{x}_k)\mathbf{w}_k \quad (1)$$

$$\mathbf{z}_k = \mathbf{h}_k(\mathbf{x}_k) + \mathbf{v}_k, \quad (2)$$

where $\mathbf{f}_k$ and $\mathbf{h}_k$ are vector valued functions with ranges of dimension n and q respectively, and G'_k is a matrix valued function of dimension $n \times p$. We make the usual (in Kalman Filtering literature) assumptions with respect to the correlation of the noise ($\mathbf{w}_k$ and $\mathbf{v}_k$) and initial conditions ($\mathbf{x}_0$):

$$E(\mathbf{w}_k\mathbf{w}_l^T) = Q_k\delta_{kl} E(\mathbf{v}_k\mathbf{v}_l^T) = R_k\delta_{kl}$$
$$E(\mathbf{w}_k\mathbf{v}_l^T) = E(\mathbf{w}_k\mathbf{x}_0^T) = E(\mathbf{v}_k\mathbf{x}_0^T) = 0.$$

In terms of our tracking problem, $\mathbf{h}_k(\mathbf{x}_k)$ contains the image plane coordinates of each feature when the arm is at configuration $\mathbf{x}_k$. The system function $\mathbf{f}_k$ could be used to implement a motion model. For example constant velocity motion in joint space could be assumed, if this type of motion was expected to be seen often. In this case, $\hat{\mathbf{x}}_k$ would contain estimates for the angular velocities of each angle as well as estimates for the joint angles.

2.2 Expected Feature Appearance and Visibility

Our work uses a realistic model for the robotic arm, both in terms of kinematics and appearance. A perspective imaging model is used for the camera. At each time step k in the tracking process, the optimal state estimate given the observations from time 0 to $k-1$, $\hat{\mathbf{x}}_{k|k-1}$, is combined with these models (using the OpenGL graphics language) to generate a synthetic image of the scene. One such scene is shown in Figure 1. The system does not use texture mapping, but simulates the effect of multiple overhead lights shining on a relatively non-reflective surface. The background is approximated by a constant-color surface, with the same lighting model. For all surfaces, Gouraund shading is used to interpolate the color of interior of each polygon based on lighting calculations done at the vertices of the polygon. Multiple computations are based on this image, as described below.

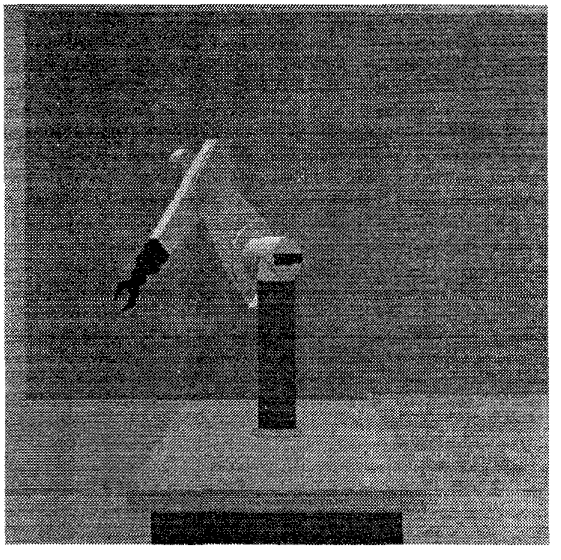

Figure 1: A Synthetic Image

Since the object models, camera imaging models, and assumed illumination of the scene are used to create the feature templates, errors in these models will effect the feature tracking to the extent that the feature templates are incorrect. The specific effect of errors in these models, as well as the use of even more sophisticated rendering techniques, is a topic for further investigation.

For each feature, the forward kinematics of the robot are used to compute the 3D location of that feature, and its 2D projection onto the image plane. If the 2D projection is not contained in the portion of the image plane actually observed, that feature is deemed not visible. During the rendering process, the depth of the closest object to the image plane at each pixel is recorded in a special buffer termed the *z-buffer*. This is a well known method for 3D rendering in computer graphics. For each feature, the depth recorded in the z-buffer is compared against the computed depth for that feature to determine if that feature is visible at the given configuration. A feature is visible in a given configuration if it's depth is equal to that recorded in the appropriate portion of the z-buffer.

For each feature that is expected to be visible by the above criterion, we record its expected appearance for use as a template to compare portions of the input image against during the feature tracking phase of the tracking. This is done by saving a region of the synthetic image about the feature's projection onto the image plane. This enables us to use arbitrary points on the surface of the object as features, and allows the use of the same framework whether the feature is a line, a spot, a corner, or the center of the letter P.

2.3 Feature Tracking

Feature detection can be accomplished by comparing an image region against a template for the feature, and using some metric, usually the Sum of Squared Differences (SSD), to rate the similarity. A good review of this technique is given in [3]. The main problem with the naive approach is that the simple template is a static 2D entity, and the image patch in a dynamic scene may undergo transformations that the template can not model.

A slightly more complex algorithm that also works in certain situations is to use an image patch from the previous image, taken from the area around the last computed position of the feature in that image, for the template. The main difficulty of this approach is feature skew, where the template slowly stops tracking the feature of interest and creeps onto another feature. The next step in sophistication is to have a template that can model more general transformations than the simple template, such as affine distortion.

An even more sophisticated template might have a 3D registered texture as part of the template, and use the predicted position of the object, along with computer graphics techniques, to render the relevant portion of the scene, complete with sophisticated texture mapping techniques, and estimate the appearance of a feature in the image for use as a template to match against [4] [8].

In our work, we presently use a basic SSD similarity measure, in conjunction with a template generated from the synthetic image as described in Section 2.2. In the current implementation, fixed size templates of size 13×13 are generated. A fixed rectangular area (size 25×25) of the image, centered about the predicted feature location, is compared with this template.

2.4 Measurement Error

One advantage to SSD based feature detection is that in addition to an extracted feature location, some knowledge of the "goodness of fit" of that feature to the surrounding image is returned, in the form of an SSD surface [1]. The local shape of the SSD surface can tell in which directions the template is a good match with the local image content, and in which directions the template differs greatly from the local image content [10].

If we interpret the feature location as a two dimensional random vector, the surface returned by an SSD measurement can be normalized and treated as a probability measure on the image plane position of the feature. Since the matching properties of the SSD measurement are only valid within some region of the feature location in the image, this interpretation and normalization needs to be a local one. Away from the feature location, other features interfere with the matching process, and the match score no longer reflects the spatial uncertainty for the given feature.

The mode, or most probable value, of the random vector is located at the peak of the SSD surface. We take this as the feature location measurement, $\mathbf{z}_k$. Assuming a symmetric unimodal random vector (for the purposes of the EKF, the random vector is assumed to be Gaussian), contours of equal probability surround the mode.

It has been noted [12] that the covariance matrix of a random vector determines the shape and orientation of these contours (in the noiseless case, they are ellipsoids) of constant probability. Kosaka and Kak [6] provide an in-depth discussion on the equivalence

between a covariance matrix and the related error ellipsoids. The orientation of the semi-major axes of the error ellipsoids determine the eigenvectors of the covariance matrix. The lengths of the semi-major axes determine the eigenvalues of the covariance matrix.

The variance of u (σ_u^2), the variance of v (σ_v^2), and the covariance between u and v ($\rho_{uv}\sigma_u\sigma_v$) can be estimated directly from the (scaled) SSD surface [11], yielding the desired covariance matrix,

$$R_k = \begin{bmatrix} \sigma_u^2 & \rho_{uv}\sigma_u\sigma_v \\ \rho_{uv}\sigma_u\sigma_v & \sigma_v^2 \end{bmatrix}, \tag{3}$$

which contains complete information about the *orientation* and *shape* of the error ellipsoids.

To date, researchers have used the local shape of the SSD surface only as a confidence measure, and those researchers have only looked at the curvature along the image plane axes [1] or the image plane axes and 45° lines in the image plane [10]. More importantly, the curvature estimates are used only as a computation of the certainty of the match between the template and the local image structure, not as an uncertainty measurement on the location of the match.

Retaining only uncertainty measurements along image plane axes is equivalent to recording σ_u and σ_v. Thus, the implicit assumption made by these researchers is that the contours of equal probability have their semi-major axes aligned with an image plane axis or a 45° line in the image plane. By computing the covariance as well as the variances, we retain information about the orientation of the ellipsoids of constant probability, as well as their intersection with the u and v axes. Therefore, we gain the ability to maintain information about directions of good spatial discrimination even if that direction of that discrimination doesn't happen to be aligned with the (u, v) image plane axes.

2.5 Extended Kalman Filtering

Extended Kalman Filtering is an extension of the classic Kalman Filtering algorithm to the nonlinear case. Kalman Filtering computes the optimal linear least squares solution to observations of a linear system corrupted by white Gaussian noise [2]. The following equations can be derived from the linear Kalman Filter by linearizing the system function $\mathbf{f}_k$ and the observation function $\mathbf{h}_k$ about the optimal state estimate $\hat{\mathbf{x}}_{k|k-1}$. This set of equations, collectively defining a recursive estimation algorithm, is generally collectively called the *Extended Kalman Filter*:

$$P_{k,k-1} = \left[\frac{\partial \mathbf{f}_{k-1}}{\partial \mathbf{x}_{k-1}}(\hat{\mathbf{x}}_{k-1})\right] P_{k-1,k-1} \left[\frac{\partial \mathbf{f}_{k-1}}{\partial \mathbf{x}_{k-1}}(\hat{\mathbf{x}}_{k-1})\right]^T + G_{k-1}(\hat{\mathbf{x}}_{k-1}) Q_{k-1} G_{k-1}^T(\hat{\mathbf{x}}_{k-1}) \tag{4}$$

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{f}_{k-1}(\hat{\mathbf{x}}_{k-1})$$

$$P_{k,k}^{-1} = P_{k,k-1}^{-1} + \left[\frac{\partial \mathbf{h}_k}{\partial \mathbf{x}_k}(\hat{\mathbf{x}}_{k|k-1})\right]^T R_k^{-1} \left[\frac{\partial \mathbf{h}_k}{\partial \mathbf{x}_k}(\hat{\mathbf{x}}_{k|k-1})\right] \tag{5}$$

$$K_k = P_{k,k} \left[\frac{\partial \mathbf{h}_k}{\partial \mathbf{x}_k}(\hat{\mathbf{x}}_{k|k-1})\right]^T R_k^{-1}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + K_k(\mathbf{z}_k - \mathbf{h}_k(\hat{\mathbf{x}}_{k|k-1})), \tag{6}$$

with initial conditions $P_{0,0} = Var(\mathbf{x}_0)$, and $\hat{\mathbf{x}}_0 = E(\mathbf{x}_0)$.

As described above, our system uses realistic object and imaging models to define $\mathbf{h}_k$. The system Jacobian, $\frac{\partial \mathbf{h}_k}{\partial \mathbf{x}_k}$, is computed algebraically. As we currently use a constant position motion model (all movement is modeled as random noise $\mathbf{w}_k$ injected into the state vector), $\mathbf{f}_k = I$, and $\frac{\partial \mathbf{f}_{k-1}}{\partial \mathbf{x}_{k-1}} = 0$. Currently, Q_k is a constant diagonal matrix, and the computation of R_k is described above.

3 Tracking Results

In this section, we will present some tracking results for the case of a three degree of freedom arm being observed by a 2D sensor. In order to concentrate on the effect that the object kinematics have on the tracking algorithm, no motion model is used. In all cases, the 512×485 input image is subsampled by a factor of three before tracking. Joint speeds are restricted so that features do not escape the fixed search areas.

Joints 0, 1, and 2 are the first three joints of a Unimation PUMA robotic arm, as assigned by the standard Denevit-Hartenberg parameters. In the first experiment, the arm moves from a parking position down towards the table, as if picking up a block from the table. The arm then moves up away from the table, swings right (from the point of view of the camera), and moves down toward the table again. All three joints move significantly, in constant velocity motion in joint space. Note that this fact is not exploited at the current time. The actual joint angles commanded, and the estimation of the joint angles by our filter, are shown in Figure 2. The error in the tracking is shown in Figure 3.

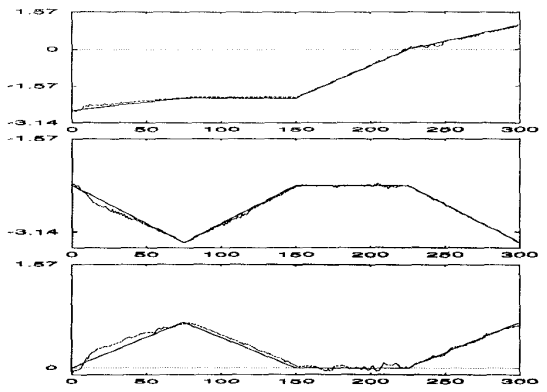

Figure 2: Exercising three degrees of freedom. Top: x_0 and $\hat{x}_0$, Middle: x_1 and $\hat{x}_2$, Bottom: x_2 and $\hat{x}_2$

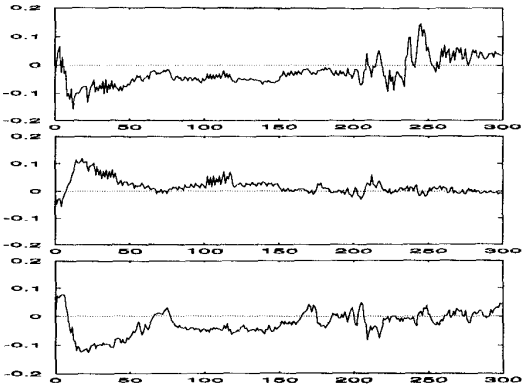

Figure 3: Tracking Error. Top: $x_0 - \hat{x}_0$, Middle: $x_1 - \hat{x}_1$, Bottom: $x_2 - \hat{x}_2$

There are 11 features more or less evenly distributed about the arm. Only a subset of these are visible at any given time. We described in Section 2.2 how we determine the expected visibility of a feature. For the same experiment described above, Figure 4 illustrates the visibility of features throughout the experiment. The visibility of feature 0 is depicted on the top row, feature 11 on the bottom. Each time step is a column of Figure 4. Thus, a dark rectangle in a column indicates that a feature is visible at a particular time step, and a light rectangle indicates that a feature is not visible at that time step.

One feature of the system is the characterization and use of the uncertainties in the kinematic and imaging chain, as modeled by our object and imaging models. See [9] for an in-depth discussion on this. Figure 5 illustrates the uncertainty present in the system dur-

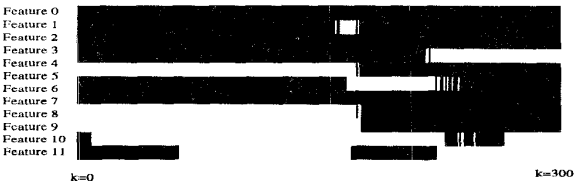

Figure 4: Feature Visibility

ing the experiment described above. The variance of the estimate for each joint angle is shown. The time scale is the same as in Figure 2. Note particularly the behavior around time step 230 - 240. In this region, the arm is approximately parallel to the image plane. Also in this region, there is an increase in uncertainty for x_0, and a decrease in the uncertainty for the estimates for x_1 and x_2. Given the known kinematic models, this result corresponds to the well known fact that for a perspective imaging model, changes in depth are not easily observable, while changes in the two spatial dimensions parallel to the image plane are easily observable.

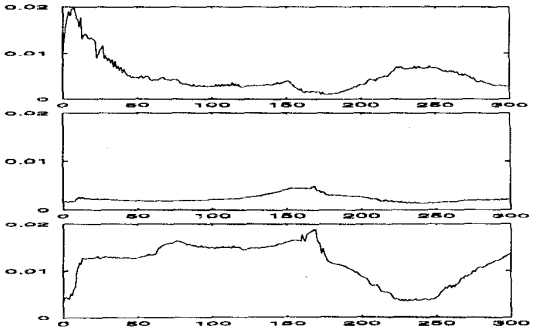

Figure 5: Uncertainty in the tracking system. Top: $P[0,0]$, Middle: $P[1,1]$, Bottom: $P[2,2]$

This change in uncertainty can be predicted by looking at the behavior of the system Jacobian in these regions. In this region, the system Jacobian, $\frac{\partial \mathbf{h}_k}{\partial \mathbf{x}_k}$, becomes ill-conditioned with respect to x_0, and well-conditioned with respect to x_1 and x_2. That is, measurements from the features do not change with incremental changes in x_0, and are therefore effectively ignored when updating x_0 in (6). Measurements from the features do, however, change significantly with changes in x_1 or x_2, and are therefore used when updating x_1 and x_2 in (6). Observing (4) and (5), we see that for state elements with low update weights in $\frac{\partial \mathbf{h}_k}{\partial \mathbf{x}_k}$, the time update increases the uncertainty in the system (corresponding to an increase in the uncertainty of our estimates as time advances), but we get

no corresponding decrease in the uncertainty from the measurement update.

This selective weighting of image data is a feature of our tracking system. By analyzing the kinematic models, we can evaluate feature extraction results in the proper context, instead of assigning the same validity to all extractions, regardless of their usefulness in tracking the parameters of interest (the joint angles, in this case). This selective weighting of features can be seen in Figure 6, where we illustrate the average feature weighting factor for each state element, as a function of time. In this figure, a value of 0 indicates complete disregard of the measurement update computed by the feature extraction measurement, and figures away from zero indicate different levels of confidence in the updates. In the region mentioned above, note that the weight assigned to the updates for x_1 and x_2 increase, and the weight assigned to the update for x_0 decreases toward zero, as described above.

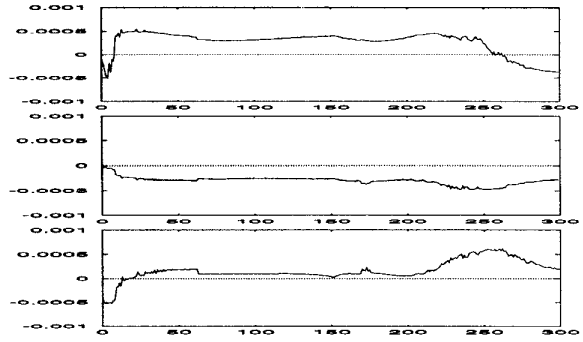

Figure 6: Update Weights (Average over all features). Top: x_0, Middle: x_1, Bottom: x_2

4 Conclusions

We have described how to characterize the uncertainty in data observations in terms of both the perceptibility of feature motion and the quality of feature extraction. We have incorporated these characterizations into an EKF formalism, and presented preliminary results for the case of 3D tracking. We are currently extending our results to a variety of more complicated problems, addressing such issues as occlusion, higher dimensional systems, tracking more complicated (possibly non-rigid) objects, and tracking objects with certain unknown kinematic parameters, such as link lengths.

References

[1] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *Int'l J. Comuter Vision*, 2:283–310, 1989.

[2] R. G. Brown. *Introduction to Random Signal Analysis and Kalman Filtering.* John Wiley & Sons, New York, NY, 1983.

[3] R. Brunelli and T. Poggio. Face recognition: Features versus templates. *IEEE Transactions on pattern alanlysis and machine intelligence*, pages 1042–1052, October 1993.

[4] G. Hagar and P. Belhumeur. Real-time tracking of image regions with changes in geometry and illumination. In *Proc. 1996 Conf. Computer Vision and Pattern Recognition*, pages 403–410, 1996.

[5] A. Hauck, S. Lanser, and C. Zierl. Hierarchical recognition of articulated objects from single perspective views. In *Proc. 1997 Conf. Computer Vision and Pattern Recognition*, pages 870–883, 1997.

[6] A. Kosaka and A. C. Kak. Fast vision-guided robot navigation using model-based reasoning and prediction of uncertainties. *Comput. Vis. Image Understanding*, pages 271–329, November 1992.

[7] John E. Lloyd, Jeffrey S. Beis, Dinesh K. Pai, and David G. Lowe. Model-based telerobotics with vision. In *International Conf. Robotics AutomationAlbequerque, NM*, April 1997.

[8] R. Lopez, A. Colmenarez, and T. S. Huang. Vision-based head and facial feature tracking. In *Advanced Displays and Interactive Displays Federated Laboratory Consortium, Annual Symposium*, January 1997.

[9] K. Nickels and S. Hutchinson. Characterizing the uncertainties in point feature motion for model-based object tracking. In *Proc. Workshop on New Trends in Image-Based Robot Servoing, Grenoble, France*, pages 53–63, 1997.

[10] N. P. Papanikolopoulos. Selection of features and evaluation of visual measurements during robotic visual servoing tasks. *Journal of intelligent and robotic systems*, pages 279–304, July 1995.

[11] A. Singh and P. Allen. Image flow computation: An estimation-theoretic framework and a unified perspective. *Comput. Vis. Image Understanding*, pages 152–177, September 1992.

[12] Ali Zolghadri. An algorithm for real-time failure detection in Kalman filters. *IEEE Trans. on Automatic Control*, 41(10), October 1996.