

Object Tracking using Affine Multiple Views Geometry

*A thesis submitted in partial fulfillment of
the requirements for the degree of*

Bachelor of Technology
in
Computer Science and Engineering

Gurmeet Singh Manku

Himanshu Nautiyal

Supervisor : **S. Banerjee**



Department of Computer Science and Engineering
Indian Institute of Technology, Delhi

May 1995

CERTIFICATE

This is to certify that the thesis entitled **Object Tracking using Affine Multiple Views Geometry** being submitted by Gurmeet Singh Manku and Himanshu Nautiyal in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering, at the Indian Institute of Technology, Delhi, is a bonafide record of the project work done by them.

This project was carried out under my supervision and the matter and results embodied in this thesis are original and have not been submitted elsewhere, wholly or in part, for the award of any degree or diploma, or for any other purpose.

Dr. S. Banerjee

Department of Computer Science and Engineering
Indian Institute of Technology
Delhi

ACKNOWLEDGMENTS

We are deeply indebted to our supervisor **Dr. S. Banerjee** for the continuous encouragement, guidance and support he provided during the course of this project. His unflagging zeal and superlative technical advice have been a source of both inspiration and help to us during our work.

We would also like to express our gratitude to Dr. S. D. Joshi who spent some of his valuable time in explaining the theory of Kalman filtering to us.

We are grateful to our colleague, Mr. A. Bhatia. who helped us in no small measure with some of the problems that arose during the project.

Thanks are also due to Mr. Jaswant, Mr. Prasad and Mr. Kaushik for the cooperation we received from them regarding laboratory activity and facilities.

Gurmeet Singh Manku

Himanshu Nautiyal

Abstract

Tracking and extraction of structure and motion parameters have been treated in the past as two separate problems. Our work attempts to unify them and provides solutions to both, each of which aids the other.

Traditional approaches to tracking, most of which involve Kalman filtering, have been based only on intensity correlation matching. Little attempt has been made to impose structural constraints on the features being tracked. And the possibility of extracting information from multiple views has been largely ignored.

Recent work in the direction of computing invariants to represent structural information has motivated the inquiry into structure-based tracking schemes. This report develops one such scheme, using the constraints imposed by affine geometry, valid under the simple assumptions that the camera model is affine and that the object being tracked is undergoing affine motion.

Experimentation with real image sequences has been carried out, followed by comparisons with the Kalman filtering algorithm. The results clearly indicate that the new approach is more robust and brings about reduction in the error involved in computing the affine structure of the object in motion.

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Aims and Objectives	1
1.3	Feature detection	2
1.4	Structure of the report	3
2	Kalman Filtering	4
2.1	Kalman Filter Equations	4
2.2	Application in Tracking	5
2.3	Our Implementation	6
2.4	Parameters	6
2.5	Algorithm	7
2.6	Performance of Kalman filtering	8
3	Affine Multiple Views Geometry	9
3.1	Why the affine camera model?	9
3.2	Affine camera model: preliminaries	11
3.3	Invariance of coordinates in centered system	12
3.4	Issues arising from invariance of centered coordinates	13
4	Epipolar geometry	14
4.1	Epipolar line geometry	14
4.2	Mathematical formulation of epipolar line	15
4.3	The affine fundamental matrix	17
4.4	Solution of the epipolar equation	18
4.4.1	Choice of cost function	19
4.4.2	Use of the solution in algorithm	19
5	Affine structure from motion	21
5.1	Koenderink and van Doorn	21

5.2	Tomasi and Kanade	23
5.2.1	Our implementation of the factorization method	23
5.2.2	Analysis of the implementation	24
6	Overall Algorithm	25
6.1	Matching algorithm	25
6.2	Gaze point computation	28
6.3	Universal Hashing for Neighborhood Search	29
7	Results and Analysis	30
7.1	Implementation Issues	30
7.2	Experimentation	31
7.3	Analysis	31
7.4	Conclusions	32
8	Future Work	49
A	Best epipolar line fit	50
A.1	Statement of the problem	50
A.2	Solution and proof	50
B	Least Square Solutions	52
B.1	Error in only one variable	52

List of Figures

3.1	Perspective camera geometry : Epipolar line constraint	10
3.2	Concurrency of epipolar lines in perspective epipolar geometry.	11
4.1	Affine camera geometry : All projection rays are parallel	16
4.2	Parallelism of epipolar lines in affine epipolar geometry	17
5.1	Koenderink and van Doorn's determination of structure	22
7.3	Total number of matches in nod1.	36
7.4	Number of matches reported by Kalman filter in nod1.	36
7.5	Average age of matched corners in nod1.	36
7.6	Number of corners for affine structure computation in nod1.	36
7.7	Average pixel error in affine structure computation	37
7.8	X and Y coordinates of gaze point in affine-structure based method. .	37
7.9	X and Y coordinates of gaze point using only Kalman filtering.	37
7.10	Gaze demand in nod1.	37
7.18	Total number of matches in rand1.	45
7.19	Number of matches reported by Kalman filter in rand1.	45
7.20	Average age of matched corners in rand1.	46
7.21	Number of corners for affine structure computation in rand1.	46
7.22	Average pixel error in affine structure computation	47
7.23	X and Y coordinates of gaze point in affine-structure based method. .	47
7.24	X and Y coordinates of gaze point using only Kalman filtering.	48
7.25	Gaze demand in rand1.	48

Chapter 1

Introduction

In the paradigm of *active vision* [AB87], an observer makes deliberate movements to obtain a succession of images to accomplish some given visual task. The recent interest in the development of active vision systems has arisen because of the fact that most vision tasks which are ill-posed in snap-shot situations, become better constrained for an active observer. The need for tracking arises both as a vision task and as a primary function of the system to enable other tasks which might require a prolonged view of an object.

1.1 Motivation

Till the present, practically implemented systems have tended to concentrate on intensity correlation-based methods like Kalman filtering [RM93]. These methods are inadequate in that they do not take into account the structure of the object of interest and also in that they have been based solely on intensity correlation techniques. However in many of the applications of tracking e.g. autonomous guided vehicles, automatic path planning, surveillance systems and goal directed analysis of a scene, it is also important to derive the structure of the object being tracked. This motivates the quest to carry out the tracking and the determination of structure concurrently and in a cooperative manner. A similar approach has recently been implemented and investigated, as reported in [SBZ93].

1.2 Aims and Objectives

Our project aims to

- Implement some known strategies for correlation based tracking.

- Devise and implement schemes for affine structure based tracking.
- Analyze the performance of the structure based tracking strategies *vis-a-vis* the traditional correlation based strategies.

1.3 Feature detection

Any scheme for tracking would require the detection of certain features which would persist across frames. These features could be corners, contours or any other feature deemed suitable. However, for any real-time tracking system any features other than corners would demand prohibitive computation. Corners are few in number as compared to the total image size. Therefore, keeping feasibility in real-time into account, our work is corner-based.

Obtaining the corners requires the use of a corner detection operator. For our implementation, we chose the one given by Wang and Brady [WB92, WB91]. Their operator for corner detection uses the fact that the total curvature of the image surface is proportional to the second order derivative in the direction of the tangent, and also inversely proportional to the edge strength, *i.e.* the magnitude of the first derivative. They defined their operator as

$$\left\{ \begin{array}{l} \Gamma = \left(\frac{\partial^2 F}{\partial t^2}\right)^2 - S |\nabla F|^2 = \text{maximum} \\ \frac{\partial^2 F}{\partial n^2} = 0 \\ |\nabla F|^2 > T_1, \quad \Gamma > T_2 \end{array} \right.$$

where S is a constant measure of image surface curvature, dependent on the differentiation masks used, F is the image intensity after Gaussian smoothing, and T_1 and T_2 are experimentally determined thresholds.

Although Wang and Brady have shown that Gaussian smoothing causes the displacement of corners from the true positions, a further result by Wang states that such a smoothing with $\sigma = 0.5$ reduces noise to a significant extent, thus increasing corner reliability, and displaces the corners by less than half a pixel.

Any corner detector has to operate under two mutually contradicting requirements, *detection* and *localization*. The criterion of detection requires that all actual corners be detected by the operator and only these be detected while that of localization requires that all detected corners be placed by the detector at locations “close” to the actual ones. If the preliminary Gaussian smoothing is done with a small σ , then noise remains dominant and though the localization of the actual corners is good, lots of spurious corners are detected due to noise. If the σ is large, however, noise effects

are removed, thus leading to better detection but the localization deteriorates. Any operator has to achieve the right balance between these requirements.

In view of the above, any corner based tracking method must be robust with respect to the localization and detection errors of the corner detectors, which it invokes. The primary concern of this work is to design such robust tracking strategies.

1.4 Structure of the report

The organization of the report is as follows.

The motivation and aims of our project are identified in this introduction.

Chapter 2 presents a synopsis of the theory of Kalman filtering and its application to tracking along with a description of its performance.

Chapter 3 deals with the affine camera model - the reasons behind its choice for implementing the structure based schemes and the development of the mathematical background for it.

Chapter 4 describes the geometry of the epipoles and epipolar lines and how these can be employed for reduction of the search space in tracking. Included in this chapter is a brief overview of the method of determining the best-fit epipolar line.

Chapter 5 reviews the approaches suggested by Koenderink and van Doorn [KvD91] and by Tomasi and Kanade [TK92] for deducing affine structure from motion.

Chapter 6 contains the description of the overall algorithm that we have implemented to some level of detail. This chapter also describes the hashing scheme that has been used in the match searches.

Chapter 7 discusses the experimentation on the implementations of both the strategy based only on Kalman filtering and of that proposed in this report. The findings and their implications are set out here.

Chapter 8 suggests the further work possible after this project.

We round off with a summary of the conclusions of our implementation and experimentation and its significance to the choice of tracking methods, followed by details of the mathematical ideas used in the project in the appendices and a list of references.

Chapter 2

Kalman Filtering

An important problem in signal processing and control theory is to *track* (maintain an estimate of) a time-varying signal in the presence of noise. If a signal is known to be characterized by some number of parameters that vary only slowly, then the formalism of *Kalman filtering* tells how the incoming raw measurements of signal should be processed to produce the best parameter estimates as a function of time.

2.1 Kalman Filter Equations

Kalman filters [BSF88] comprise a class of linear unbiased minimum-error covariance sequential state estimation algorithms. Let the time-varying state vector we wish to estimate be $\mathbf{x}(k)$, where k is the time step. Assume that the variation over time is modeled by

$$\mathbf{x}(k+1) = F(k)\mathbf{x}(k) + G(k)\mathbf{u}(k) + \mathbf{v}(k),$$

where $F(k)$ and $G(k)$ are time-varying matrices and $\mathbf{u}(k)$ is an input or control vector and $\mathbf{v}(k)$ is zero-mean, temporally uncorrelated Gaussian noise with covariance matrix $Q(k)$. The initial estimates, $\hat{\mathbf{x}}(0|0)$ of the state vector and $P(0|0)$ of the state covariance are known *a priori*, where $(k_1|k_2)$ denotes an estimate at time step k_1 , given the evidence upto time step k_2 . Also, we assume that $\mathbf{z}(k)$, our observation of $\mathbf{x}(k)$, is noisy and modeled by

$$\mathbf{z}(k) = H(k)\mathbf{x}(k) + \mathbf{w}(k),$$

where \mathbf{w} is zero-mean, temporally uncorrelated Gaussian noise with covariance $R(k)$.

Using the same notation as used in [BSF88], one iteration of the Kalman filter algorithm at time step $k+1$ is given by

Prediction

State prediction :

$$\hat{\mathbf{x}}(k+1|k) = F(k)\hat{\mathbf{x}}(k|k) + G(k)\mathbf{u}(k)$$

Covariance prediction :

$$P(k+1|k) = F(k)P(k|k)F(k)^T + Q(k)$$

Filtering

State update :

$$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + W(k+1)v(k+1)$$

Covariance update :

$$P(k+1|k+1) = P(k+1|k) - W(k+1)S(k+1)W(k+1)^T,$$

where

Gain matrix :

$$W(k+1) = P(k+1|k)H(k+1)^T S(k+1)^{-1},$$

State innovation :

$$v(k+1) = \mathbf{z}(k+1) - H(k+1)\hat{\mathbf{x}}(k+1|k) \text{ and}$$

Innovation covariance :

$$S(k+1) = H(k+1)P(k+1|k)H(k+1)^T + R(k+1)$$

2.2 Application in Tracking

To be able to deduce the structure of any object in the field of view, it is essential to solve the corner correspondence problem across successive frames. This can be achieved by using a constant *image velocity* Kalman filter for each corner being tracked [RM93].

The state vector for any corner, $\mathbf{x}(k)$ is given by a 4 x 1 matrix $[x(k), y(k), \dot{x}(k), \dot{y}(k)]^T$. State transition is modeled by $\mathbf{x}(k+1) = F\mathbf{x}(k) + \mathbf{v}(k)$, where $\mathbf{v}(k)$ are iid Gaussian zero-mean, temporally uncorrelated noises with covariance Q , and

$$F = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Measurement is noisy and given by a 2 x 1 vector $\mathbf{z}(k) = H\mathbf{x}(k) + \mathbf{w}(k)$, where $\mathbf{w}(k)$ are iid Gaussian, zero-mean, temporally uncorrelated noises with covariance R , and

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

2.3 Our Implementation

Assuming that the motion of the object is small and smooth, the *constant image-velocity* Kalman filter, as described above, will work well only if the motion of the camera itself is small and without jitter.

We have carried out experimentation on image sequences taken with a static camera. Each image has size 288 x 384 pixels. However, we assume that *our* camera can see only a smaller sub-window, a fovea of size 200 x 200 and possesses degrees of freedom in the x - and y - coordinates. Carrying over the theory to an actual camera, with degrees of freedom along elevation and vergence axes, is straightforward.

The Kalman filter we have used is of the *constant image-velocity* type, if looked upon in the coordinate system of the real camera. However, if we work with the coordinate system of the fovea, we need to introduce a control vector in the state prediction equation to account for *our* camera's ego-motion. The equation then becomes,

$$\mathbf{x}(k+1) = F\mathbf{x}(k) + \mathbf{v}(k) - \mathbf{u}(k),$$

where $\mathbf{u}(k)$ is our camera's displacement from time step $k-1$ to k (in either coordinate system).

2.4 Parameters

We assume that the distribution of $\mathbf{v}(k)$ is time-invariant and has covariance given by

$$Q = \begin{bmatrix} 5.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 5.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 5.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 5.0 \end{bmatrix}$$

Similarly, we assume that the distribution of $\mathbf{w}(k)$ is independent of time, and has covariance given by

$$R = \begin{bmatrix} \sigma_{corner} & 0.0 \\ 0.0 & \sigma_{corner} \end{bmatrix}$$

where σ_{corner} is the value of σ used for Gaussian smoothing before the Plessey corner detection stage. In case no smoothing is carried out, a reasonable estimate of R is :

$$R = \begin{bmatrix} 2.0 & 0.0 \\ 0.0 & 2.0 \end{bmatrix}$$

The values for R are based on a result that the variance of error in corner displacement is the same as that used in Gaussian smoothing before corner detection [WB91].

2.5 Algorithm

While attempting to match corners, there are two kinds of corners in the previous frame : (a) those which have been matched (*i.e.* its Kalman filter has been initialized), and (b) those which are unmatched *i.e.* pending. The overall algorithm for matching is :

For all matched corners :

- Predict the new position of each corner, using the Kalman filter prediction equations and define a search area, whose size depends upon the predicted error covariance matrix P .
- Search the area for corners and make a list of candidates.
- For each candidate, carry out a cross-correlation with the corresponding neighborhood (size is user-defined, typically 5x5) of the original corner.
- Select the candidate having the largest correlation value, above a certain minimum threshold (user-defined, typically 0.70).
- If a candidate was selected, the corner as been matched. Update its Kalman filter.
- If no candidate was finally selected, delete this corner from the list of matched corners.

For all pending corners :

- Predict the position of each corner, assuming it to be static and taking the camera's ego-motion into account.
- Search a neighborhood of the predicted position for candidates. (Its size is user-defined and depends upon the estimate of the maximum velocity of the corners that the system hopes to track).
- For each candidate, carry out a cross-correlation with the corresponding neighborhood (size is user-defined, typically 5x5) of the original corner.
- Select the candidate having the largest correlation value, above a certain minimum threshold (user-defined, typically 0.70).
- If a candidate was finally selected, initialize its state vector as

$$\mathbf{x}(1) = [x_1, y_1, (x_1 - x_0)/\Delta t, (y_1 - y_0)/\Delta t]^T,$$

where (x_0, y_0) is the predicted corner position and (x_1, y_1) is the actual position of the match.

2.6 Performance of Kalman filtering

Kalman filtering is a fairly justifiable technique to use in tracking applications and has, in practice, been the method of choice in most of the presently existing object tracking systems. However, it suffers from certain disadvantages that we proceed to describe below.

The essential process in Kalman filter based matching is a search for matches based on the correlation of intensities of features (corners in our case). This is also its fundamental weakness, since it requires some restriction on the extent of motion, and on the illumination conditions in which the object of interest is being viewed, for assumptions of this nature to hold.

A further shortcoming of Kalman filter based matching is the absence of use of a substantial part of the information that may be derived from the multiple views of the object being tracked. It is very rarely the case that a system is designed solely for the purpose of tracking. Most generally, the problem is to be able to recognize the object and tracking arises from the need to fixate gaze on the object for a sufficient length of time to be able to garner structural information about it. It would be desirable to be able to unify these two requirements of tracking and recognition by the deduction of structure which would be useful to both [SBZ93]. However, Kalman filtering tracks each feature individually, without regard to the interrelation of the features among themselves, i.e. the structure, even though this could (and has been shown to be able to) serve a useful purpose in tracking.

Chapter 3

Affine Multiple Views Geometry

For our implementation, we use the affine camera model and attempt to deduce the affine structure of the object of interest. We also assume that the world coordinate transformation of the object across successive frames can be represented by an affine transformation. These choices are justified in this chapter and the mathematical formulation of various aspects of the model is described.

3.1 Why the affine camera model?

Before any investigation in the field of vision is started, a choice of the camera model to be used must be made. This must take into account, most importantly, the fidelity of the model to reality, the extent of computation and complexity involved in using that model, and also the numerical stability of the methods to be used during the application of the model.

In our approach, we have decided to use the affine camera model instead of the more familiar perspective model. When the perspective effects are small, as it is expected to be for foveal tracking, the use of the perspective camera model leads to estimation of parameters which are inherently ill-conditioned. In such a case, the major problem with the perspective model is that it would require the calculation of the epipoles (illustrated in Figures 3.1 and 3.2) which tend to move away on the image plane and this is an ill-conditioned problem with very high errors possible in the computation. The compromise to this is offered by the affine camera model, which assumes all epipolar lines to be parallel and hence places the epipole at ∞ , or the *ideal point* (illustrated in Figures 4.1 and 4.2). Making this choice allows us to obviate the computation of the epipole and thus to impose well-posedness on the dependent calculations. In what follows, we describe the affine camera model and the affine multiple view geometry [SAB93].

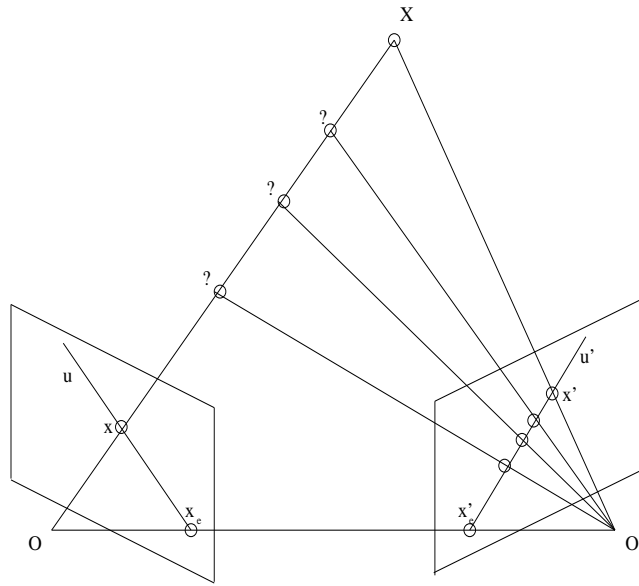


Figure 3.1: Perspective camera geometry : Epipolar line constraint

3.3 Invariance of coordinates in centered system

Consider four non-coplanar scene points $\mathbf{X}_0 \dots \mathbf{X}_3$ with \mathbf{X}_0 as the origin. Define the three *axis* vectors

$$\mathbf{E}_j = \mathbf{X}_j - \mathbf{X}_0 \quad (j = 1, 2, 3)$$

which span \mathcal{R}^3 . Thus each of the real world difference vectors $\Delta \mathbf{X}_i$ may be expressed as a linear combination of these E_j 's

$$\Delta \mathbf{X}_i = \alpha_i \mathbf{E}_1 + \beta_i \mathbf{E}_2 + \gamma_i \mathbf{E}_3 \quad (i = 0 \dots n - 1)$$

If the points \mathbf{X}_i now undergo the motion described by Equation 3.1 then we can say that

$$\Delta \mathbf{X}'_i = \mathbf{X}'_i - \mathbf{X}'_0 = \mathbf{A}(\mathbf{X}_i - \mathbf{X}_0) = \alpha_i \mathbf{A} \mathbf{E}_1 + \beta_i \mathbf{A} \mathbf{E}_2 + \gamma_i \mathbf{A} \mathbf{E}_3 \quad (3.3)$$

This motion would transform coordinate axes \mathbf{E}_j to \mathbf{E}'_j

$$\mathbf{E}'_j = \mathbf{X}'_j - \mathbf{X}'_0 = \mathbf{A} \mathbf{E}_j$$

From Equation 3.3, it is obvious that if we replace the old axes by the new ones the coefficients of the corresponding axes for the same difference vectors in the two images are the same i.e.

$$\Delta \mathbf{X}_i = \alpha_i \mathbf{E}_1 + \beta_i \mathbf{E}_2 + \gamma_i \mathbf{E}_3 \quad (3.4)$$

$$\Delta \mathbf{X}'_i = \alpha_i \mathbf{E}'_1 + \beta_i \mathbf{E}'_2 + \gamma_i \mathbf{E}'_3 \quad (3.5)$$

Thus the affine coordinates $(\alpha_i, \beta_i, \gamma_i)$ are *invariant* to the 3D affine transformation $\{\mathbf{A}, \mathbf{D}\}$. The physical reason for this is that the coordinates α, β and γ encode the constant 3D structure relative to a variable frame.

In summary: An object-centered reference frame $\{\mathbf{E}_1, \mathbf{E}_2, \mathbf{E}_3\}$ is chosen and every feature is assigned a 3D *affine coordinate* (α, β, γ) within that frame. When the object moves, it is sufficient to observe how the frame changes in order to know how all the other points move.

Since the difference image vectors are related to the world difference vectors by a simple matrix multiplication, i.e.

$$\Delta \mathbf{x} = \mathbf{M} \Delta \mathbf{X} \quad \text{and} \quad \Delta \mathbf{x}' = \mathbf{M}' \mathbf{A} \Delta \mathbf{X}$$

From what has already been done, it is easy to see that

$$\Delta \mathbf{x}_i = \alpha_i \mathbf{e}_1 + \beta_i \mathbf{e}_2 + \gamma_i \mathbf{e}_3 \quad (3.6)$$

$$\Delta \mathbf{x}'_i = \alpha_i \mathbf{e}'_1 + \beta_i \mathbf{e}'_2 + \gamma_i \mathbf{e}'_3 \quad (3.7)$$

where $\mathbf{e}_i = \mathbf{M}\mathbf{E}_i$ and $\mathbf{e}'_i = \mathbf{M}'\mathbf{E}'_i = \mathbf{M}'\mathbf{A}\mathbf{E}_i$. Thus, the same kind of conclusion holds true in the image too, viz. if we know the movement in the frame with respect to which the coordinates α, β and γ are assigned, we can predict the new image coordinates of all the features and the coordinates assigned remain the same for each feature if the old 3-axis set is replaced by a new 3-axis which corresponds to the previous one in the older image.

3.4 Issues arising from invariance of centered coordinates

Here, we are describing the image structure by means of *three* image axes although two would be sufficient, since it lies on a plane. However, the use of the additional axis enables us to compute *true affine coordinates* of the feature points which are invariant.

Another important point is that \mathbf{A} and \mathbf{D} , which fix the 3D frame *cannot* be recovered. Thus the correct 3D scene structure is said to have been extracted up to an arbitrary 3D affine transformation.

For the computation of the affine structure, two images with at least four points in correspondence, which create the basis, are required. After the basis or spanning set has been established, every additional point gives 4 equations in 3 unknowns.

$$\begin{bmatrix} \Delta x_i \\ \Delta x'_i \end{bmatrix} = \begin{bmatrix} e_1 & e_2 & e_3 \\ e'_1 & e'_2 & e'_3 \end{bmatrix} \begin{bmatrix} \alpha_i \\ \beta_i \\ \gamma_i \end{bmatrix} \quad (3.8)$$

Clearly the matrix containing the axes should have rank 3. If the rank is 2, it implies a planar object or that the object *appears* to be planar e.g. in a rotation about the optical axis. The rank being 4 is an indication of the fact that the affine camera model is not valid or that the motion cannot be represented by an affine transformation. Thus, if our assumptions of affineness are not valid in the real world, the algorithm will be able to detect the fact.

From the theory already discussed, equations for image transfer and for the dynamic change of basis during processing (this may be required, say when one of the 4 features is occluded) are easily deduced.

Chapter 4

Epipolar geometry

It is intuitively clear that among two images of the same object with some difference in either or both camera or object positions between them, the specification of one point in one image constrains the corresponding point in the other image in some manner. This notion of constraint can be formalized as the epipolar line.

This constraint has been used in the final algorithm to

- Reject matches produced by Kalman filtering which show too much deviation from epipolar constraints.
- Search for enforced matches for correspondences of points for which the more reliable methods have failed.

This chapter develops the theory behind the epipolar geometry for an affine camera. This is followed by a description of the method that is used to obtain the epipolar constraint parameters and the use of this constraint for rejecting outlying points. The treatment follows [SAB93].

4.1 Epipolar line geometry

Given two frames produced by a stereo-pair, or by movement of the object and/or a single camera, the epipolar of an image point in one frame is defined as the set of possible locations, in the other frame, of the image of any of the world points whose image the original point could have been.

Figure 3.1 shows the classical epipolar line for the case of a stereo-pair under the assumption of *perspective* projection, wherein all the rays converge to some central point. It is seen that if all we know of point X is that it corresponds, in the left image, to x , then the point X could be any one of the several on the ray OX. Depending on where exactly it is on this ray, its image on the right side lies at some point on the

line u' . The line u' is called the *epipolar line* of the point x on left image. Also, the points where the line OO' , i.e. the line joining the two centers of projection intersects the two image planes are called the *epipoles*. It is quite easy to see that all epipolar lines pass through the corresponding epipole and so, are concurrent as illustrated in Figure 3.2

For an affine camera model, the two central points are the *ideal* points or points at infinity. As is seen in Figure 4.1, this implies that while epipolar lines exist similar to those in perspective projection, the epipoles are at infinity. This is manifested as the parallelism of all the epipolar lines as seen from Figure 4.2.

4.2 Mathematical formulation of epipolar line

For the affine camera it is known that if \mathbf{x}_i is the location of the projection of real world point (X_i, Y_i, Z_i) in the image, then

$$\mathbf{x}_i = \mathbf{B} \begin{bmatrix} X_i \\ Y_i \end{bmatrix} + Z_i \mathbf{b} + \mathbf{t} \quad (4.1)$$

where \mathbf{B} and \mathbf{b} are respectively 2×2 and 2×1 arrays dependent on the affine transformation induced by the camera. If the object undergoes an affine transformation in the real world to produce the conditions for the next image, then for the same point i ,

$$\mathbf{x}'_i = \mathbf{B}' \begin{bmatrix} X_i \\ Y_i \end{bmatrix} + Z_i \mathbf{b}' + \mathbf{M}' \mathbf{D} + \mathbf{t} \quad (4.2)$$

where $\mathbf{B}' = \mathbf{B}$ and $\mathbf{b}' = \mathbf{b}$ if the camera remains the same. Eliminating the real world coordinates $(X_i, Y_i)^T$ gives us

$$\mathbf{x}'_i = \mathbf{\Gamma} \mathbf{x}_i + Z_i \mathbf{d} + \epsilon \quad (4.3)$$

where

$$\begin{aligned} \mathbf{\Gamma} &= \mathbf{B}' \mathbf{B}^{-1} \\ \mathbf{d} &= \mathbf{b}' - \mathbf{B}' \mathbf{B}^{-1} \mathbf{b} \\ \epsilon &= \mathbf{t}' - \mathbf{\Gamma} \mathbf{t} + [\mathbf{B}' \mid \mathbf{b}'] \mathbf{D} \end{aligned}$$

The above expression proves that if \mathbf{x}_i is fixed, \mathbf{x}'_i is constrained to lie on a line. The depth Z_i determines how far along the line \mathbf{x}'_i actually lies. The parallelism of all epipolar lines follows from the constancy of \mathbf{d} *w.r.t.* which point is being considered. The converse epipolar line, for \mathbf{x}'_i in the first image, follows by inverting Equation 4.3.

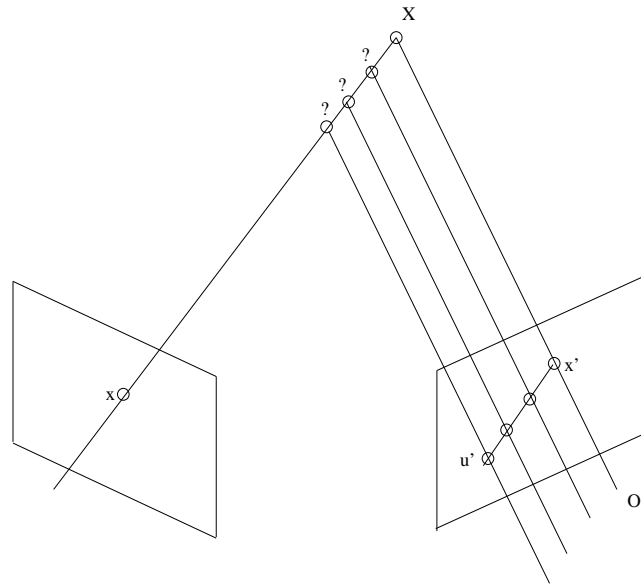


Figure 4.1: Affine camera geometry : All projection rays are parallel

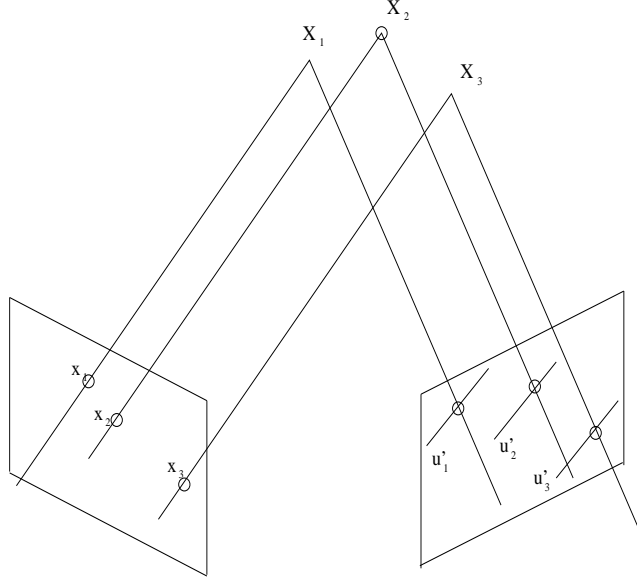


Figure 4.2: Parallelism of epipolar lines in affine epipolar geometry

Finally, considering that the image coordinates are centered i.e. are defined with respect to the image of a fixed object point whose coordinates are assumed to describe the origin, and introducing

$$\Delta \mathbf{x}_i = \mathbf{x}_i - \mathbf{x}_0$$

$$\Delta \mathbf{x}'_i = \mathbf{x}'_i - \mathbf{x}'_0$$

gives the translation-invariant versions of the formulae by the same derivation as already described.

$$\Delta \mathbf{x}'_i = \Gamma \Delta \mathbf{x}_i + \Delta Z_i \mathbf{d} \quad (4.4)$$

4.3 The affine fundamental matrix

Though the previous section has defined the epipolar line, we would prefer an explicit form for it, in which Z_i did not occur. To do this we can separate out the two equations, one for each coordinate, that Equation 4.3 represents and eliminate Z_i from these to get

$$(\mathbf{x}'_i - \Gamma \mathbf{x}_i - \epsilon) \mathbf{d}^\perp \quad (4.5)$$

where

$$\mathbf{d} = \begin{bmatrix} d_x \\ d_y \end{bmatrix} \quad \text{and} \quad \mathbf{d}^\perp = \begin{bmatrix} d_y \\ -d_x \end{bmatrix}$$

This may equivalently written as

$$ax'_i + by'_i + cx_i + dy_i + e = 0 \quad (4.6)$$

where $(a, b)^T = \mathbf{d}$, $(c, d)^T = -\mathbf{\Gamma}^T \mathbf{d}^\perp$, $e = -\epsilon^T \mathbf{d}^\perp = (\mathbf{\Gamma} \mathbf{x}_0 - \mathbf{x}'_0)^T \mathbf{d}^\perp$. The unknown parameters a, b, \dots, e depend on the camera parameters and the particular geometry or motion but are the same for all points of the object. Equation 4.6 is known as the *affine epipolar constraint equation*. It may also be expressed in the form of an *fundamental matrix* \mathbf{Q} by

$$\mathbf{p}'^T \mathbf{Q} \mathbf{p} = 0$$

that is

$$\begin{bmatrix} x'_i & y'_i & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & a \\ 0 & 0 & b \\ c & d & e \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = 0 \quad (4.7)$$

where $\mathbf{p}' = (x', y', 1)^T$ and $\mathbf{p} = (x, y, 1)^T$ are homogeneous 3-vectors representing points in the image plane \mathcal{P}^2 . The epipolar lines corresponding to \mathbf{p} and \mathbf{p}' are $\mathbf{u}' = \mathbf{Q} \mathbf{p}$ and $\mathbf{u} = \mathbf{Q}^T \mathbf{p}'$ respectively, where $\mathbf{u} = (u_1, u_2, u_3)^T$ represents the line $u_1 x + u_2 y + u_3 = 0$ (and correspondingly for \mathbf{u}'). The *normal* to the epipolar line \mathbf{u} is thus in the direction $(u_1, u_2) = (c, d)$ and that to \mathbf{u}' is in direction $(u'_1, u'_2) = (a, b)$. The difference vector form of Equation 4.6 is

$$a \Delta x'_i + b \Delta y'_i + c \Delta x_i + d \Delta y_i = 0 \quad (4.8)$$

It is easy to see that given an image point in one image, the constraint forces its corresponding point in the other to lie on a particular line. All of these lines for various points are parallel as seen from the constancy of the slope.

4.4 Solution of the epipolar equation

Only four unknown quantities exist in Equation 4.6 because only the *ratios* of the five parameters a, \dots, e therein can be calculated. Four point correspondences are thus sufficient to solve for obtaining the value of the four unknowns, since each gives one equation on use of Equation 4.6. However, most generally, we would have n correspondences with $n > 4$. Also, the inevitability of noise and errors means that the points do not lie exactly on the epipolar lines. This necessitates a least squares solution of the system of conditions induced by application of Equation 4.6 on each of these correspondences. Carrying this out involves choosing a particular *cost function* to be minimized.

4.4.1 Choice of cost function

The possible cost functions which arise from previous work are

$$E_1 = \sum_{i=0}^{n-1} \frac{(ax'_i + by'_i + cx_i + dy_i + e)^2}{a^2 + b^2} + \frac{(ax'_i + by'_i + cx_i + dy_i + e)^2}{c^2 + d^2}$$

subject to $a^2 + b^2 = 1$,

$$E_2 = \sum_{i=0}^{n-1} \frac{(ax'_i + by'_i + cx_i + dy_i + e)^2}{a^2 + b^2}$$

subject to $a^2 + b^2 = 1$

and

$$E_3 = \sum_{i=0}^{n-1} \frac{(ax'_i + by'_i + cx_i + dy_i + e)^2}{a^2 + b^2 + c^2 + d^2}$$

subject to $a^2 + b^2 + c^2 + d^2 = 1$. Each of the functions is *scale-invariant* that is, multiplication of the solution vector $(a, b, c, d, e)^T$ by a scalar gives another solution. The first function aims to minimize the distances of points from the epipolar lines in both images simultaneously. The second attempts the same in only one image. The third, however minimizes distances in 4D space and involves a 4D eigenvector solution. While the third function has been criticized in the literature for sensitivity to noise for a perspective camera. But this does not hold for the affine camera model since the strong constraint of parallelism of epipolar lines makes \mathbf{Q} robust to noise. When scale is close to unity, as it frequently is in an affine camera model, E_3 performs as well as E_1 but at a lesser computational cost. The drawbacks of function E_2 are that it minimizes distances in only one image and also assumes that the distribution of the points about the centroid, when weighted by distances, sums to zero. This makes the E_2 method unattractive. Thus, it was decided to use E_3 as the cost function to be minimized.

4.4.2 Use of the solution in algorithm

The affine epipolar geometry constraint is used in the algorithm in two ways which will be expatiated upon herein.

The actual solution proceeds as follows.

- Define

$$r_i^2 = \frac{(ax_i^O + by_i^O + cx_i^N + dy_i^N + e)^2}{a^2 + b^2 + c^2 + d^2} \quad (4.9)$$

and

$$\chi^2 = \sum_{i=1}^n r_i^2 \quad (4.10)$$

where superscripts “O” refer to old image and “N” to new image corresponding respectively to no superscript and priming in the previous discussion.

- We solve for $(a, b, c, d, e)^T$ in a least square sense as described in Appendix A.
- Define

$$\sigma_{epi} = \sqrt{\frac{\chi^2}{P - 5}} \quad (4.11)$$

The number of degrees of freedom is P-5 because the determination of the five parameters (a, b, c, d, e) reduces one degree of freedom each.

- Reject the outliers i.e. all points with

$$r_i > 2.0 \sigma_{epi} \quad (4.12)$$

The justification for this is in terms of the level of confidence we desire to achieve.

- Recompute all the motion parameters and $(a, b, c, d, e)^T$ themselves to get more reliable values.

The other use pertains to the forcing of matches. When some point remains unmatched after the more reliable methods have been applied, an attempt is made to force matches for it. At that time, the epipolar constraint gives an additional requirement on the possible matches, so that the dimension of search space is reduced from two to one, and enabling a greater level of confidence in the matches.

Chapter 5

Affine structure from motion

In this chapter, we discuss two proposed approaches for the recovery of *affine structure*, i.e. scene structure upto some 3D affine transformation. The scheme requires the selection of an object-centered local coordinate system, i.e. a *perceptual frame* as described in Chapter 3 and the use of two images, with no assumption of any knowledge of camera parameters or world coordinates. Among the various approaches which have been proposed for the above and the form of storage of the deduced information, the most notable are those by Ullman and Basri, by Koenderink and van Doorn [KvD91] and by Tomasi and Kanade [TK92], of which the last named has been actually used in our implementation.

5.1 Koenderink and van Doorn

Koenderink and van Doorn worked under the assumption of a weak perspective camera. While their first two basis vectors are chosen in some arbitrary manner (Koenderink and van Doorn formed them from the three points which were the corners of the triangle of largest area), the third was chosen in the *direction of viewing in the first frame* i.e. $\mathbf{E}_k = \mathbf{k}$. Thus $\mathbf{e}_k = \mathbf{M}\mathbf{E}_k = \mathbf{M}\mathbf{k} = \mathbf{0}$, this being the projection of the viewing direction vector on the image plane, obviously zero. Thus only two basis vectors are chosen in the first frame. This enables α_i and β_i to be calculated easily for point i .

$$\Delta \mathbf{x}_i = \alpha_i \mathbf{e}_1 + \beta_i \mathbf{e}_2$$

The third axis vector, \mathbf{e}'_k is non-degenerate in the next frame, but here $\mathbf{e}'_1, \mathbf{e}'_2, \alpha_i$ and β_i are already known, so that calculation of γ_i is straightforward from their observation that the disparity in the position of the actual corresponding point in second frame and its prediction on the basis of assumed coplanarity with plane of

first two axes equals $\gamma_i \mathbf{e}'_k$. The parallelism of these discrepancies verifies the weak perspective camera model.

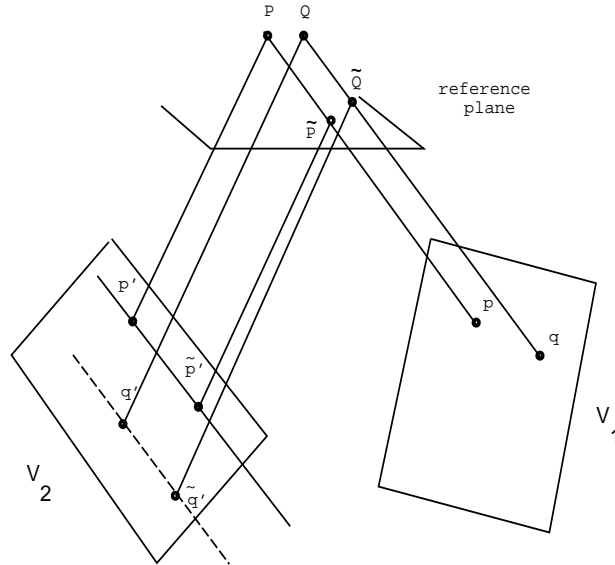


Figure 5.1: Koenderink and van Doorn's determination of structure

Figure 5.1 shows points P and Q which do not lie on the reference plane formed by the two vectors \mathbf{e}_1 and \mathbf{e}_2 . Their respective projections on the reference plane are \tilde{P} and \tilde{Q} in the viewing direction for the first frame V_1 . In this frame both P and \tilde{P} have image p and both Q and \tilde{Q} have image q . In the next frame V_2 , the image of P occurs at p' while it is predicted to be at \tilde{p}' on the assumption that P lies on the reference plane and on the projecting ray through it for the frame V_1 . Thus, \tilde{p}' is the image of \tilde{P} in frame V_2 . So $p' - \tilde{p}'$ is the discrepancy vector for point P . Similarly $q' - \tilde{q}'$ is the discrepancy vector for point Q . Since both of these are scalar multiples of the same vector, \mathbf{e}_k , they are parallel as is clear from the figure.

Here the structure and motion effects are combined *explicitly* in that the first two basis vectors are based purely on structure but the third i.e. \mathbf{e}'_k is also dependent on motion.

The approach has two disadvantages. Firstly, the distribution of errors is uneven among the three basis vectors because of the fact that the computation of the third coordinate, γ is dependent on the results of computations involving the other two. Equation 3.7 does not suffer from this anisotropy of error. Further, it requires several layers of computation in their formalization while Equation 3.7 requires only one ma-

trix calculation. Sudhir *et al* [SBZ93] use this method of affine structure computation in their 3 frame point correspondences algorithm based on structural constraints, and their methods suffer from the above mentioned drawbacks.

5.2 Tomasi and Kanade

One weakness of the approach discussed thus far is that if a particular feature is used as the origin, the values of α , β and γ become susceptible to noise in the measurement of that feature in particular. Tomasi and Kanade [TK92] circumvent this difficulty by using the *centroid* of the features as the origin. Though the two are theoretically equivalent approaches, the second is practically preferable because the centroid can usually be calculated to sub-pixel accuracy and because sensitivity to outlying points is also reduced. Tomasi and Kanade formulated the following which can also be derived by application of the discussion herein over each frame.

$$\begin{bmatrix} \Delta x_1 & \Delta x_2 & \dots & \Delta x_{P-1} \\ \Delta x'_1 & \Delta x'_2 & \dots & \Delta x'_{P-1} \\ \Delta x''_1 & \Delta x''_2 & \dots & \Delta x''_{P-1} \\ \vdots & & & \end{bmatrix} = \begin{bmatrix} e_1 & e_2 & e_3 \\ e'_1 & e'_2 & e'_3 \\ e''_1 & e''_2 & e''_3 \\ \vdots & & \end{bmatrix} \begin{bmatrix} \alpha_1 & \alpha_2 & \dots & \alpha_{P-1} \\ \beta_1 & \beta_2 & \dots & \beta_{P-1} \\ \gamma_1 & \gamma_2 & \dots & \gamma_{P-1} \end{bmatrix}$$

They termed the left matrix the “measurement matrix” \mathbf{W} . Each row indexes a frame and each column a particular feature. There are F frames and P features but only $P - 1$ need be considered as these are centered with respect to the centroid and thus sum to zero. The matrices on the right side were respectively termed as the “motion matrix” \mathbf{M} and the “structure matrix” \mathbf{S} . From this they deduced that \mathbf{W} must have a maximum rank of 3 if noise were absent. They performed a singular value decomposition [WTVF92] on \mathbf{W} and then used the three largest singular values to reconstruct \mathbf{W} as the product of two matrices $\hat{\mathbf{M}}$ and $\hat{\mathbf{S}}$. In the absence of noise $\hat{\mathbf{M}}\hat{\mathbf{S}}$ would equal \mathbf{W} and all remaining singular values would be zero.

To solve for \mathbf{A} , Tomasi and Kanade imposed *metric conditions* on the motion matrix \mathbf{M} by requiring the motion between views to be rigid. This gives a unique solution except when the rank of the system is less than 2, but this case was not investigated by them.

5.2.1 Our implementation of the factorization method

We decompose the “measurement matrix” into 3 matrices, A , D and B by singular value decomposition [WTVF92] as

$$W = [A_1 \ A_2] \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} \quad (5.1)$$

$$= A_1 D_1 B_1 + A_2 D_2 B_2 \quad (5.2)$$

$$= W_T + W_E \quad (5.3)$$

where W is the $P \times 2F$ matrix

$$\begin{bmatrix} \Delta x_1^1 & \Delta y_1^1 & \Delta x_1^2 & \Delta y_1^2 & \dots \\ \Delta x_2^1 & \Delta y_2^1 & \Delta x_2^2 & \Delta y_2^2 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

where $(\Delta x_i^j, \Delta y_i^j)$ are the centered coordinates of the i^{th} corner in the j^{th} frame. In this scenario, D_1 is a 3×3 matrix and D_2 the remnant non-zero part of the D matrix.

5.2.2 Analysis of the implementation

In the implementation given, we obtain the separation of W into two matrices W_T and W_E . It is easy to see that W_T has a rank of 3. It is also the closest such matrix to W in the sense that it minimizes the quantity $\|W_E\| = \|W - W_T\|$ [BIG74] where we define the norm for any matrix M as

$$\|M\| = \text{trace}(MM^*) = \sum_{\text{all } i} \sum_{\text{all } j} m_{i,j}^2$$

This quantity would be expensive to evaluate but singular value decomposition provides a method to calculate it quickly [BIG74] as

$$\begin{aligned} \text{icl}\|W_E\| &= \|W - W_T\| \\ &= \text{trace}((W - W_T)(W - W_T)^*) \\ &= \text{trace}((A_2 D_2 B_2)(A_2 D_2 B_2)^*) \\ &= \sum_{i=4}^{2F} \lambda_i^2 \end{aligned}$$

This allows the norm of the error matrix to be calculated by a linear summation.

Chapter 6

Overall Algorithm

6.1 Matching algorithm

1. Find matches for corners in the previous frame by using the Kalman filter algorithm described in Section 2.5. This yields P quadruples $r_i = (x_i^O, y_i^O, x_i^N, y_i^N)$, $1 \leq i \leq P$.

2. Compute

$$v_i = (\Delta x_i^O, \Delta y_i^O, \Delta x_i^N, \Delta y_i^N) = (x_i^O - \bar{x}_i^O, y_i^O - \bar{y}_i^O, x_i^N - \bar{x}_i^N, y_i^N - \bar{y}_i^N).$$

and

$$\mathbf{w} = \sum_{i=1}^P v_i v_i^T$$

3. Compute the eigenvalues of \mathbf{w} by Jacobi's method¹ and decide whether its rank is 4 or more. If so, stop².
4. Compute a 2 x 2 matrix \mathbf{g} by the Least Squares Method³, such that

$$\begin{bmatrix} \Delta x_1^O & \Delta y_1^O \\ \Delta x_2^O & \Delta y_2^O \\ \vdots & \vdots \\ \Delta x_P^O & \Delta y_P^O \end{bmatrix} \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix} = \begin{bmatrix} \Delta x_1^N & \Delta y_1^N \\ \Delta x_2^N & \Delta y_2^N \\ \vdots & \vdots \\ \Delta x_P^N & \Delta y_P^N \end{bmatrix}$$

¹Since \mathbf{w} is real symmetric.

²Either it is wrong to assume that the camera is affine or the object is undergoing non-affine motion.

³Appendix B

Also compute

$$\sigma_g = \begin{bmatrix} \sigma_{g11} & \sigma_{g12} \\ \sigma_{g21} & \sigma_{g22} \end{bmatrix}$$

and

$$\chi_{gx}^2 = \sum_{i=1}^P \left[\frac{\Delta x_i^N - \Delta x_i^O g_{11} - \Delta y_i^O g_{21}}{\sigma} \right]^2$$

$$\chi_{gy}^2 = \sum_{i=1}^P \left[\frac{\Delta y_i^N - \Delta x_i^O g_{12} - \Delta y_i^O g_{22}}{\sigma} \right]^2$$

where χ_{gx}^2 and χ_{gy}^2 are the chi-square errors in the least squares fit and σ is the standard deviation of the Gaussian error in each Δx_i^N and Δy_i^N . The value of σ is the same as that used in Gaussian smoothing used in the corner detection algorithm. Also compute $R(\frac{P-3}{2}, \frac{\chi_{gx}^2}{2})$ and $R(\frac{P-3}{2}, \frac{\chi_{gy}^2}{2})$, where $R(n, \chi^2)$ is the probability that a random variable having chi-square distribution with n degrees of freedom, exceeds the observed value χ^2 . Compare these two probabilities against some threshold to decide whether \mathbf{g} is a *good fit* or not. This tests whether \mathbf{w} is rank 2 or 3.

5. If \mathbf{w} has rank 2,

- Reject outliers using \mathbf{g} *i.e.* reject any match v_i if

$$\left| \frac{\Delta x_i^N - \Delta x_i^O g_{11} - \Delta y_i^O g_{21}}{\sigma} \right| \geq 2\sqrt{\frac{\chi_{gx}^2}{P-3}}$$

or

$$\left| \frac{\Delta y_i^N - \Delta x_i^O g_{12} - \Delta y_i^O g_{22}}{\sigma} \right| \geq 2\sqrt{\frac{\chi_{gy}^2}{P-3}}$$

- Re-compute v_i and \mathbf{w} . Re-compute the eigenvalues and eigenvectors of \mathbf{w} by Jacobi's method.
- For each corner in the previous frame that is unmatched, predict its new position using \mathbf{g} and define a search area with x - and y - widths proportional to $\sqrt{\frac{\chi_{gx}^2}{P-3}}$ and $\sqrt{\frac{\chi_{gy}^2}{P-3}}$.
- Search the neighborhood region for matches and choose the one with the highest correlation value above a threshold.

6. If \mathbf{w} has rank 3,

- Compute $(a, b, c, d)^T = e_w$, where e_w is the eigenvector of \mathbf{w} corresponding to its least eigenvalue in magnitude. (See *Appendix A*).
- Compute $e = -e_w \cdot \bar{r}$.
- Compute

$$\chi^2 = \sum_{i=1}^P (ax_i^O + by_i^O + cx_i^N + dy_i^N + e)^2$$

- The best estimate of the variance of errors is given by

$$\sigma_{epi} = \sqrt{\frac{\chi^2}{P-5}}$$

- Remove outliers using the affine epipolar constraint *i.e.* any point such that

$$|ax_i^O + by_i^O + cx_i^N + dy_i^N + e| \geq 2.0 \sigma_{epi}$$

- Re-compute $r_i, \bar{r}, v_i, \mathbf{w}$ and the affine epipolar constants a, b, c, d, e .
- Consider the corners in the previous frame which have been matched and for which, the affine structure exists. Compute a 4 x 2 matrix \mathbf{h} using the Least Squares Method such that

$$\begin{bmatrix} \alpha_1 & \beta_1 & \gamma_1 & 1 \\ \alpha_2 & \beta_2 & \gamma_2 & 1 \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \\ h_{31} & h_{32} \\ h_{41} & h_{42} \end{bmatrix} = \begin{bmatrix} x_1^N & y_1^N \\ x_2^N & y_2^N \\ \vdots & \vdots \end{bmatrix}$$

The matrix on the right hand side contains the centered coordinates of the matched corners in the new frame⁴.

- For each unmatched corner in the previous frame, whose affine structure is known, predict the position of that corner in the new frame using \mathbf{h} . The size of the neighborhood to be searched depends upon the covariance matrix σ_h . Choose the corner which maximizes the cross-correlation value, above a certain minimum threshold.
- For each unmatched corner in the previous frame (irrespective of whether its affine structure is known or not), compute the corresponding epipolar line in the new frame,

$$L : cx + dy + (ax_i^O + by_i^O + e) = 0$$

⁴Although the same notation has been used, it should not be confused with the one used for all the matched corners, as in step 1

Compute that point on this line so that the speed of the corner does not exceed the average speed of the currently matched corners substantially. Define a search area and choose that candidate which has the highest correlation value above a minimum threshold.

7. Compute the affine structure of the corners which have a match history of at least F frames (F is a user-defined parameter, typically 6). Construct W as the $P \times 2F$ matrix

$$\begin{bmatrix} \Delta x_1^1 & \Delta y_1^1 & \Delta x_1^2 & \Delta y_1^2 & \dots \\ \Delta x_2^1 & \Delta y_2^1 & \Delta x_2^2 & \Delta y_2^2 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

where $(\Delta x_i^j, \Delta y_i^j)$ are the centered coordinates of the i^{th} corner in the j^{th} frame. Compute the singular value decomposition of W to yield

$$W = [A_1 \ A_2] \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} \quad (6.1)$$

$$= A_1 D_1 B_1 + A_2 D_2 B_2 \quad (6.2)$$

$$= W_T + W_E \quad (6.3)$$

where A_1 is $P \times 3$, D_1 is 3×3 and B_1 is $3 \times 2F$. The matrix A_1 contains the affine structure of the P points and the matrix B_1 contains the basis vectors. Compute the average pixel-error in the corners as

$$\epsilon = \sqrt{\sum_{i,j} w_{E(ij)}^2}$$

8. Reject any corner which has an average pixel error exceeding two times ϵ . Re-compute W and the affine structure A_1 .

6.2 Gaze point computation

We have implemented with the following choices of gaze-point

- Centroid of all matched corners.
- Centroid of all matched corners, each weighted by its age.
- Follow the projection of a fixed affine coordinate (may or may not be a real corner). Switch to a nearby point if the projection moves too far away from the centroid.

6.3 Universal Hashing for Neighborhood Search

To search a given neighborhood for possible candidates, we need to access the corners in the least possible time. One approach is to use a probabilistic method i.e. hashing. We have implemented hashing using a Universal class of hash functions, as proposed by Wegman and Carter [CW79], which ensure that the expected time for k searches is a linear function of k , averaged over all the functions belonging to that class.

Chapter 7

Results and Analysis

We have implemented two algorithms. The first is the traditional Kalman filtering based tracking method. The second incorporates the affine epipolar constraint as well as computes the affine structure of the object.

The overall algorithm involves several user-defined parameters, whose typical values were mentioned where they were used in the algorithms.

7.1 Implementation Issues

As described in Section 6.1, in step 3, we need to compute the rank of \mathbf{w} . Typically, the case when the rank is 4 is easily detected. However, due to the small motion of the object across successive frames and due to small perspective effects, it is difficult to differentiate between the cases when the rank is 2 and when it is 3. A typical instance of the square roots of eigenvalues¹ of \mathbf{w} is $\langle 1.619, 1.280, 0.034, 0.017 \rangle$.

To distinguish between ranks 2 and 3, we compute \mathbf{g} , the best 2 x 2 affine transformation that maps the centered corners in the previous frame to their new positions in the new frame. The *goodness of fit* indicates how well \mathbf{g} maps the corners. If R is smaller than a certain user-defined threshold, then the rank of \mathbf{w} is 3.

Another issue to be resolved is to segment the image into two parts : one which is the foreground and moving, and the rest which comprises the static background. In our implementation, we have not distinguished between the two kinds of corners. When the background corners are few, they will get ruled out when outliers in the affine structure computation in step 8 of the algorithm are removed. This was the case in all our examples. However, if the number of background corners is substantial, we

¹Actually, we are interested in the eigenvalues of $\mathbf{v} = [v_1|v_2|\dots|v_P]$, and the eigenvalues of \mathbf{w} are squares of these.

need better approaches to remove them before even carrying out the affine structure computation.

Segmentation of an image, using optic flow equations, for example, is an expensive operation, as it sweeps through the entire image at least once. If time is at a premium, we could assume that only those corners which exceed a certain velocity threshold are *interesting*, *i.e.* need to be tracked. However, as a compromise, we stand to lose some structural information in case the motion is very small for an extended period of time.

7.2 Experimentation

We have implemented an off-line version of our algorithm using the *HORATIO* image-processing library [McL93] and carried out extensive off-line experimentation on image sequences of a head, undergoing motion of varying nature. In the first image sequence of 43 frames (taken at 7.0 *Hz*), the motion is primarily in the vertical direction, with the person nodding his head up and down. We refer to this sequence as *nod1*.

The second sequence of 170 frames (at 7.5 *Hz*) has a head moving in arbitrary directions, including rotation, nodding, shaking etc. We refer to this sequence as *rand1*.

Some statistics contrasting both the methods have been plotted in the figures in this chapter.

7.3 Analysis

Some statistics contrasting the performance of simple Kalman filtering against our method have been plotted in Figures 7.3 to 7.10.

In *nod1*, the total number of corners detected, on the average, is approximately 85 (Figure 7.3). Kalman filtering alone is able to match about 30 to 60 corners, whereas our approach is able to match a total of about 40 to 65 corners (Figure 7.3). Moreover, our method appears to *aid* Kalman filtering, as seen in Figure 7.4, because the number of matches reported by the Kalman filter is larger in our approach.

The average age of all the corners is larger if only Kalman filtering is used (Figure 7.5). This is to be expected, because those corners which can be reliably tracked over several frames (because their image velocity and the intensity pattern around their location are slowly varying) will not be *lost* by the Kalman filter. Our method helps to prolong their age and in the process, matches so many corners that the average age falls. However, the large value for average age (about 8 - 12) shows the stability of both the algorithms.

The number of corners used for structure computation is larger in our algorithm, as seen in Figure 7.6.

The superiority of the structure-based approach is amply exemplified by Figure 7.7 where the average pixel error ϵ in the corners (See Step 7) is plotted for the two cases. Our approach gives a pixel error of only about 0.7 to 2.3 per corner. However, using Kalman filtering alone gives a pixel error of as high as 3.1. The error is almost always larger for this method.

The fixation of the gaze point using the centroid of the corners, weighted by their age, exhibits a lot of jitter (Figures 7.8 and 7.9), leading to instabilities in real-time tracking with an active head-eye platform, even with the 12.5 *Hz* Nyquist rate filtering of the gaze demand as carried out by Yorick [RSMD92].

However, gaze point fixation by tracking a specific corner is very smooth in our method (Figure 7.8). In the case of Kalman filtering alone, tracking a fixed corner does not reduce jitter to that extent (Figure 7.9). A comparison of the gaze demand (between two frames) for the two approaches can be seen in Figure 7.10. The gaze demand is much smaller for our method using a specific point for fixation, as compared to Kalman filtering with weighed centroid of corners.

Similar observations hold for the sequence rand1. The total number of corners is about 75-80, out of which the structure-based method is able to match about 50-55 corners. Kalman filtering alone matches upto 10 corners less. The age of corners is again smaller for the former approach. However, the average pixel error in the affine structure is invariably smaller for the structure-based method.

7.4 Conclusions

The structure-based method yields the same order of matched corners as the traditional Kalman filtering approach. The number is usually marginally larger. However, the larger number of matches reduces the average age of the matched corners. Even the average age of corners used for affine structure computation is lower.

However, the matches obtained by the structure-based approach are much more reliable, as shown by the consistently lower average pixel error in the corner positions used for affine structure computation.

Moreover, both methods are stable, with the average age of matched corners being in the range 15-25.

Fixation of gaze point is robust, as the head is always in the field of view. However, the jitter in the camera is very large if the gaze point is fixed at the centre of mass of corners, weighed by their ages. This jitter can be effectively removed by fixating the gaze at a particular corner, switching over to a nearby corner if this gets occluded or moves far away from the centroid.

In short, with the affine structure based approach, we have gained in the total number of matches. And the quality of matches is definitely superior, as shown by the error in the pixels while computing the affine structure. The disappearance and re-appearance of corners and the problem of corner displacement has been effectively handled in the new approach.

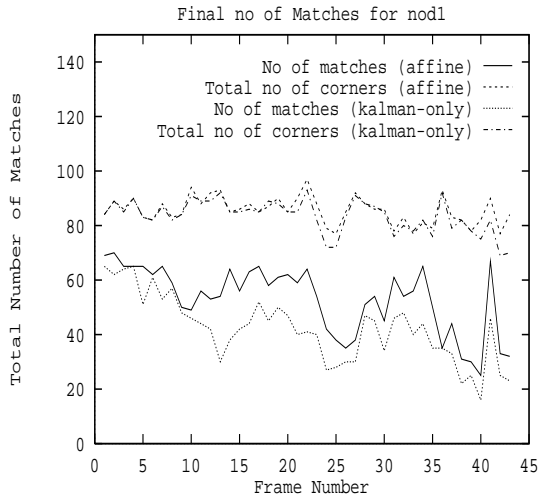


Figure 7.3: Total number of matches in nod1.

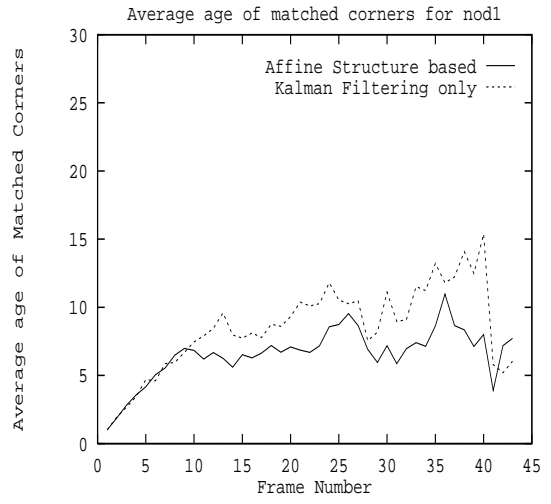


Figure 7.5: Average age of matched corners in nod1.

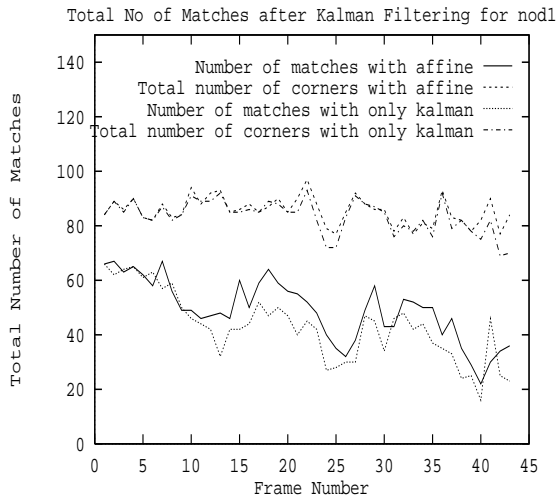


Figure 7.4: Number of matches reported by Kalman filter in nod1.

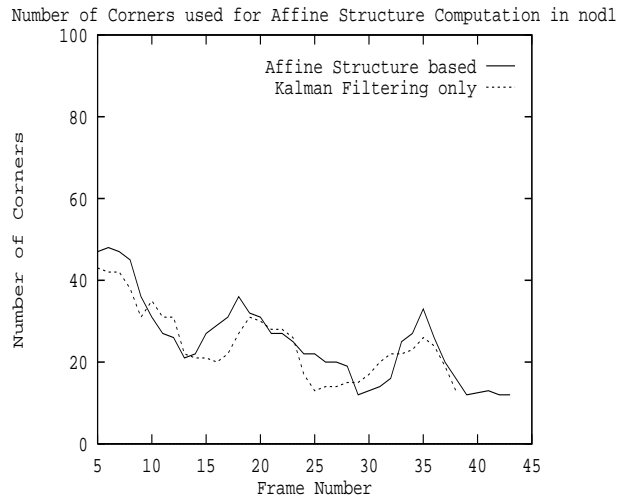


Figure 7.6: Number of corners for affine structure computation in nod1.

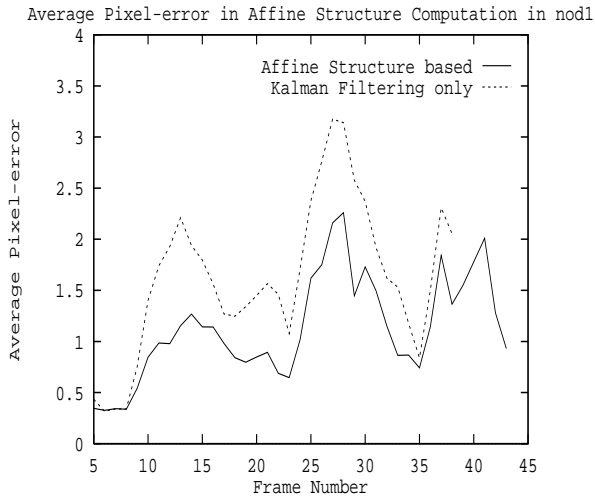


Figure 7.7: Average pixel error in affine structure computation

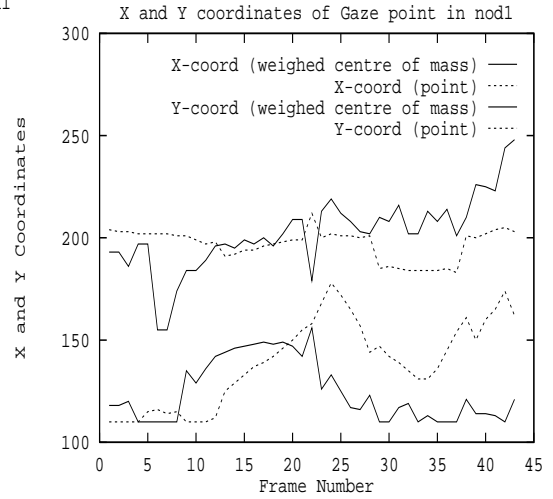


Figure 7.9: X and Y coordinates of gaze point using only Kalman filtering.

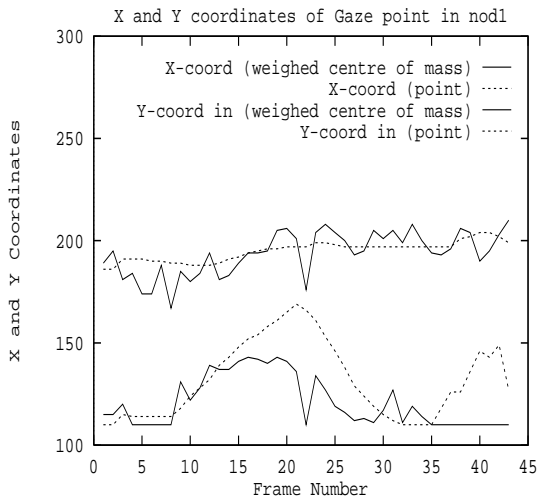


Figure 7.8: X and Y coordinates of gaze point in affine-structure based method.

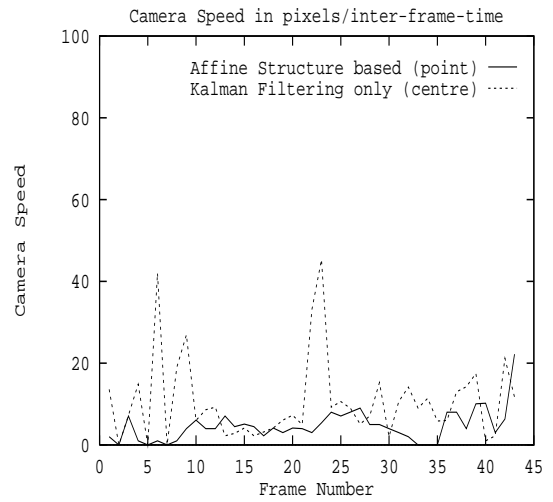


Figure 7.10: Gaze demand in nod1.

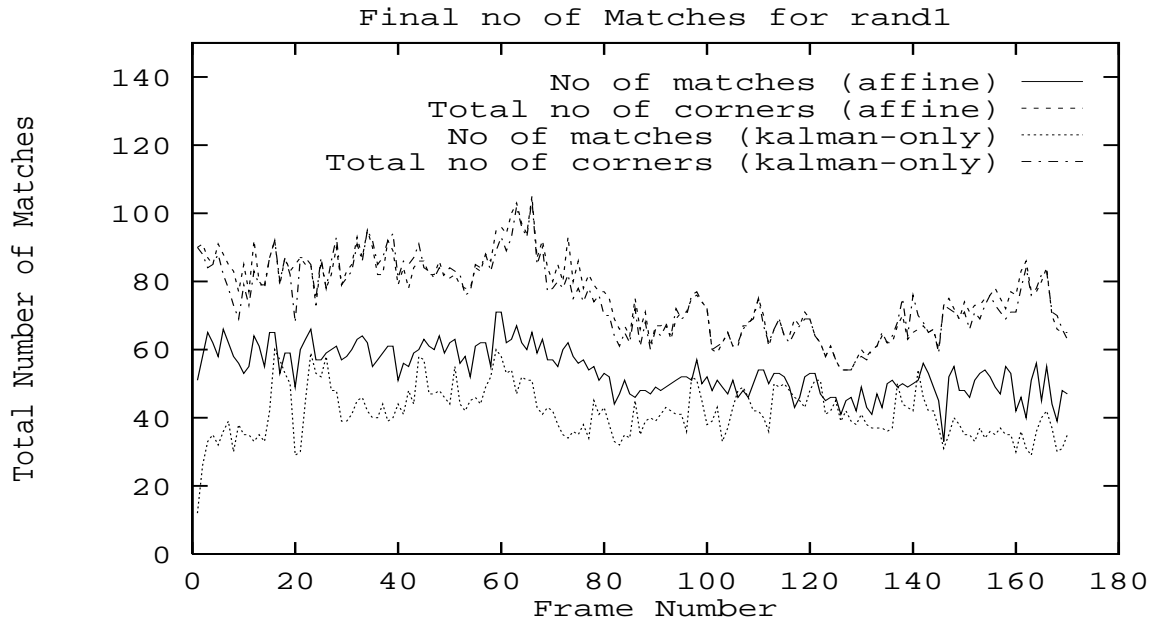


Figure 7.18: Total number of matches in rand1.

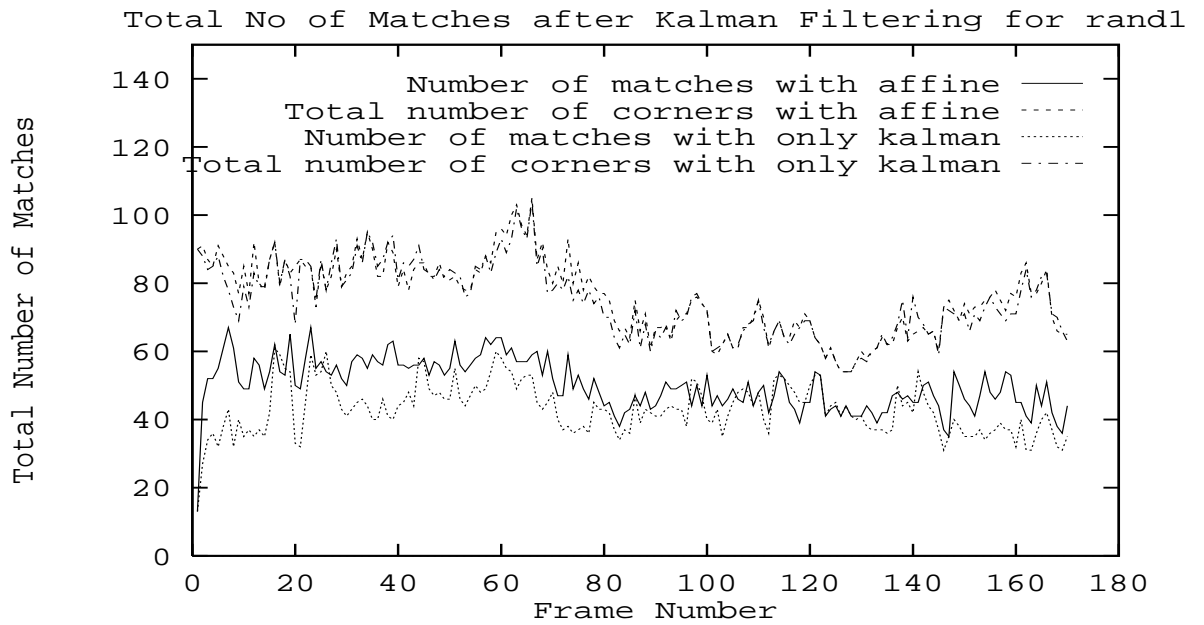


Figure 7.19: Number of matches reported by Kalman filter in rand1.

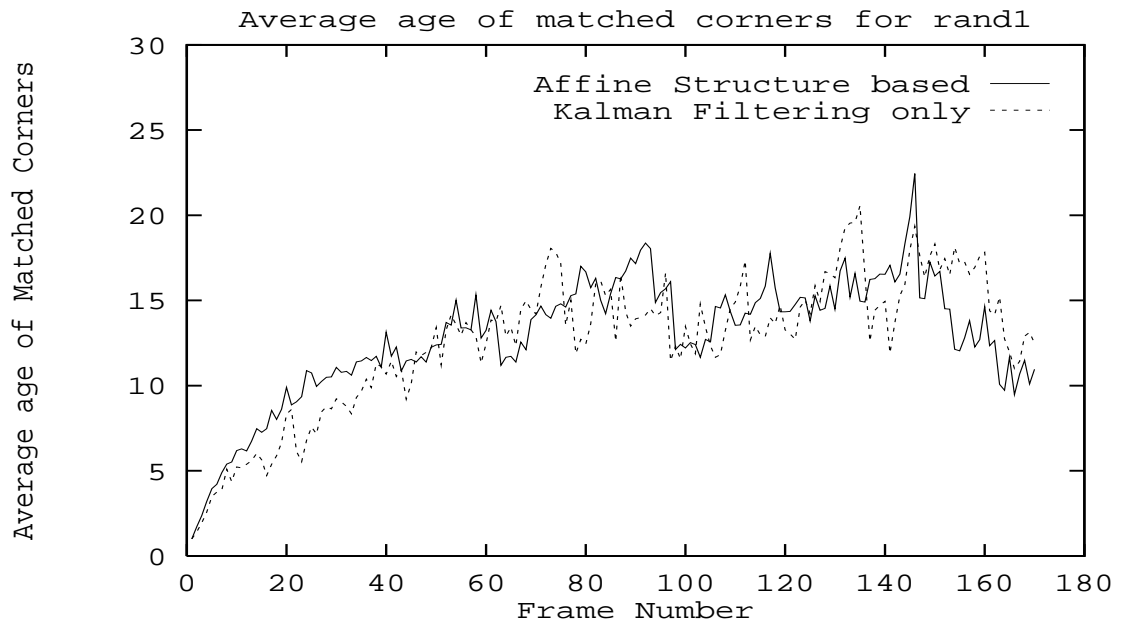


Figure 7.20: Average age of matched corners in rand1.

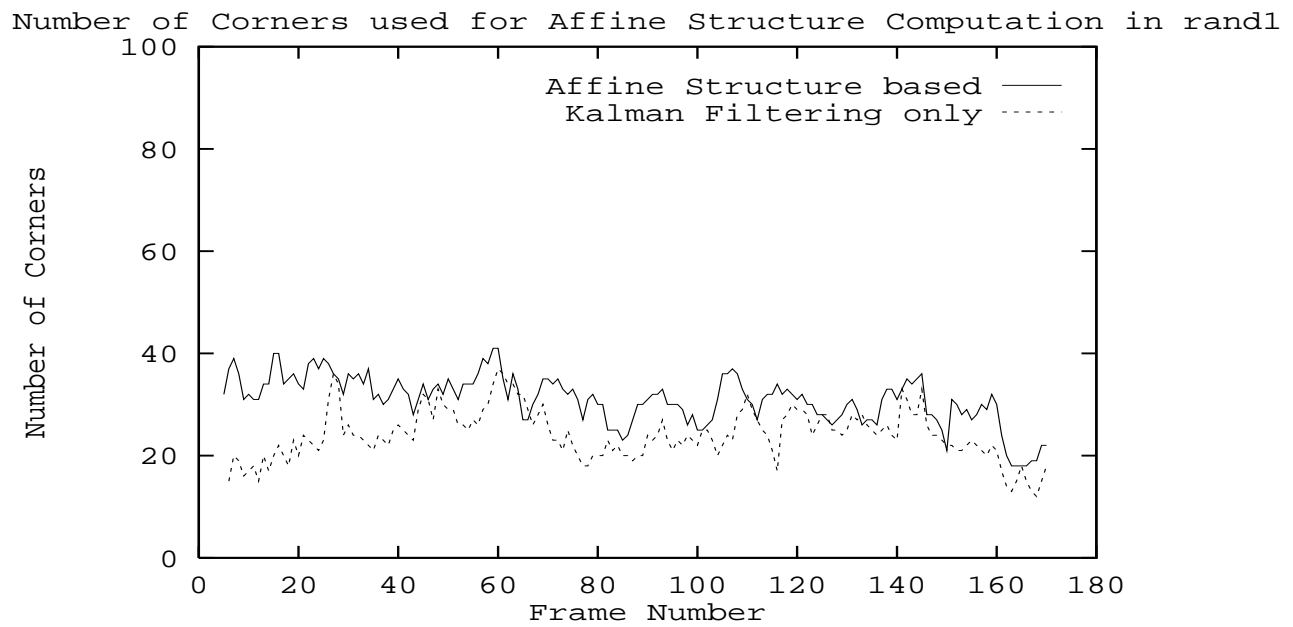


Figure 7.21: Number of corners for affine structure computation in rand1.

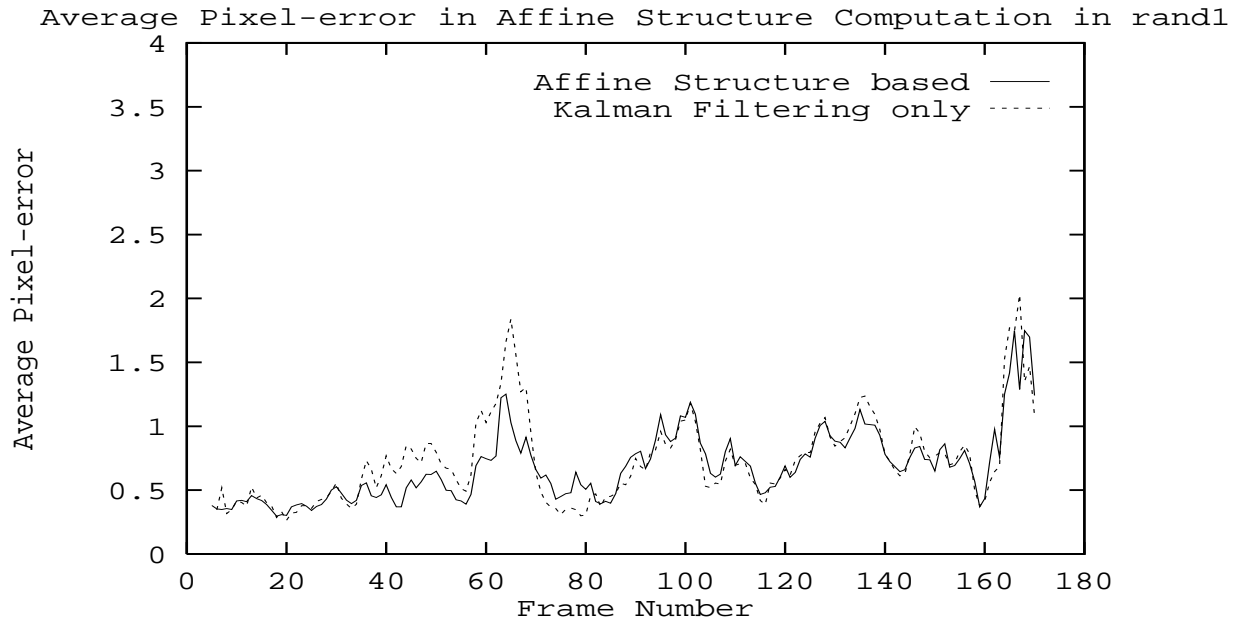


Figure 7.22: Average pixel error in affine structure computation

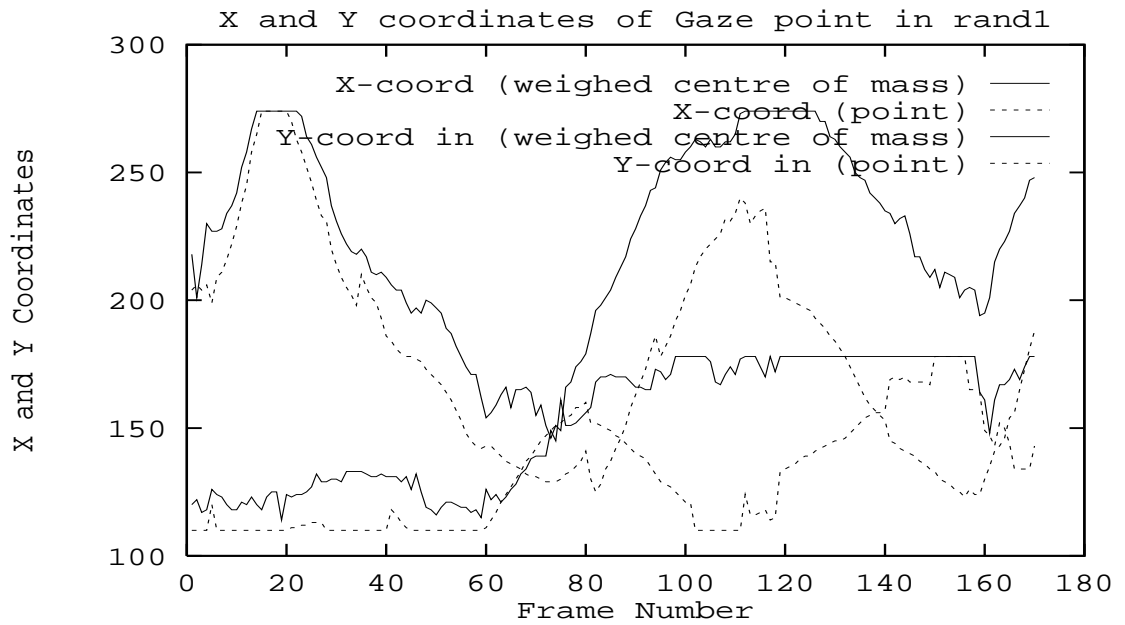


Figure 7.23: X and Y coordinates of gaze point in affine-structure based method.

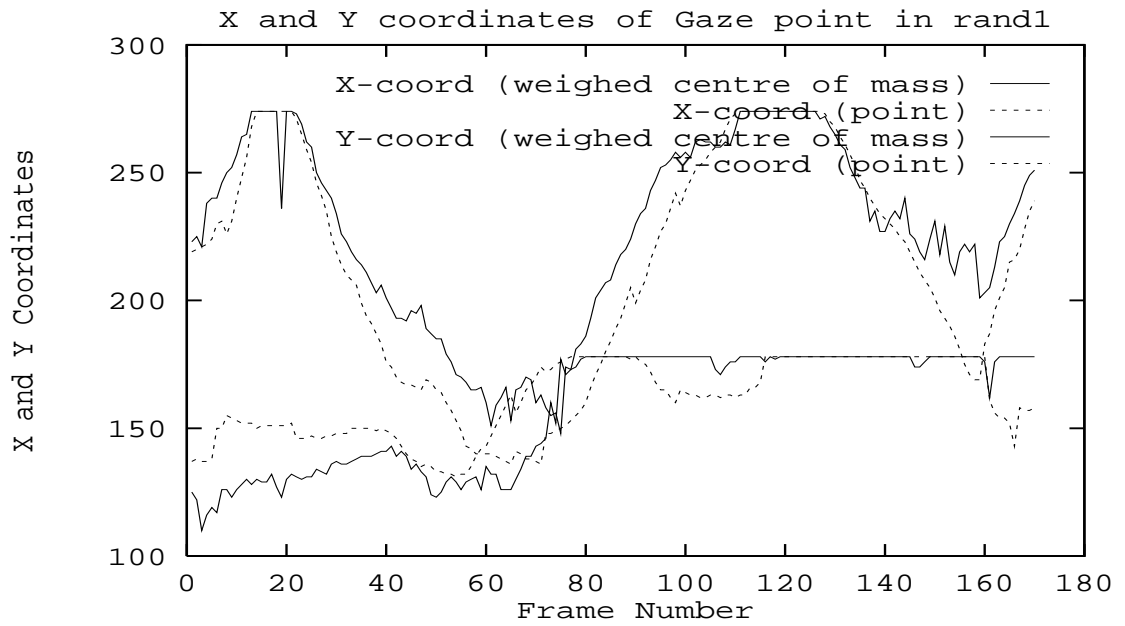


Figure 7.24: X and Y coordinates of gaze point using only Kalman filtering.

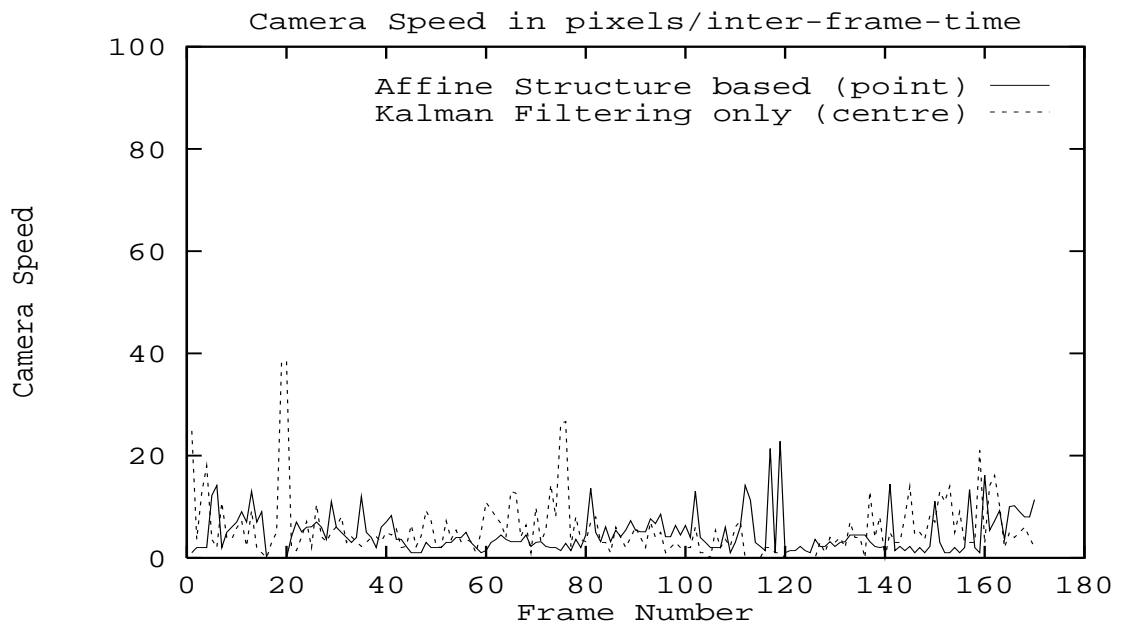


Figure 7.25: Gaze demand in rand1.

Chapter 8

Future Work

We have presented a tracking algorithm based on affine multiple views geometry which significantly improves the performance over simple Kalman filtering based tracking. However, the method is not suitable for real-time implementation because of the high computational overheads. Unlike the simple Kalman filter, the new algorithm does not have a linear $O(n)$ time complexity (for tracking an average of n corners).

The structure based approach described in this report uses the factorization method proposed by Tomasi and Kanade[TK92]. One drawback of such an approach is that it is a batch process. It needs reliable matches for several previous frames and re-uses the same information of a corner repeatedly in successive affine structure computations. Another drawback is that the singular value decomposition algorithm is iterative, and may converge very slowly, consuming a large amount of time. The iterative nature of the SVD adds to the computational burden.

For real-time implementation, a more suitable scheme would be to sequentially update the Affine Structure estimate. We are presently experimenting with such a method. Finally, we would like to point out that most of the ideas presented in the paper can be extended to deal with perspective cases using projective structure (which requires a 5 point basis) and projective epipolar geometry [Fau92].

Appendix A

Best epipolar line fit

A.1 Statement of the problem

Given

$$a^2 + b^2 + c^2 + d^2 = 1$$
$$\mathbf{n} = (a, b, c, d)^T$$

and n 4-tuples, $(x'_i, y'_i, x_i, y_i) = \mathbf{r}_i$ from pairs of corresponding points in two images, it is required to minimize

$$E_3 = \frac{1}{|\mathbf{n}|^2} \sum_{i=0}^{n-1} (\mathbf{r}_i \cdot \mathbf{n} + e)^2$$

A.2 Solution and proof

Using the Lagrange multiplier λ , we obtain

$$E' = \sum_{i=1}^n (\mathbf{n} \cdot \mathbf{r}_i + e)^2 - \lambda (\mathbf{n}^T \mathbf{n} - \mathbf{1}).$$

The optimum hyperplane passes through the data centroid $\bar{\mathbf{r}}$. After substituting $e = -\mathbf{n} \cdot \bar{\mathbf{r}}$ and writing $\mathbf{W} = \sum_i \mathbf{v}_i \mathbf{v}_i^T$, we obtain

$$E' = \mathbf{n}^T \mathbf{W} \mathbf{n} - \lambda (\mathbf{n}^T \mathbf{n} - \mathbf{1}).$$

Differentiating with respect to \mathbf{n} yields

$$\frac{\partial E'}{\partial \mathbf{n}} = \mathbf{0} = 2\mathbf{W}\mathbf{n} - 2\lambda\mathbf{n} \Rightarrow \mathbf{W}\mathbf{n} = \lambda\mathbf{n} \tag{A.1}$$

Thus \mathbf{n} is found to be an eigenvector of \mathbf{W} corresponding to the eigenvalue λ . To decide the particular eigenvalue to be used, we substitute into the expression for E_3 .

$$E_{3,min} = \mathbf{n}^T \mathbf{W} \mathbf{n} = \mathbf{n}^T \lambda \mathbf{n} = \lambda |\mathbf{n}|^2 = \lambda$$

Since we are minimizing E_3 , λ must be the minimum eigenvalue of \mathbf{W} and \mathbf{n} its associated eigenvector.

Appendix B

Least Square Solutions

Here, we summarize the main results used from the theory of least square solutions. No attempt has been made to rigorously derive any results.

B.1 Error in only one variable

Problem

Assume that y is a linear function of M variables, modeled by

$$y = \sum_{k=1}^M a_k x_k.$$

Given a set of N data points $(x_{i1}, x_{i2}, \dots, x_{iM}, y_i), 1 \leq i \leq N$, the problem is to estimate the parameters (a_1, a_2, \dots, a_M) .

Solution

It can be shown that the *Maximum Likelihood Estimate* of (a_1, a_2, \dots, a_M) can be obtained if we assume Gaussian errors in each observation y_i and minimize the function

$$\chi^2 = \sum_{i=1}^N \left[\frac{y_i - \sum_{k=1}^M a_k x_{ik}}{\sigma_i} \right]^2,$$

where σ_i is the measurement error (standard deviation) of the i^{th} data point. No error is assumed in the measurement of any x_{ik} .

Define an $N \times M$ matrix \mathbf{A} , with entries

$$A_{ij} = \frac{x_{ij}}{\sigma_i}$$

Matrix \mathbf{A} is called the design matrix of the problem. Also, define an $N \times 1$ matrix \mathbf{b} as

$$b_i = \frac{y_i}{\sigma_i}$$

and let \mathbf{a} denote the column vector consisting of the parameters to be estimated.

The value of χ^2 is maximized if its derivative with respect to all the M parameters a_k vanishes. This yields the system of equations :

$$0 = \sum_{i=1}^N \frac{1}{\sigma_i^2} \left[y_i - \sum_{j=1}^M a_j x_{ij} \right] x_{ik}$$

These are called the normal equations of the problem and are equivalent to the system of equations given by

$$(\mathbf{A}^T \cdot \mathbf{A}) \cdot \mathbf{a} = \mathbf{A}^T \cdot \mathbf{b} \tag{B.1}$$

And it can be showed that the uncertainty in the parameters is given by the matrix $\mathbf{C} = (\mathbf{A}^T \cdot \mathbf{A})^{-1}$, which is the covariance matrix for \mathbf{a} .

Solution by LU decomposition

Equations B.1 can be solved efficiently by LU decomposition and \mathbf{C} can then be computed¹. However, if the matrix $\mathbf{A} \cdot \mathbf{A}^T$ has rank less than M , then LU decomposition will fail. In such a case, it is useful to factorize the matrix using SV decomposition.

Solution by Singular Value Decomposition

In terms of the design matrix \mathbf{A} and the vector \mathbf{b} , the minimization of χ^2 can be written as :

Find \mathbf{a} to minimize $\chi^2 = |\mathbf{A} \cdot \mathbf{a} - \mathbf{b}|^2$.

This is precisely the problem solved by SV Decomposition, which factorizes the matrix $\mathbf{A} = \mathbf{U} \mathbf{W} \mathbf{V}^T$, where \mathbf{U} is a column-orthogonal matrix, \mathbf{W} is a diagonal matrix and \mathbf{V} is an $M \times M$ orthogonal matrix. If $\mathbf{U}_i, 1 \leq i \leq M$ denotes the columns of \mathbf{U} and $\mathbf{V}_i, 1 \leq i \leq M$ denotes the columns of \mathbf{V} , the least squares solution is given by

$$\mathbf{a} = \sum_{i=1}^M \left(\frac{\mathbf{U}_i \cdot \mathbf{b}}{w_i} \right) \mathbf{V}_i$$

where w_i are the singular values (diagonal elements of \mathbf{W}).

¹In theory, since $\mathbf{A} \cdot \mathbf{A}^T$ is positive definite, Cholesky factorization is also possible

It turns out that the standard error in the fitted parameters is such that we can re-write the previous equation as :

$$\mathbf{a} = \sum_{i=1}^M \left(\frac{\mathbf{U}_i \cdot \mathbf{b}}{w_i} \right) \mathbf{V}_i + \sum_{i=1}^M \epsilon_i - \mathbf{V}_i.$$

The covariance matrix \mathbf{C} is given by

$$C_{jk} = \sum_{i=1}^M \frac{1}{w_i^2} V_{ji} V_{ki}$$

Goodness of Fit

Apart from the estimation of \mathbf{a} and its covariance matrix, we need a measure how good the fit is. Note that the value of χ^2 is an instance of a random variable with χ_{N-M}^2 distribution². The probability that an instance of χ_{N-M}^2 exceeds χ^2 is given by $R(N - M, \chi^2)$. The detailed theory may be found in [WTVF92]. We can set a threshold on this probability to decide whether the fit is *good* or not.

²With mean $N - M$ and variance $\sqrt{2(N - M)}$

Bibliography

- [AB87] J. Aloimonos and A. Bandopadhyay. Active vision. In *Proc. Intl. Conf. Computer Vision*, pages 35–54, 1987.
- [BIG74] Adi Ben-Israel and Thomas N. E. Greville. *Generalized Inverses: Theory and Applications*. John Wiley & Sons., 1974.
- [BSF88] Y. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association*. Academic Press., 1988.
- [CW79] J. L. Carter and M. N. Wegman. Universal Classes of Hash Functions. In *J. Comput. System Sci.*, pages 143–154, April 1979.
- [Fau92] O. D. Faugeras. What can be seen in Three Dimensions with an Uncalibrated Stereo Ring? In *Proc. European Conf. on Computer Vision, Santa Margherita, Italy*, pages 563–576, 1992.
- [KvD91] J. J. Koenderink and A. J. van Doorn. Affine Structure from Motion. In *Journal of the Optical Society of America.*, pages 377–385, 1991.
- [McL93] P. F. McLauchlan. HORATIO : Libraries for Vision Applications. Technical report, Robotics Research Group, University of Oxford, WP3/OXFORD/930112/HORATIO, 1993.
- [RM93] I. D. Reid and D. W. Murray. Tracking Foveated Corner Clusters using Affine Structure. In *Proc. Intl. Conf. Computer Vision*, pages 76–83, 1993.
- [RSMD92] I.D. Reid, P.M. Sharkey, P.F. Mclauchlan, and Murray D.W. A modular head/eye platform for real-time reactive vision. Technical report, Dept. of Engineering Science, University of Oxford, 1992.
- [SAB93] L. S. Shapiro, Zisserman A., and J. M. Brady. What Can One See With an Affine Camera? Technical report, Robotics Research Group, University of

Oxford, 1993. Also see Motion from Point Matches using Affine Epipolar Geaometry, to appear in IJCV.

- [SBZ93] G. Sudhir, S. Banerjee, and A. Zisserman. Finding Point Correspondences in Motion Sequences Preserving Affine Structure. In *Proc. British Machine Vision Conf.*, 1993. Also revised for CVGIP: Image Understanding.
- [TK92] Carlo Tomasi and Takeo Kanade. Shape and Motion from Image Streams under Orthography: a Factorization Method. In *Intl. J. Computer Vision*, pages 137–154, 1992.
- [WB91] H. Wang and J. M. Brady. Corner Detection for 3D Vision using Array Processors. In *Proc. BARNAIMAGE-91, Barcelona, Spain*. Springer-Verlag, 1991.
- [WB92] H. Wang and J. M. Brady. Corner Detection with Sub-pixel Accuracy. Unpublished. 1992.
- [WTVF92] Saul A. Teukolsky William T. Vetterling, William H. Press and Brian P. Fleming. *Numerical Recipes in C*. Cambridge University Press., May 1992.