Robust Feature Point Extraction and Tracking for Augmented Reality

Bruno Palacios Serra
Supervisor: Prof. Touradj Ebrahimi
Resp. assistant: David Marimón Sanjuán

September 15, 2005

Contents

List of Figures

List of Tables

Abstract

For augmented reality applications, accurate estimation of the camera pose is required. An existing video-based markerless tracking system, developed at the ITS, presents several weaknesses towards this goal. In order to improve the video tracker, several feature point extraction and tracking techniques are compared in this project. The techniques with best performance in the framework of augmented reality are applied to the present system in order to prove system's enhancement.

A new implementation for feature point extraction, based on Harris detector, has been chosen. It provides better performance than Lindeberg-based former implementation. Tracking implementation has been modified in three ways. Track continuation capabilities have been added to the system with satisfactory results. Moreover, the search region used for feature matching has been modified taking advantage from pose estimation feedback. The two former modifications working together succeed in improving system's performance. Two alternatives to photometric test, based on Gaussian derivatives and Gabor descriptors, have been implemented showing that they are not suitable for real-time augmented reality. Several improvements are proposed as a future work.

Chapter 1

Introduction

An Augmented Reality (AR) system combines the real world with computer-generated objects in real time. AR applications include medical, architecture, inspection and assembling among many others. All AR applications rely on the proper registration of augmented information. Any fail in registration will prevent the user from seeing the virtual objects merged correctly in the scene. In order to obtain a correct registration in three-dimensional AR applications, accurate pose estimation of the camera is required.

AR registration has been highly successful using mechanical, magnetic or acoustic-based trackers [3]. Even optical (video)-tracking using markers has lead to good results [1]. All these tracking techniques have in common that they require preparation of the scene where AR takes place.

However, in outdoor unprepared environments and mobile applications, as noted by Azuma et al., tracking remains an enormous challenge [4]. In these scenarios, the camera has to be tracked using only natural scene features. This is called markerless tracking. These freedom, however, leads to worse pose estimation accuracy as it is estimated using feature point correspondences between frames.

1.1 Starting Point and Aims of the Project

The starting point in this project is an already implemented augmented reality system using markerless tracking. In Chapter 5 current system characteristics are shown. Previous work on our group has proven several weaknesses of video-based tracking. It fails when there are occlusions or brusque changes of illumination in the scene. Furthermore, it depends heavily on the computational cost of the algorithms, which may cause system delay.

This project aims at improving markerless tracking by enhancing feature point extraction and tracking. In order to do so, several steps are to be followed, taking into account augmented reality requirements.

Feature point extraction is performed with Lindeberg detector. In order to enhance the system, our objective is to find a better detector. Harris corner detector implementation if compared with current one shows better results will be incorporated to the system.

Present photometric test implementation is based on cross-correlation coefficient. Two more approaches will be implemented, based on Gaussian derivatives and Gabor filters, and compared to find the one with best performance.

For preventing premature track deaths due to occlusions or failures in the extraction step, a track continuation algorithm will be designed and implemented. A track quality indicator that measures if a track shall be continued has to be found.

In order to enhance the assignment technique in the tracking step, a new hybrid matching criterion will be implemented taking advantage from pose estimation information.

1.2 Organisation of the Document

This report comprises four parts. First of all, in Part I, we expose some theoretical background to ease the comprehension of further chapters.

Part II is a description of the system. Part III is the collection of experiments and results of the overall project. In Part IV the conclusions are exposed and future work is proposed.

Part I

Theoretical Background

Chapter 2

Feature Point Extraction

Feature point extraction is an important pre-processing step in image processing and computer vision for applications such as image registration, object recognition among others.

In the literature many definitions for a feature point are proposed. Schmidt and Mohr define interest point as a local feature where the signal changes in the two dimensions, such as corners or intersections [5]. Those have advantages over features like edges or regions, specially their high informational content and robustness to partial visibility.

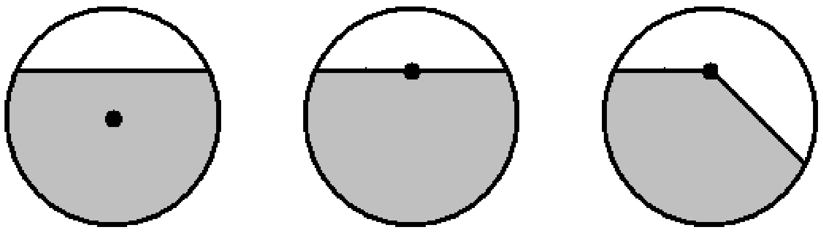
Hence we can use corner detectors as feature point extractors. There are two approaches to corner detection, we can extract the contours and search the points with maximal curvature on them, or search for feature points directly at the gray level, that is using the intensity of an image. Intensity based algorithms are preferable, as contour-based algorithms depend on the performance of their edge detectors [6].

In this chapter, a detailed description of existing intensity based corner detectors is given. We also define some evaluation criteria for comparison between techniques.

2.1 Intensity Based Corner Detectors

All the algorithms introduced here, excepting SUSAN detector, use differential operators. They are combinations of first and second derivatives of the intensity function I(x,y).

SUSAN corner detector proposed by Smith and Brady is based on the direct comparison of intensity values. It is intended for low-level image processing [7]. It consists in considering a circular window around an arbitrary pixel, the nucleus, which may contain a compact region whose pixels have similar brightness to the nucleus (Univalue Segment Assimilating Nucleus, USAN, region). The area and the gravity centre of the USAN are computed and used as parameters to find the corners. In figure 2.1 we can see three representative shapes of the USAN.



(a) *normal point* (b) *edge point* (c) *corner point*

Figure 2.1: *Representative shapes of USAN*

Beaudet [6][8] proposed to calculate Hessian determinant

$$|H(x,y)| = I_{xx}I_{yy} - I_{xy}^2 \tag{2.1}$$

of the intensity where the local extrema would be the corners.

Kitchen and Rosenfeld [8] proposed the following operator

$$K(x,y) = \frac{I_{xx}I_y^2 + I_{yy}I_x^2 - 2I_{xy}I_xI_y}{I_x^2 + I_y^2} \tag{2.2}$$

that represents the curvature of a plane curve perpendicular to the gradient of the intensity of the image. Local extrema or maxima of its absolute value are identified as features.

Lindeberg [6] proposed to use only the numerator of last operator, obtaining a polynomial operator and giving preference to the points with high value of the gradient.

$$\tilde{K}(x,y) = I_{xx}I_y^2 + I_{yy}I_x^2 - 2I_{xy}I_xI_y \tag{2.3}$$

Lindeberg proposed also to use this differential operator with the scale-space representation of an image $L(x, y, t)$ where t is the scale parameter. For each t we use a gradually blurred image for corner detection. In this way we achieve corner detection for different scales.

Förstner and Harris detectors are another family of corner detectors which only use first-order derivatives. Förstner [8] uses local maxima of

$$F(x,y) = \frac{\bar{I}_x^2 \bar{I}_y^2 - (\overline{I_x I_y})^2}{\bar{I}_x^2 + \bar{I}_y^2} \tag{2.4}$$

for corner determination.

Harris detector [9] [6] is built considering the following matrix

$$M = \begin{pmatrix} I_x \hat{} I_x & I_y \hat{} I_x \\ I_y \hat{} I_x & I_y \hat{} I_y \end{pmatrix} \tag{2.5}$$

and the following operator

$$R = det(M) - k(trace(M))^2 \tag{2.6}$$

Î stands for I filtered by a Gaussian filter. In this matrix, a feature point (corner) is characterised with rank 2, edges with rank 1 and homogeneus regions with rank 0. Instead of computing the eigenvalues Harris proposed to use the R operator. Therefore positive values of R point out corners and negative ones edges. k is a parameter set typically to 0.04 as proposed by Harris [10], the trace term in the equation is used to eliminate strong eigenvalue contour points.

Several papers, see [9] [8] for instance, refer Harris as the corner detector with best performance (in terms of repeatability, see Section 2.2.1) although it has more computational cost than others.

2.2 Evaluation of Feature Point Extraction

Schmid et al. use two criteria for the evaluation of feature point extractors: repeatability and information content [9]. Abdeljaoued uses also localisation criterion for evaluation [6]. Repeatability is widely used [6] [9] [11] [12] with multiple definitions. In this section these criteria for evaluation are defined.

2.2.1 Repeatability

Repeatability rate for two consecutive images is the number D of detected points which are repeated over the total number N of different points detected in the common part of both images.

$$r = \frac{|D|}{N}, 0 \leq r \leq 1 \tag{2.7}$$

That way we penalise if a point detected in the first image is not detected in the second and vice-versa. In which case we do not take into account the order in the sequence of two consecutive frames. On the other hand, we do not penalise for points outside the viewing area. This definition emphasises extraction performance evaluation.

Using this definition, repeatability rate for Figure 2.2 would be $r = \frac{4}{5}$. There are five different points in the common part of both images but only four points are repeated.

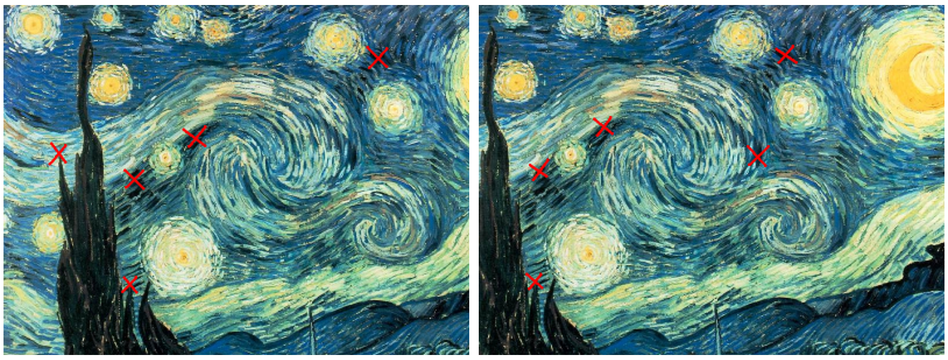


Figure 2.2: *Example of two images for Repeatability computation. Only 4 out of 5 points in the common area are repeated.*

Other definitions for repeatability rate do not modify the numerator but redefine the denominator. It can be the total number of points detected [11] [6] including points outside the common part of both images (which leads to penalise for points that are not detected because they do not belong to the second image, for example), or the minimum [12] or the mean [13] of the number of points detected in the common part of both images (more optimistic).

A repeated point, sometimes, is not detected exactly at the same corresponding position, but in a neighborhood. If we denote the radius of the neighborhood by ε, we call ε-repeatability to the repeatability within this neighborhood.

$$r(\varepsilon) = \frac{|D(\varepsilon)|}{N}, 0 \leq r(\varepsilon) \leq 1 \tag{2.8}$$

2.2.2 Reliability

Ichimura introduces the idea of feature point reliability. One feature point is reliable if it can be tracked for several consecutive frames [14]. We choose a number of frames to compute it (F_l) and being F_c the number of consecutive frames a point is tracked, the reliability is:

$$R_p = \frac{F_c}{F_l}, 0 \leq R_p \leq 1 \tag{2.9}$$

2.2.3 Localisation

The localisation criterion is a measure of how accurate are the positions of extracted points with respect to the corresponding real points [6].

We can use the root mean square error (RMSE) to compute this criterion:

$$RMSE = \sqrt{\frac{1}{N}\sum_{n=1}^{N}[(x_{n0} - x_n)^2 + (y_{n0} - y_n)^2]} \tag{2.10}$$

Where N is the total number of extracted points, (x_{n0}, y_{n0}) are the coordinates of the real feature point and (x_n, y_n) the coordinates of the extracted feature point.

2.2.4 Complexity

The complexity of a detector, or the speed at which identifies features depends heavily on the implementation. In augmented reality computational cost is very important, like in any real time application. Therefore, complexity is a critical criterion.

2.2.5 Other

Vincent and Laganière summarise several evaluation criteria in [15]. We could add, to the list presented above, the following:

- Accuracy, which can be evaluated with localisation criterium.
- Robustness or insensivity to noise.
- Stability, meaning that a point should be detected under geometrical and/or illumination transformations.
- Controllability, or the number of parameters required and their influence.

Chapter 3

Feature Point Characterisation

In video-based markerless tracking for AR, one challenging problem is to achieve more accuracy in the feature point correspondences between frames. A photometric test consists in the comparison of the neighbourhood of two feature points from different frames. It is done to ensure matching in the tracking step by preventing the association of two features when they do not resemble each other.

One simple method to implement a photometric test is the Normalised Cross Correlation coefficient, NCC, which is a widely used metrics to compare and also for classification purposes.

Another approach is to implement photometric test using descriptors. Local descriptors are becoming more and more used for texture classification and object recognition. Local descriptors should be distinctive and robust to changes in viewing conditions and errors of the feature point extractor. They also should be selective to local patterns as well as invariant to rotation, scale and translation.

In this chapter descriptors based in Gaussian derivatives and Gabor functions are explained. A solution to rotation variance in such descriptors is proposed. In addition, a definition for normalised cross correlation is given.

3.1 Normalised Cross Correlation Coefficient

Normalised cross correlation coefficient is a similarity measure like, for example, the Sum of Squared Differences, SSD, or Euclidean distance. When the two elements to compare are identical in absolute value, their NCC is equal to one. The most common definition for NCC is:

$$c(f,g) = \frac{\sum\sum(f-\bar{f})(g-\bar{g})}{\sqrt{\sum\sum(f-\bar{f})^2}\sqrt{\sum\sum(g-\bar{g})^2}} \tag{3.1}$$

f and g are the two elements (in our case images) to compare, and $\bar{f}$ and $\bar{g}$ stand for their means.

Applied to image, NCC is invariant to translation but performance degrades when relative rotations occur. Ravela stablishes failure beyond 15 degree rotation[16]. But in a frame-to-frame tracking basis we will not find, in principle, such rotations.

NCC has low computational cost. Some tracking techniques use it as the only method for data association. Nevertheless, NCC does not achieve very good performance in these cases due to its rotation variance and because we can find images with repeated patterns in it which can cause mismatches. We reduce the problem if we apply NCC for photometric test to ensure matching in small regions only.

3.2 Gaussian Derivatives Descriptors

The first, second and third directional (0 degrees, x axis, standard deviation $\sigma = 1$) derivatives of the Gaussian, can be written as:

$$G_1^0 = -xe^{-\frac{x^2+y^2}{2}} \tag{3.2}$$

$$G_2^0 = (1-x^2)e^{-\frac{x^2+y^2}{2}} \tag{3.3}$$

$$G_3^0 = x(3-x^2)e^{-\frac{x^2+y^2}{2}} \tag{3.4}$$

for other orientations we can apply rigid rotations.

In Figure 3.1 plots of the three first derivatives of the Gaussian are shown.

The response of the local image patch at a feature point coordinates to one of the set of filters, is obtained via the following convolution operation (valid convolution) in [17]:

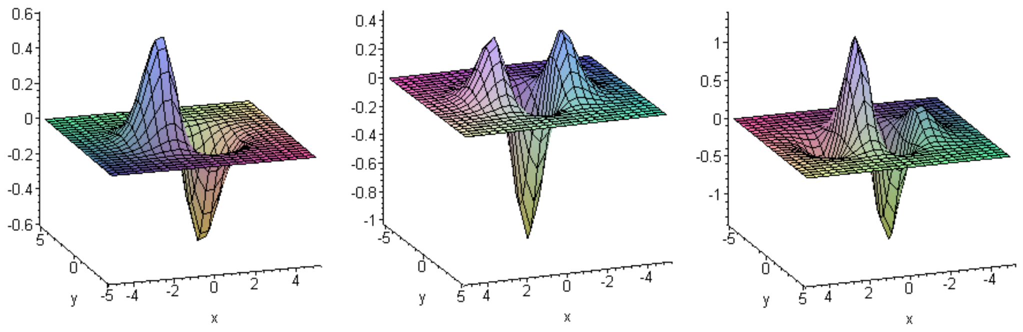


Figure 3.1: *Spatial-domain 3d plots of the three first derivatives of the gaussian*

$$r_{i,j}(x_0, y_0) = \iint G_i^{\theta_j}(x_0 - x, y_0 - y) I(x, y) dx dy \tag{3.5}$$

the local descriptor is the vector of responses for a given number of orientations.

Schmidt and Mohr propose a ”Local Jet”, a descriptor formed by combinations of differential invariants obtained from convolution with Gaussian derivatives up to third order [5]. They do a multiscale approach with Gaussian variance σ values around one.

The response of these filters is not invariant to rotation. Narrower widths of the Gaussian lead to more selectivity but even less invariance. However, invariance can be achieved by other means (see Section 3.4).

Gaussian derivatives descriptors are one of the approaches to texture analysis. They are used also for low level characterisation of image content for encoding purposes.

3.3 Gabor Descriptors

Complex Gabor functions can be expressed in real and imaginary parts as:

$$h_e(x, y; f, \theta) = g(x, y) cos(2\pi f(x cos\theta + y sin\theta)) \tag{3.6}$$

$$h_o(x, y; f, \theta) = g(x, y) sin(2\pi f(x cos\theta + y sin\theta)) \tag{3.7}$$

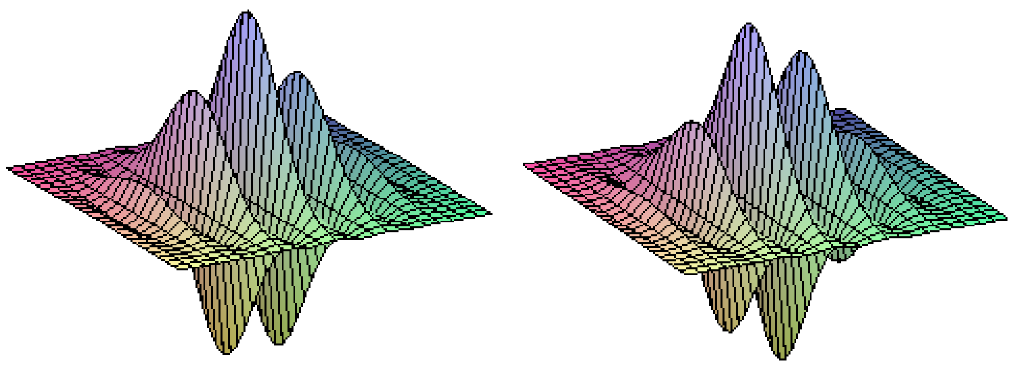


Figure 3.2: *Spatial-domain 3d plots of real and imaginary parts of Gabor Filters*

where g(x,y) is the Gaussian function, f the spatial frequency and θ the orientation.

The Gabor 2D functions are widely used in texture analysis and characterisation for many reasons. One of them is that they have tunable orientation and radial frequency bandwidths and tunable centre frequencies.

Gabor filters can be seen as different modulations of the Gaussian function. They can be used to approximate derivatives of the Gaussian function, depending on the parameters choice.

Although many functions may be used for multiresolution space-frequency analysis (Gaussian derivatives), Gabor functions are particularly useful as they they provide optimal spatial resolution for a given bandwidth and they are considered to share many properties with the human visual system. Human visual system uses a set of parallel mechanisms or channels (tuned to a specific narrow band of spatial frequency) [18] [19].

In Figure 3.2 plots of real and imaginary Gabor functions are shown for 0 degree orientation and spatial frequency set to 1.

Tan and other authors build the local descriptor filtering separately the intensity function with real and imaginary filters, and then computing the root of the squared sum [18] [20] [21]. Tuceryan and Jain use only the real part for filtering [22].

3.4 Rotation Invariance

If we have two equal image patches but one is a rotated version of the other, and we want to consider them the same, we have to use invariant-to-rotation descriptors.

Tan et al. propose descriptors based on Gabor filters responses [18] [23]. The response of a rotated version of an image would be a shifted version of the former response. But the Fourier transform is invariant to translation. Therefore rotation invariance of the descriptor can be achieved via Fourier transformation.

Another strategy is to estimate the orientation and then apply the suitable steerable filter [24]. In [17] we can find the comparison of some implementations of this philosophy. One drawback is that we have to be accurate while estimating the orientation and we do not use the same filter for all orientations but a combination of a basic set. Several papers [25] [26] [27] [13] propose invariant Gaussian descriptors using steerable filters.

Chapter 4

Feature Point Tracking

Tracking applications include surveillance (traffic, security, space), RADAR and intelligent information retrieval, among others.

In this chapter, a brief description of existing tracking algorithms is given. This chapter is intended to ease the comprehension of further modifications to these algorithms, for a complete review of tracking theory see, for instance, [28].

In the following, some tracking techniques will be introduced as well as some important steps commonly used in many algorithms. Then, a short introduction to pose estimation will be given, as it will also be used for tracking purposes.

4.1 Tracking Basics

Tracking basics shall begin with an introduction to the terminology adopted. In this document we will use extensively the following:

- State: Characteristics vector of a target (feature point in our case) that summarise the past and permits us to predict the future. Position and velocity are typical components. Often, it cannot be measured directly.
- Measurement: Sensor observation related to the state of a target, position, for instance. In our case, the sensor is a camera.
- Track: list of measurements over time which correspond to the same target.

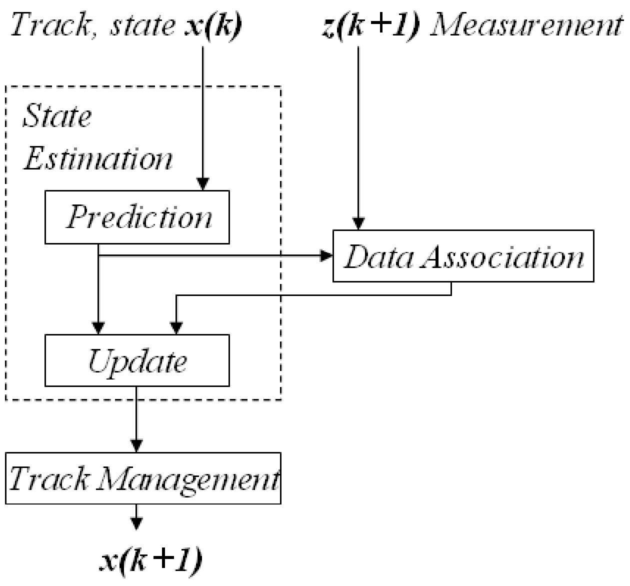


Figure 4.1: *Typical tracking system operation*

A typical multitarget tracking system predicts current state from previous state. The estimate is improved by the incorporation of a measurement. Data association decides which measurements shall be combined with which states. This proceeding is to be done every frame: tracks are updated, created or resumed, which involve a track management step.

4.1.1 Kalman Filter

State estimation can be implemented with Kalman filter which combines a motion model with measurements in order to obtain the best target state estimate.

We assume the following motion model:

$$\mathbf{x}(k+1) = \mathbf{F}(k+1)\mathbf{x}(k) + \mathbf{v}(k) \tag{4.1}$$

where $x(k+1)$ is the state vector at instant $k+1$, F is the transition matrix between frames k and $k+1$, and v is a zero-mean white gaussian noise sequence, whose covariance matrix is $Q(k)$.

The measurement is modelled by:

$$\mathbf{z}(k+1) = \mathbf{H}(k+1)\mathbf{x}(k+1) + \mathbf{w}(k+1) \tag{4.2}$$

where z is the measurement vector, H the measurement matrix and w a zero-mean white gaussian noise sequence, whose covariance matrix is $R(k+1)$.

Predicted state estimate and measurement are obtained applying expectation conditioned the measurements history (Z^k). Which leads to these expressions:

$$\hat{\mathbf{x}}(k+1|k) = \mathbf{F}(k)\hat{\mathbf{x}}(k|k) \tag{4.3}$$

$$\hat{\mathbf{x}}(k|k) = E\{\mathbf{x}(k)|Z^k\} \tag{4.4}$$

$$\hat{\mathbf{z}}(k+1|k) = \mathbf{H}(k+1)\hat{\mathbf{x}}(k+1|k) \tag{4.5}$$

Predicted state covariance is:

$$\mathbf{P}(k+1|k) = \mathbf{F}(k)\mathbf{P}(k|k)\mathbf{F}^{\mathbf{T}}(k) + \mathbf{Q}(k) \tag{4.6}$$

The predicted measurement covariance, $S(k+1)$, defined as:

$$\mathbf{S}(k+1) = E\{\tilde{\mathbf{z}}(k+1|k)\tilde{\mathbf{z}}^T(k+1|k)|Z^k\} \tag{4.7}$$

where $\tilde{\mathbf{z}}(k+1|k) = \mathbf{z}(k+1) - \hat{\mathbf{z}}(k+1|k)$ is the estimation error, can be computed via the measurement matrix, the predicted state covariance and the measurement noise covariance as follows:

$$\mathbf{S}(k+1) = \mathbf{H}(k+1)\mathbf{P}(k+1|k)\mathbf{H}^T(k+1) + \mathbf{R}(k+1) \tag{4.8}$$

Data association provides a measurement, $z(k+1)$, the difference with the predicted measurement is called innovation $\nu(k+1)$:

$$\nu(k+1) = \mathbf{z}(k+1) - \hat{\mathbf{z}}(k+1|k) \tag{4.9}$$

The measurement updated state estimate, $\hat{\mathbf{x}}(k+1|k+1)$, is computed combining predicted state estimate and innovation. Innovation is weighted by filter gain, W, in order to give preference to the measurement if the innovation is small (which is the same as an small measurement covariance S):

$$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + \mathbf{W}(k+1)\nu(k+1) \tag{4.10}$$

$$\mathbf{W}(k+1) = \mathbf{P}(k+1|k)\mathbf{H}^{\mathbf{T}}(k+1)\mathbf{S}^{-\mathbf{1}}(k+1) \tag{4.11}$$

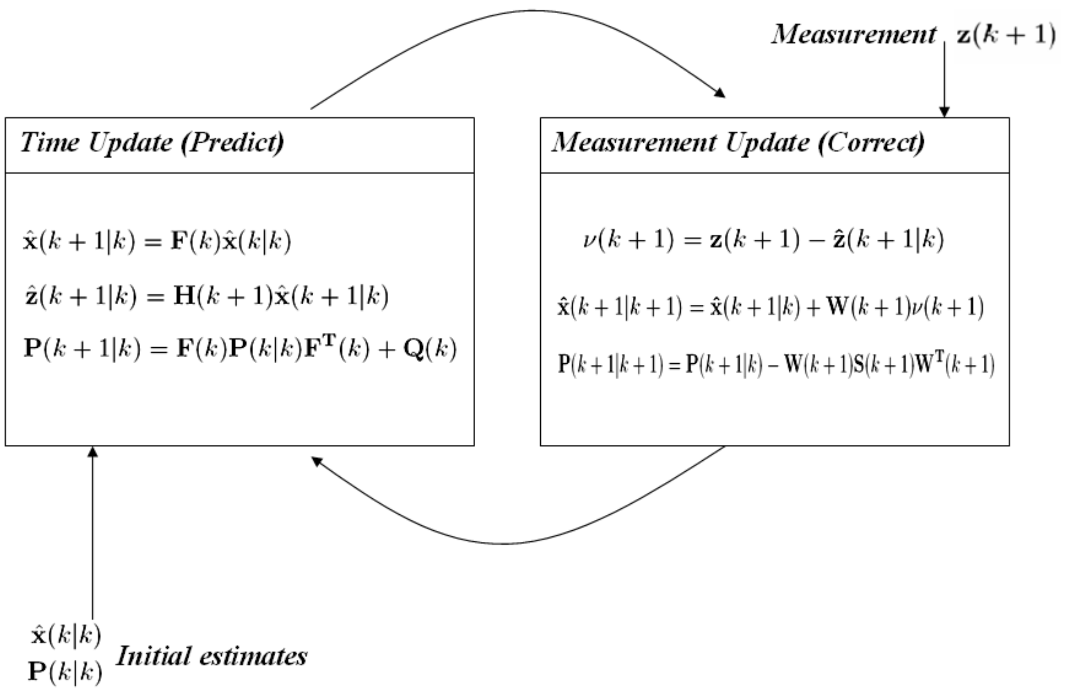

Measurement $\mathbf{z}(k+1)$
Time Update (Predict)
$\hat{\mathbf{x}}(k+1|k) = \mathbf{F}(k)\hat{\mathbf{x}}(k|k)$
$\hat{\mathbf{z}}(k+1|k) = \mathbf{H}(k+1)\hat{\mathbf{x}}(k+1|k)$
$\mathbf{P}(k+1|k) = \mathbf{F}(k)\mathbf{P}(k|k)\mathbf{F}^{\mathbf{T}}(k) + \mathbf{Q}(k)$
Measurement Update (Correct)
$\nu(k+1) = \mathbf{z}(k+1) - \hat{\mathbf{z}}(k+1|k)$
$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + \mathbf{W}(k+1)\nu(k+1)$
$\mathbf{P}(k+1|k+1) = \mathbf{P}(k+1|k) - \mathbf{W}(k+1)\mathbf{S}(k+1)\mathbf{W}^{\mathbf{T}}(k+1)$
$\hat{\mathbf{x}}(k|k)$
$\mathbf{P}(k|k)$
Initial estimates

Figure 4.2: *Kalman operation: Prediction and Correction*

The main drawback of Kalman filter is the single motion model assumption. In most cases, target motion has some occasionally unpredictable changes or maneuver. Therefore, a single motion model is not suited for this kind of targets.

In this section, an extremely brief explanation of Kalman filter has been shown. For a complete version and for beginners to Kalman filtering "An Introduction to the Kalman Filter" by Welch and Bishop [29] is very useful.

4.1.2 IMM Filter

In order to avoid single motion model assumption, interactive multiple model switches between several models for target motion depending on the probability/suitability of each model.

IMM implementation contains a set of filters working in parallel. For example, with two filters, one is the model for small maneuvering and the other for fast maneuvering.

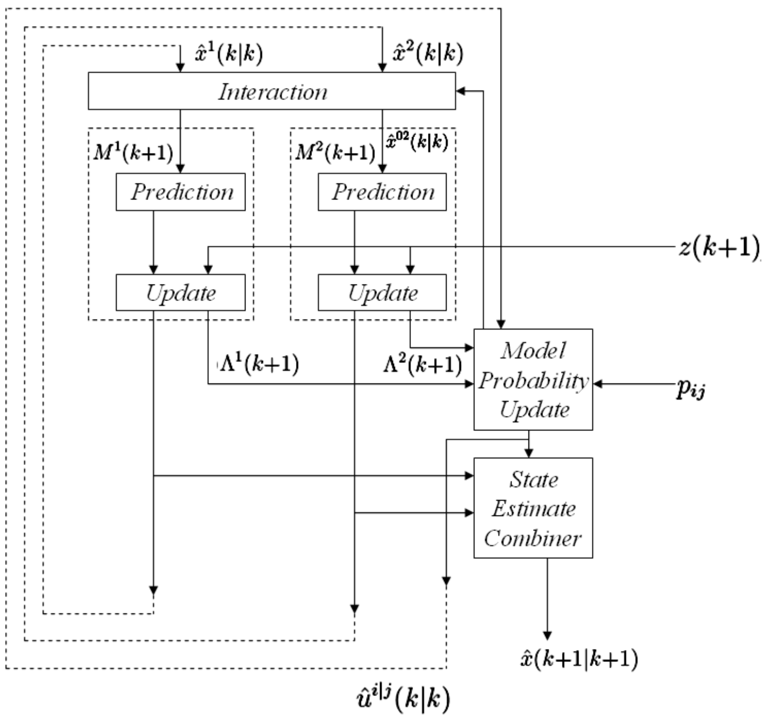


Figure 4.3: *IMM scheme*

In Figure 4.3 one iteration of the IMM filter is shown. There are two motion models, which are implemented with two filters, M^1 and M^2. Each iteration involves a set of equations, related to four steps:

1. Interaction/mixing: The state estimates of both models are combined taking into account model probability, u. Model transitions/switching is modelled by

Markov processes and p_{ij} is the probability of switching from model M^i to M^j

2. Filtering: This step is like any Kalman filter, one per model. The likelihood of each filter can be computed as follows:

$$\mathbf{\Lambda}(k+1) = \frac{1}{\sqrt{det[2\pi \mathbf{S}(k+1)]}} e^{-\frac{1}{2}\nu^{\mathbf{T}}(k+1)\mathbf{S}^{-1}(k+1)\nu(k+1)} \tag{4.12}$$

3. Model probability update: new model probabilities are obtained weighting the old ones with filter likelihoods.

4. Combination: The overall state estimate is obtained combining the estimates of both filters like in the mixing step.

IMM filter allows us to use multiple motion models for the target's motion. Therefore, the main drawback of Kalman filter is partially solved.

4.1.3 Track Management

Targets to track in a sequence, may appear at the beginning of the sequence or later, they may disappear either for ever or for some frames due to an occlusion or non-detection. Therefore, as the number of tracks varies continuously, track management becomes important. A mechanism of track management is expected to be capable of initiate, delete and continuate tracks. This last aspect is discussed in detail in Section 4.2.

4.1.4 Data Association

We call data association to the step of relating new measurements with existing tracks. A data association algorithm can be as simple as Nearest Neighbour filter. Which is one of the most used and simply assigns the closest measurement to the prediction to that track. Even more complex algorithms (based on assignment problems or on different hypothesis) can use gating procedure to enhance performance.

Gating

In order to fasten data association step, one strategy is to limit the search region for the measurement. The gating procedure is used to select candidates from the measurements to match with a certain track. The gate, if using Kalman filters, is defined as follows:

$$\mathbf{V}(k+1,\gamma) = \{\mathbf{z}(k+1) : \nu^{\mathbf{T}}(k+1)\mathbf{S}^{-\mathbf{1}}(k+1)\nu(k+1) < \gamma\} \tag{4.13}$$

where $V(k+1)$ is the gate at time $k+1$, γ is the threshold and ν and S come from Equations 4.9 and 7.5. The product $\nu^T(k+1)\mathbf{S}^{-\mathbf{1}}(k+1)\nu(k+1)$ is called normalised innovation squared. The shape of the gate is elliptical and is centered around the estimated position, as it can be seen on Figure 4.4.

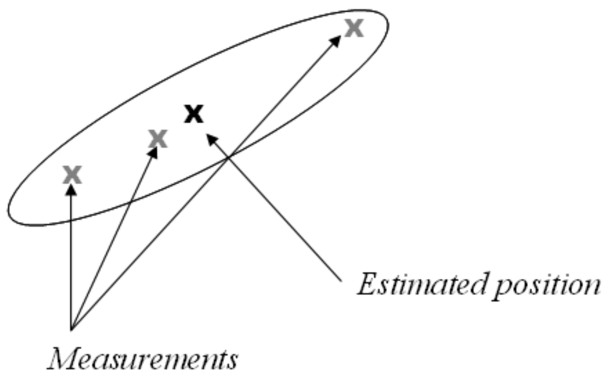


Figure 4.4: *Typical ellipsoid gating region*

MHT Filter

Multiple hypothesis tracking (MHT) is a tracking technique where different hypothesis on data association are generated. Hypothesis such as a new target, a false alarm or a continuation of an existing track are contemplated. The decision is taken a posteriori, hypothesis that match future measurements are selected.

The number of hypothesis increases exponentially-like with the number of measurements. Adding the fact that the decision is held for a period of time leads to a very slow algorithm. Absolutely incompatible with real time.

Cox and Hingorani discuss an efficient implementation of MHT algorithm in [30] applied in visual tracking. Although they implement MHT efficiently, its principal problem is complexity.

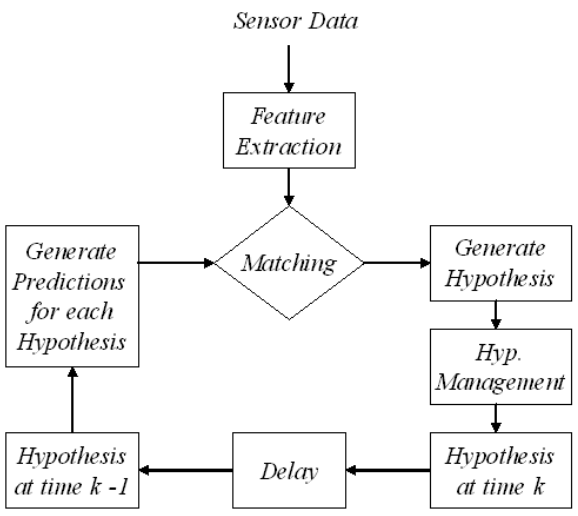


Figure 4.5: *MHT scheme*

4.2 Track Continuation

MHT filter contains track continuation capabilities, as is one of the hypothesis [30]. However, when using IMM filter and nearest neighbour association there are some alternatives for track continuation:

- Extrapolation, consists in using the prediction as far as there are not measurements.
- Use of prediction if the track is good enough. Which means that a track quality indicator (TQI) shall be defined.
- Interpolation (or slave measurements [31]) between two real measurements, those would be the measurements before and after the measurement absence.

The former and the latter are inadequate in our application. The former because when the absence, due to occlusion for example, lasts various frames, prediction deviate much from true motion. The latter implies that we have to wait until a measurement reappears to be able to interpolate the motion. Absolutely incompatible with real time.

Abdeljaoued proposes to use a modified version of the likelihood as a track quality indicator [6]. Modified log-likelihood function is defined as the exponent of the likelihood function taken from IMM filter.

$$\mathbf{\Lambda}(k+1) = \frac{1}{\sqrt{det[2\pi\mathbf{S}(k+1)]}} e^{-\frac{1}{2}\nu^{\mathbf{T}}(k+1)\mathbf{S}^{-\mathbf{1}}(k+1)\nu(k+1)} \tag{4.14}$$

$$\mathbf{p}[Z^k] = \prod[\sqrt{det[2\pi\mathbf{S}(k+1)]}]e^{-\frac{1}{2}\sum\nu^{\mathbf{T}}(k+1)\mathbf{S}^{-\mathbf{1}}(k+1)\nu(k+1)} \tag{4.15}$$

$$\lambda(k) = \lambda(k-1) + \nu^{\mathbf{T}}(k+1)\mathbf{S}^{-\mathbf{1}}(k+1)\nu(k+1) \tag{4.16}$$

Λ is the likelihood of the model, $p[Z^k]$ is the likelihood function and $\lambda(k)$ is the modified log-likelihood, which can be calculated recursively adding the normalised innovation squared. When the modified log-likelihood exceeds a threshold, it is an indication that the track is not good enough. It may come due to mismatches or to incorrect predictions.

We can use this indicator for track continuation. When we continuate a track, that is we use prediction, we can penalise the track quality indicator by adding the threshold used in the gating process, γ, instead of the normalised innovation squared.

$$\lambda(k) = \lambda(k-1) + \gamma \tag{4.17}$$

Therefore, continuated tracks have worse quality indicator than the ones updated regularly with measurements.

4.3 Pose Estimation Applied to Tracking

Pose estimation is the third step in a typical AR system, after feature point extraction and tracking (see Section 5.2). The features in a sequence have a related motion between them, they do not move independently as they belong to the same scene. However, we are calculating a motion model for each point that is tracked, without taking into account that we have already an estimation of the camera movement.

Therefore, we can use data from pose estimation for improving feature point tracking as we are using sequence without moving objects. A brief description of the idea of homography and some examples of how it can be used are given in this section.

4.3.1 Homographies

An homography is a mapping between two images. Hartley and Zisserman define projective transformation or homography as a linear transformation on homogeneus 3-vectors that can be represented with a non-singular 3x3 matrix [32]:

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \tag{4.18}$$

This model, $x' = Hx$, describes image motion between two consecutive frames. It is exact when camera motion is pure rotation or the scene viewed is planar. However, it is a good approximation when the scene is nearly planar, or the movement of the camera is small and the viewed scene is distant.

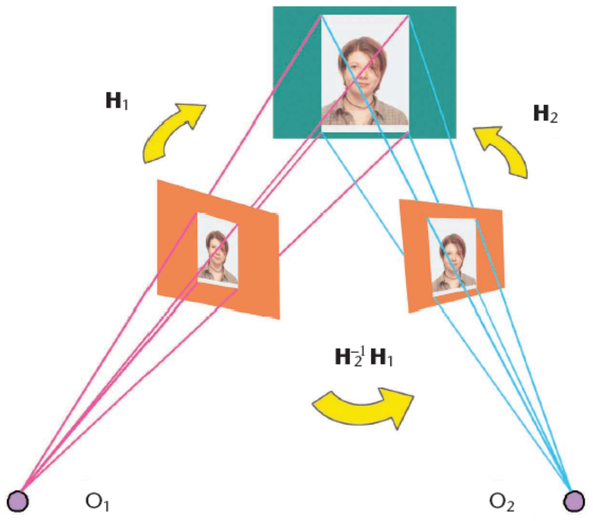


Figure 4.6: *An Homography relates two views of a planar scene. Figure taken from [1]*

One particular case in projective transformations are affinities. An affine transformation is a non-singular linear transformation followed by a translation. With matrix representation:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{4.19}$$

where t_x and t_y are the translation components. The affine A matrix, defined by the four a_{xx} components, describes the rotation and/or deformation of the image. With affine transformations, orthogonal world lines are not seen as orthogonal. Nevertheless parallel lines keep their condition. As an example, a circle is imaged as an ellipse.

4.3.2 Use of Pose Estimation Applied to Tracking

Aron et al. present in [2] an hybrid approach for markerless tracking in real time. They use an inertial sensor and couple sensor and camera data in order to obtain accurate tracking.

The system works as any classical video-based tracker; but when information from the inertial sensor is available, it is used in these 3 ways:

1. Position prediction of research windows. A predicted homography is obtained from the rotation provided by the sensor. If H_s is the predicted homography, research window shall be centred at position $m' = H_s m$ where m is the current point and m' the expected.
2. Refined research regions. As FP transformations are predictable and a covariance matrix is easily obtained. Research areas can be shaped elliptically using metrics such as Mahalanobis.
3. Deformation of correlation windows. Using the predicted homography, correlation window can be reshaped to match more precisely.

Vincent and Laganière propose a matching scheme where for all possible pair of Feature Points to match, the affine transformation between them is estimated using homographies; like in the third use of Aron's method explained above. Therefore, when similarity measure is made in the vicinity of the FP in the first view to find the matched FP in the second, the use of the transformation improves greatly the

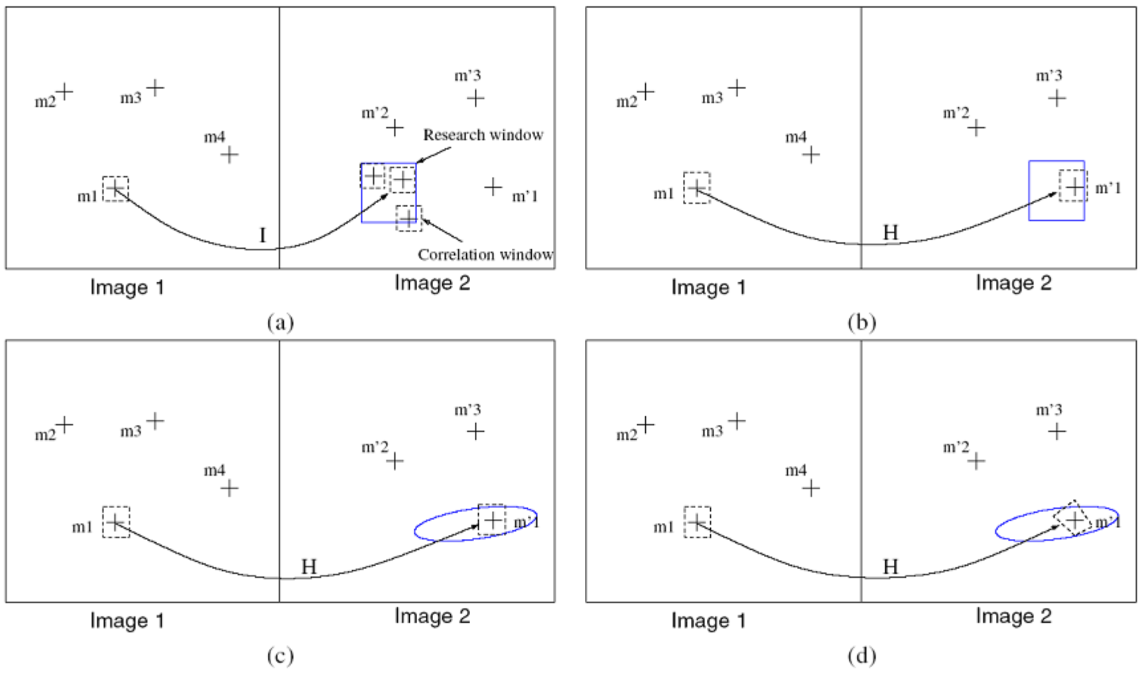


Figure 4.7: *Different methods for matching feature points. (a) Classical Video-based matching. (b) Position of the search area is predicted using homographies. (c) Research window is obtained using statistic distances. (d) Correlation window is reshaped using the predicted homograhpy. [2]*

measure. [15]. As a summary, it would be a mixture between case (a) and (b) in Figure 4.7.

Part II

System design

Chapter 5

System Overview

Merging reality and virtual objects in a realistic way requires an accurate computation of the relative position of virtual objects to the viewer's (the camera) pose. If we use markerless tracking, we only use natural features in the image. Once these features are extracted and tracked, we are able to estimate the pose. Therefore, we need some hardware components to capture and display images and algorithm modules to extract and relate points which are used for the pose estimation.

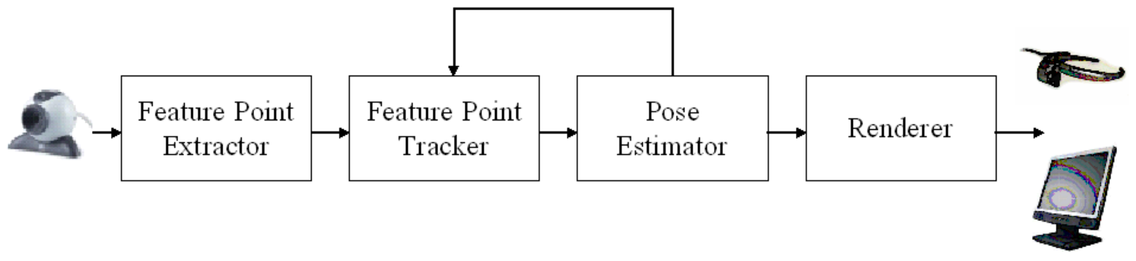


Figure 5.1: *Architecture of an AR system*

As it can be seen from Figure 5.1, our AR system consists in four steps between the capture of an image by the camera and its rendering, together with virtual objects on it, in a Head Mounted Display (HMD) or a conventional monitor. The former two compound the video tracker, which extracts and track natural features. The third computes the pose estimation with the parameters provided by the video tracker. The fourth one creates in the proper place the virtual image mixed with the real scene.

5.1 Video Tracker

Our aim is to enhance feature point extraction and tracking steps. Video tracker role is to sense the real world, and provide the parameters needed for the computation of the actual head pose.

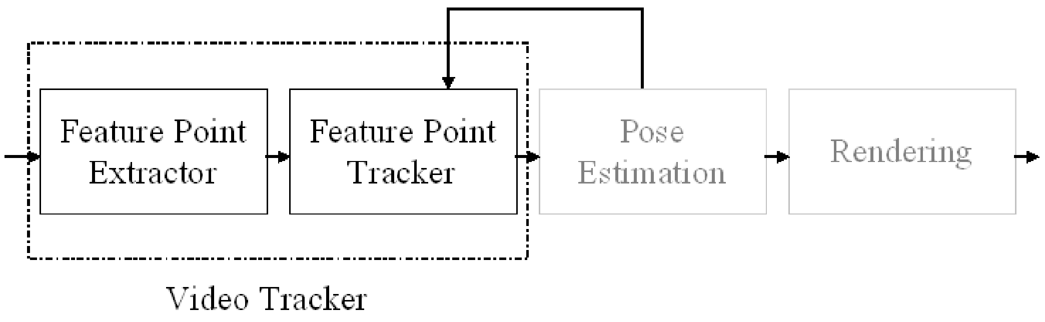


Figure 5.2: *Our goal, improve feature point extraction and tracking*

Current implementation of the system contains an implementation of Lindeberg detector as a feature point detector. This implementation as well as a new one, Harris corner detector, will be explained in Chapter 6.

Chapter 7 deals with feature point tracker implementation. Two more methods for feature point characterisation will be implemented and explained as well as current cross correlation coefficient.

Current feature point tracker is implemented using IMM filter for state estimation and Nearest Neighbour (NN) for data association with gating procedure (see Figure 5.3). Pose estimation feedback will be added for improving data association as seen in Figure 5.4. Continuation capabilities in track management will be added, too.

5.2 Pose Estimator

Pose estimation of the user allows to place the augmentation in the scene correctly. If the camera is attached to the head, like in a Head Mounted Display, rotation and translation of the camera represent the user's motion.

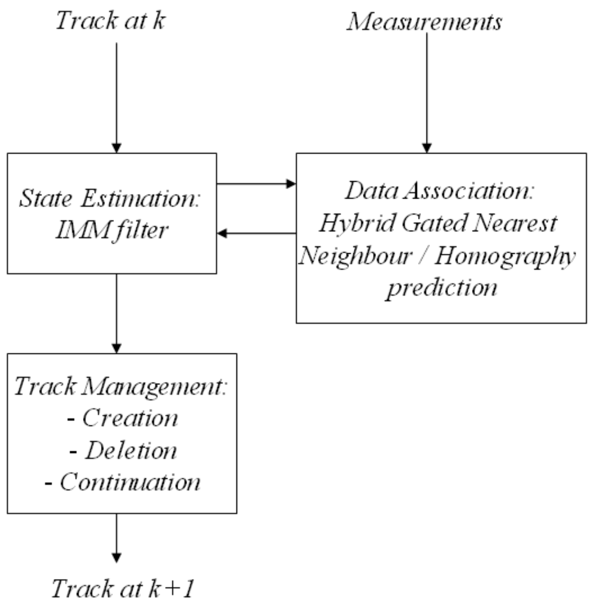

Track at k
Measurements
State Estimation: IMM filter
Data Association: Hybrid Gated Nearest Neighbour / Homography prediction
Track Management:
- Creation
- Deletion
- Continuation
Track at k+1

Figure 5.3: *Proposed tracking technique*

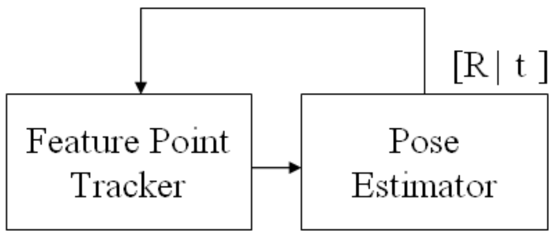


Figure 5.4: *Pose estimation feedback*

Our system performs pose estimation using homographies (see Section 4.3.1). For each new frame, using the information provided by the tracker, pose of the camera is estimated finding the homography that relates the tracked points from the former and current frames. We employ a robust statistical method, RAndom SAmple Consesus (RANSAC) [33], to find a good estimation. RANSAC procedure involves choosing different subsets of matches at random and compute the homograhies. The homography with the largest number of matches that accomplish this transformation (inliers) is chosen.

5.3 Renderer

Rendering is the process of generating an image from a model, it contains geometry, viewpoint, texture and lighting information. Once the pose is estimated, we are able to draw the rendered virtual object (with the corresponding rotation and translation)on the real image captured by the camera. Finally this combination is displayed.

Chapter 6

Feature Point Extractor

Feature Point Extraction is the first step in our system. It is in charge of extracting natural features such as corners from the image captured by the camera. A poor performance of this step degrades the performance of all the steps that are to be followed afterwards.

As seen in Chapter 2, Harris corner detector is widely used and credited as the one with best performance. In the former implementation of the system, Lindeberg detector was chosen.

This chapter explains in detail the current implementation of the system (Lindeberg detector) and proposes Harris as a new Feature Point extractor. An evaluation of the performance of both detectors will be discussed in Section 8.2.

6.1 Lindeberg Detector

In Figure 6.1 we can see the implementation of Lindeberg detector. Recalling from Chapter 2, the differential operator used by Lindeberg algorithm is:

$$\tilde{K}(x,y) = I_{xx}I_y^2 + I_{yy}I_x^2 - 2I_{xy}I_xI_y \tag{6.1}$$

By omitting the multi-scale approach, we simplify and lighten the Feature Point Extractor. Reducing computational cost is very important in Augmented Reality, like in any real time application.

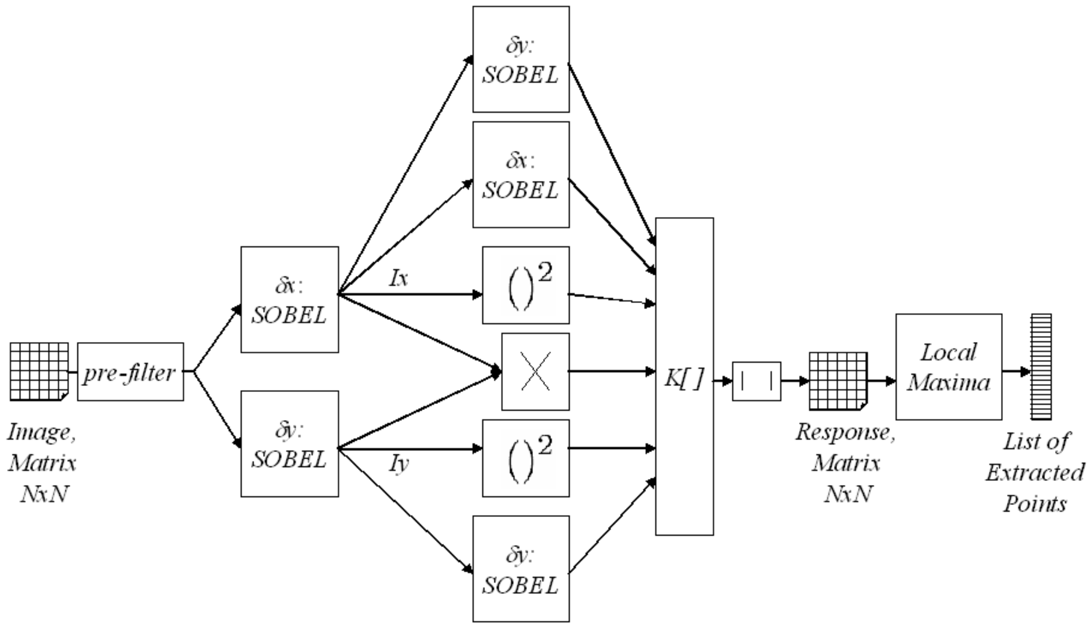


Figure 6.1: *Implementation of Lindeberg Corner Detector*

There is a previous filtering step (blurring) to eliminate spurious in the image caused by noise. Local extrema are found searching maxima of the absolute value in small regions of the response. The number of extracted features is controlled using a threshold. By searching maxima this way, we ensure that features are extracted throughout the captured image.

Derivatives are computed applying Sobel filters. Sobel filters consist in two masks that once convolved with the image compute approximatively the gradient of a discreet image. There are other approaches but Sobel filters are widely used in image processing because of their performance. Those masks are:

$$\begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix} \qquad \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \tag{6.2}$$

Therefore, we can detect separately horizontal and vertical variations, respectively, by applying these masks to the image.

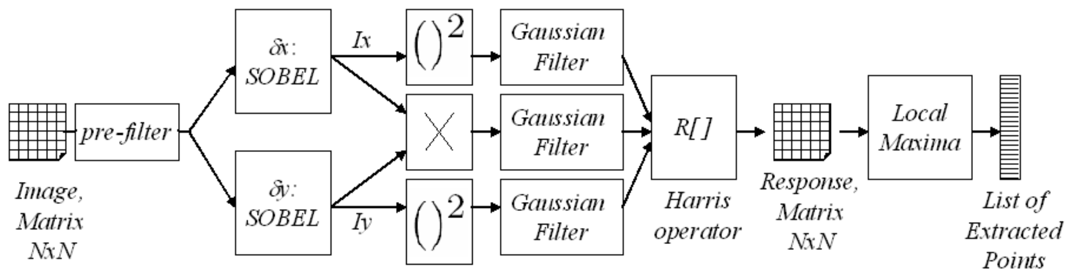


Figure 6.2: *Implementation of Harris Corner Detector*

6.2 Harris Detector

From Equation 2.5 we define the following operation to be implemented.

$$R = \hat{I}_x^2 \hat{I}_y^2 - \widehat{I_x I_y}\,\widehat{I_x I_y} - 0.04(\hat{I}_x^2 + \hat{I}_y^2)^2 \tag{6.3}$$

In Figure 6.2 we can see the implementation of Harris corner detector. Here we apply blurring twice, firstly to soften nosy images and secondly as it is required by Harris detector. Local maxima is computed in a similar way of Lindeberg's implementation, that is defining a threshold to control the number of desired extractions and looking for maxima in small regions. This time though, if we are looking for corners we just have to search local maxima. Local extrema would lead us to find corners and edges (as seen in Section 2.1).

Harris detector computational cost cannot be much higher than Lindeberg's. Certainly, we add Gaussian filtering, but we only compute the first order derivatives. Therefore, we apply Sobel filtering only once per axis.

Chapter 7

Feature Point Tracker

Our starting point is an already implemented augmented reality system as seen in Chapter 5. For the tracking step, some new techniques are designed and implemented in order to improve the system's performance.

The parts of the tracking system that are going to be modified are photometric test, track management (in the sense of adding track continuation capabilities to the system) and the gating procedure (benefiting from pose estimation data). This chapter explains in detail current implementation in this areas, if any, as well as shows the implementation of the improvements proposed.

7.1 Photometric Test

In feature point tracking, it is interesting to ensure correct matches with a photometric test which consists of comparing the neighbourhoods of the feature points within the candidates to match. One simple way to compare two feature points is to compute the NCC.

Another possible approach is to characterise feature points with descriptors and compare them. Most invariant feature descriptors are constructed using Gabor filters or Gaussian derivatives as seen in Chapter 3.

A comparison of these three technique's performance can be seen in Section 8.3.4.

7.1.1 Normalised Cross Correlation Coefficient

As it is said above, NCC is one simple method for Photometric Test. In the current implementation of the system, it is computed applying the definition given in Section 3.1. The 3x3 pixel neighbourhood of a Feature Point is compared with the 5x5 pixel neighbourhood of the extracted candidates of the next frame, as it is proposed by Cox and Hingorani [30] among others. In Section 8.3.1 the size of the patch is tested, giving as a result that the choice 5x5 is adequate.

7.1.2 Gaussian Derivatives Descriptors

In order to compare two Feature Points, a neighbourhood of each is captured in a patch and then a local descriptor which describes these patches is computed.

Gaussian descriptors are computed following the idea of [17] which implies the use of valid convolution (that can be seen as normal convolution omitting zero padding, so calculus outside the patch is not done) which has far lower computational cost than a normal convolution. Patches are sized 5x5 pixels or 7x7. Lower patches would not contain enough information as they will be too small; bigger patches may include neighbouring information of another feature point leading to more correlated descriptors. The size of the patch is tested in Section 8.3.2.

The filter set is built in four main orientations (0, 45, 90 and 135 degrees) or eight orientations (0, 22.5, 45, 67.5, 90, 112.5, 135, 157.5). And only for one scale ($\sigma = 1$) which is matched with the one used in the filtering of corner detection. With four orientations we cover the main direction components of an image, horizontal, vertical and diagonals. With eight orientations we have double resolution, we will evaluate in Section 8.3.2 if it is worth-doing.

To obtain rotation invariance, DFT is applied as it is explained in Section 3.4. Best results, as seen in Section 8.3.2, are obtained using both second and third order derivatives descriptors.

7.1.3 Gabor Descriptors

The implementation of Gabor descriptors is done following the same idea of Gaussian derivatives descriptors.

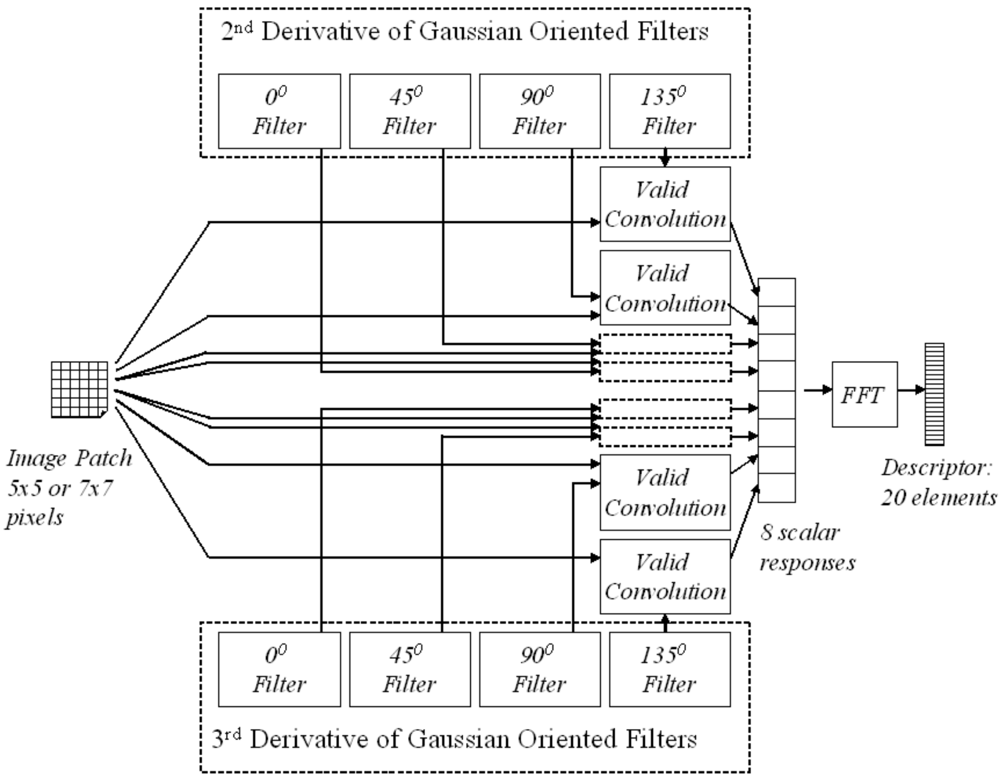

2nd Derivative of Gaussian Oriented Filters
0° Filter
45° Filter
90° Filter
135° Filter
Valid Convolution
Image Patch 5x5 or 7x7 pixels
8 scalar responses
FFT
Descriptor: 20 elements
3rd Derivative of Gaussian Oriented Filters

Figure 7.1: *Implementation of 4 orientations with 2nd and 3rd Gaussian derivatives descriptor*

Now, we only have two filters but we do the root of the sum of the squared responses (because of real and imaginary parts) [18].

$$q(x, y, f, \theta) = \sqrt{q_e^2 + q_o^2} \tag{7.1}$$

Recalling from Section 3.3:

$$q_e(x, y, f, \theta) = [g(x, y)cos(2\pi f(xcos\theta + ycos\theta)] * p(x, y) \tag{7.2}$$
$$q_o(x, y, f, \theta) = [g(x, y)sin(2\pi f(xcos\theta + ycos\theta)] * p(x, y) \tag{7.3}$$

Where $g(x, y)$ is the Gaussian function, f the spatial frequency, and θ the orientation.

We use four orientations (0, 45, 90, 135 degrees) [20]. To obtain different shapes of the Gabor function (like we did with Gaussian derivatives using different orders) this time we set four scales (and 4 different spatial frequencies). We do a multiresolution analysis.

These four scales have to be chosen. First of all, we choose four different spatial frequencies. In [34] to guarantee that the bandpass of the filter of highest frequency falls inside the image patch, the highest frequency has to be: $\frac{N_c}{4}\sqrt{2}$ being N_c the width of the patch in pixels and power of 2.

For 5x5 pixel patches, we choose $f = \sqrt{2}/4, \sqrt{2}/2, 3\sqrt{2}/4, \sqrt{2}$. We have also to change the standard deviation of the Gaussian according to these frequencies. For simplicity reasons $\sigma = \frac{1}{f}$.

7.2 Track continuation

In track management, we may be interested in continuating a track when no measurement is found just in case it is due to an occlusion or a temporary absence. If it is so, we prevent a track from dying and being born in a few frames time by only using the prediction when measurements are not present.

As seen in Section 4.2 the most adequate strategy to continuate a track when measurement absences occur is to use a track quality indicator. But it is not the only parameter that will be taken into account in our system for continuation.

Using track absence, a counter of the absences occurred in a track, we can control more accurately the maximum number of frames that a track should be continued. Track lifetime is also useful as we should not continue nearly newborn tracks as we do not have enough information to predict their motion for several frames.

We will combine these three parameters in our system in the following way:

We use the prediction when no measurement is found and these three conditions happen:

- Track quality indicator is lower than a quality threshold.
- Track absence counter is lower than the maximum absences permitted.
- Track lifetime is higher than a lifetime threshold. At least, the track should

not be a newborn.

Abdeljaoued used the gating threshold to penalise the track quality indicator when continuation occurs. Recalling from Section 4.2:

$$\lambda(k) = \lambda(k-1) + \gamma \tag{7.4}$$

We will do it in an indirect way and meanwhile we will solve another problem: when we continuate, we ”find” a very close ”measurement” to the prediction. In fact we obtain an estimation error ($\tilde{\mathbf{z}}(k+1|k) = \mathbf{z}(k+1) - \hat{\mathbf{z}}(k+1|k)$) equal to zero as the prediction and the ”measurement” are the same.

$$\mathbf{S}(k+1) = E\{\tilde{\mathbf{z}}(k+1|k)\tilde{\mathbf{z}}^T(k+1|k)|Z^k\} \tag{7.5}$$

This fact will lead to a lower predicted measurement covariance which will reduce the gating region:

$$\mathbf{V}(k+1,\gamma) = \{\mathbf{z}(k+1) : \nu^{\mathbf{T}}(k+1)\mathbf{S}^{-\mathbf{1}}(k+1)\nu(k+1) < \gamma\} \tag{7.6}$$

which will, at the same time, make the matching more difficult.

One solution to that problem is to expand, or at least maintain, the gate size by adding measurement noise. Recalling from Section 4.1.1, predicted measurement covariance is computed as follows:

$$\mathbf{S}(k+1) = \mathbf{H}(k+1)\mathbf{P}(k+1|k)\mathbf{H}^T(k+1) + \mathbf{R}(k+1) \tag{7.7}$$

where $R(k+1)$ is the covariance matrix of the Gaussian noise we assume in the Kalman measurement model. By increasing the diagonal values of that matrix, we compensate the reduction of predicted measurement covariance. If this compensation results in a predicted measurement covariance growth, we increase our track quality indicator:

$$\lambda(k) = \lambda(k-1) + \nu^{\mathbf{T}}(k+1)\mathbf{S}^{-\mathbf{1}}(k+1)\nu(k+1) \tag{7.8}$$

and it is not necessary to add the penalisation.

This variant is tested, as well as the other parameters tuned, in Section 8.4.

7.3 Tracking using Pose Estimation feedback

We can benefit from Pose Estimation data, translation and rotation, as it has been showed in Section 4.3.2. In our system, though, we use IMM filter for tracking but we can improve its performance by means of predicting more precisely where will future features be located for each track.

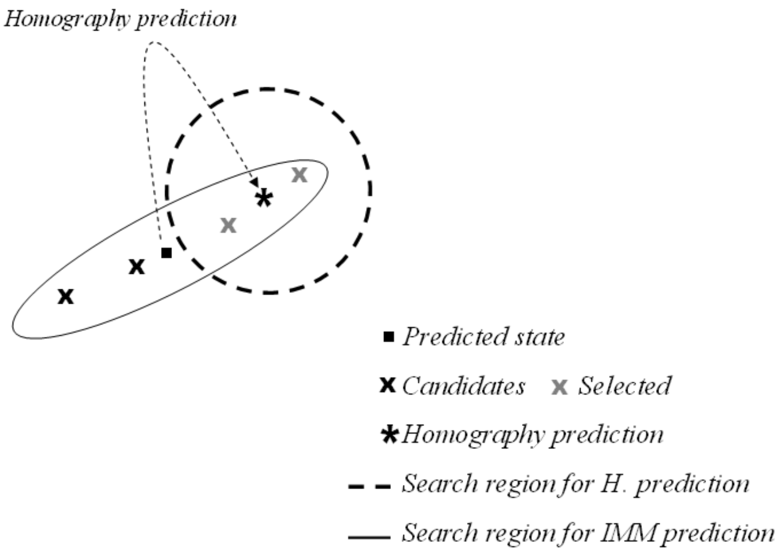


Figure 7.2: *Gatings intersection using Homography prediction*

An hybrid tracking can be performed by intersecting the search region from IMM filter Gating and a new one defined in a circular neighbourhood of the prediction based on homographies. Following the idea of Section 4.3.2, we predict the position of a feature point in a track multiplying last homography between frames with the last tracked point. The circular search region will be defined using squared Euclidean distance. This threshold is computed in Section 8.5.1 with Mean Square Error (MSE).

Part III

Experiments and results

Chapter 8

Feature Point Extraction and Tracking Evaluation

This chapter contains the experiments and results for the overall project. First of all, some experiments have been done for tuning, obtaining the best parameters to use. Then, some performance tests are used for evaluation purposes.

8.1 Test Framework

8.1.1 Equipment

Experiments have been performed on an Intel® Pentium® M 4 at 2GHz, 768MB R.A.M.. The camera used is a Logitech® QuickCam® for Notebooks Pro. The resolution used has been 320*240 pixels.

System and experiments are implemented in C/C++ using Microsoft® Visual C++ .NET with Intel® Integrated Performance Primitives (IPP), Intel® OpenCV and OpenGL libraries.

8.1.2 Synthetic Sequence

To test feature point extraction and characterisation, a synthetic sequence has been created and used to simplify subjective measures such as repeatability(Section 2.2.1). The sequence consists in a number five rotated 15 degrees each frame, this helps to determine rotation invariance in the extraction and characterisation step. In Figure 8.1 we can see the complete sequence in snaphots. The sequence has ben called FIVE.

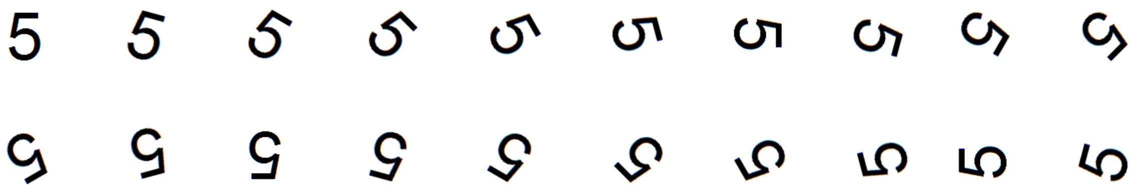


Figure 8.1: *FIVE complete sequence snapshots*

8.1.3 Real Indoor Sequence

To test feature point extraction and tracking in real conditions, a sequence has been recorded. Since the camera is a simple webcam, the quality of the recording is poor, noisy images are recorded. The sequence lasts 70 frames and consists in a camera scrolling in the ITS Laboratory here at EPFL. This sequence has been labelled by hand, which means that feature points have been noted down and real tracks have been stated. By doing so, we have obtained a ground truth for our experiments. In Figure 8.2 we can see the first 10 frames in snapshots. The sequence has ben called LABO.

8.2 Feature Point Extraction Evaluation

In this section, in order to evaluate and compare Harris and Lindeberg detectors, some tests have been done. First and second tests, that is repeatability and reliability, are useful to evaluate the detectors performance.

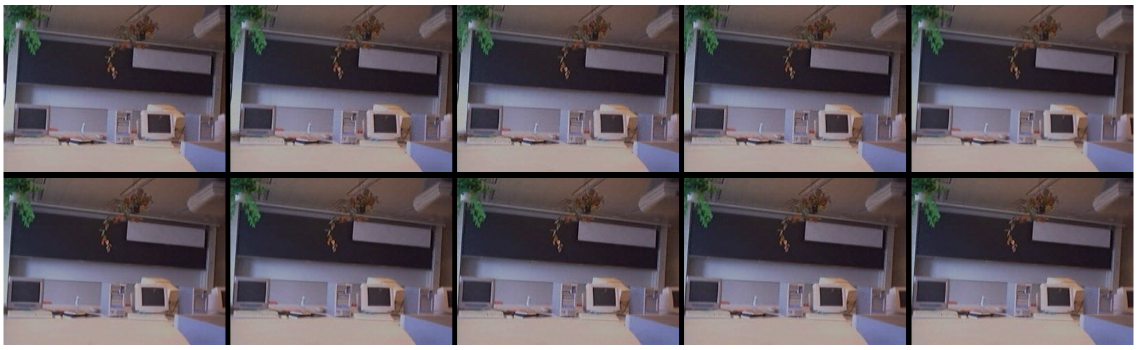


Figure 8.2: *First 10 frames of the LABO sequence*

However, in our application, it is also very important to measure computational cost of these algorithms. To do so, measures of computational time have been effectuated for both detectors.

8.2.1 Repeatability

Recalling from Section 2.2.1, ε-repeatability is:

$$r(\varepsilon) = \frac{|D(\varepsilon)|}{N}, 0 \leq r(\varepsilon) \leq 1 \tag{8.1}$$

We do some feature extraction experiments and compute 3-repeatability to evaluate the performance of Feature Point Extractors.

Kind of Features

To begin with, we compare Harris corner detector with Harris corner and edge detector and with Lindeberg detector. Thus, we can evaluate which kind of image features are more suitable for our goals. The experiment has been performed with the following conditions:

- Criteria: 3-repeatability.
- Sequence used: FIVE, 20 frames.

- We set the thresholds in order to extract around 12 points by Harris corner & edge and Lindeberg detectors.
- Around 8 points extracted for Harris corner detector, as it is impossible to detect more than 9 corners in this sequence (see Figure 8.4a).

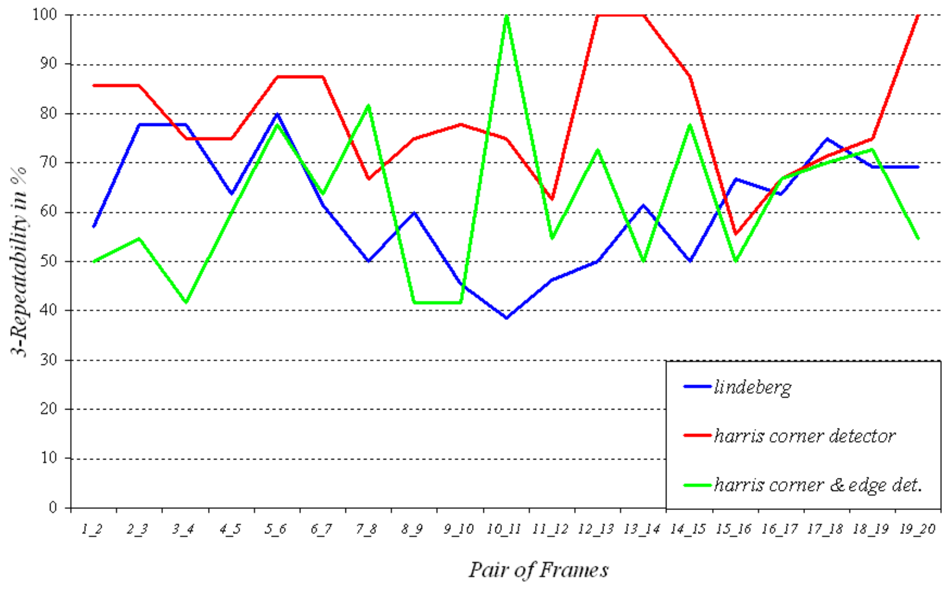
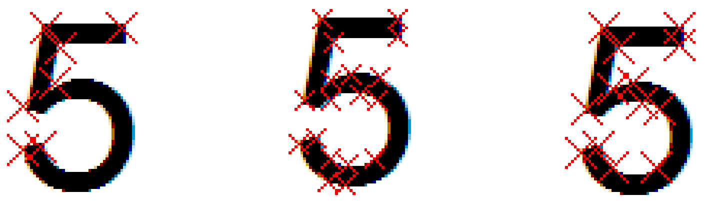


Figure 8.3: *3-repeatability rate with respect to 15-degree rotation during 20 frames (FIVE sequence) for Harris and Lindeberg algorithms.*

In Figure 8.3 we observe that best performance is achieved with Harris corner detector. If we use Harris detector to extract corners and edges, the performance is very irregular as extracted edges are very unstable. The position of edge feature points changes easily as they are not characterised by variations in two directions. Lindeberg detector extracts some edge points too (see Figure 8.4c). The mean over these 20 frames is for Harris corner detector nearly 80%. For the other two is around 60%.

Real Indoor Scene

We repeat the experiment with a real noisy sequence. This time, only for Harris and Lindeberg corner detectors. The experiment has been performed with the following conditions:



(a) *Harris corner detector* (b) *Harris corner & edge* (c) *Lindeberg detector*

Figure 8.4: *Features detected in frame one of FIVE sequence*

- Criteria: 3-repeatability.
- Sequence used: LABO, first 20 frames.
- The thresholds are set in order to extract around 50 points for both detectors.

In Figure 8.5 we observe that again best performance is achieved with Harris corner detector. Computing the mean over 20 frames, the repeatability rates are 89% and 77% for Harris and Lindeberg, respectively.

8.2.2 Reliability

As seen on Section 2.2.2, reliability gives us the idea if the same feature points are being extracted for several consecutive frames. Thus, they are suitable for tracking. We choose a number of frames to compute it. Reliability is the number number of consecutive frames a point is tracked out of the number of frames chosen for computation. We have performed the experiment with the following conditions:

- Criteria: Reliability for 10 frames.
- Sequence: FIVE, 10 frames.
- Extracted features in first frame (7 points) taken into account for Harris detector.
- 8 points taken into account for Lindeberg detector.

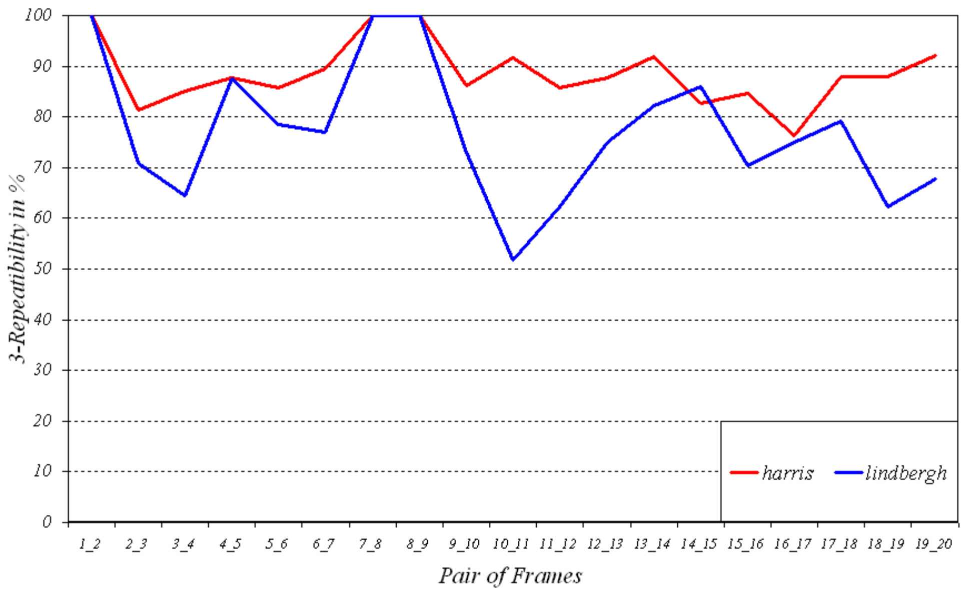


Figure 8.5: *3-repeatability rate for real indoor sequence (LABO sequence) for Harris and Lindeberg algorithms.*

Detector	Reliability (10 frames)
Harris	0.77
Lindeberg	0.67

Table 8.1: *Reliability computed for 10 frames. (FIVE sequence) for Harris and Lindeberg detectors.*

In Table 8.1, average reliability is shown for synthetic sequence FIVE. Harris corner detector is more reliable with 0.77 of reliability computed for 10 frames. Which means that in average the features detected in the first frame are detected in the 7 following frames in a ten frame sequence.

8.2.3 Complexity

For being able to compare computational cost (complexity) for the implementations of both extraction algorithms, we have measured the performance time for several realisations. This measure of time depends heavily on the number of features detected. This experiment has been performed under the following conditions:

- Measure: Computational time for detecting all features in a frame.
- Sequence used: LABO, first 30 frames.
- Around 12 features have been extracted for both detectors.
- 10 realisations.

In Figure 8.6 we can see computational time of both algorithms for an indoor sequence of 30 frames. Time required for extracting the features for a frame is being measured as the mean over 30 frames and 10 realisations. The performance for Harris corner detector is very fast in two realisations, probably due to instantaneous few CPU usage. Lindeberg algorithms is faster in mean, but there is not much difference between them. Harris algorithm is in mean less than 15% slower.

8.2.4 Comparison and Final Decision

Results confirm better performance of Harris detector in terms of Repeatability but also Reliability. Results are summarised in table 8.2.

However we have to pay a price for this improvement, computational cost is slightly higher in Harris detector. Despite this fact, Harris clearly has better performance so it would be interesting to change into Harris implementation.

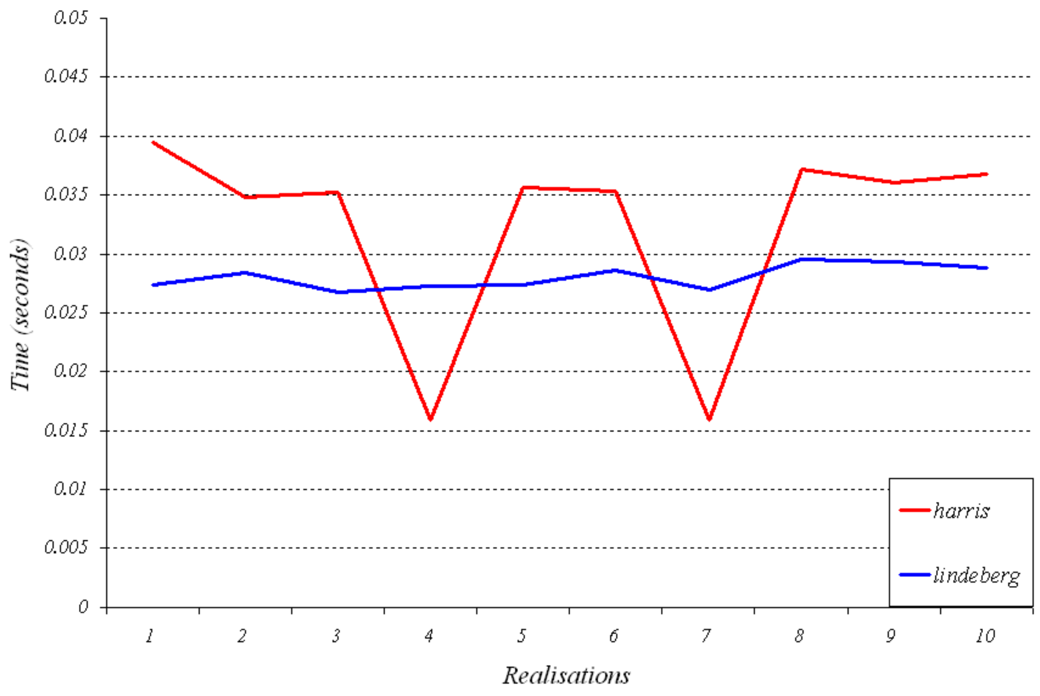


Figure 8.6: *Computational time (mean over 30 frames, 10 realisations) for Harris and Lindeberg detectors. LABO sequence.)*

Criteria	Harris	Lindeberg	Harris Improvement
Repeatability	89%	77%	15.6%
Reliability	0.77	0.67	14.9%
Complexity	32 ms	28 ms	-14.2%

Table 8.2: *Harris and Lindeberg comparison summary*

8.3 Photometric Test Techniques Comparison

In this section, experiments have been done to ensure that the best configuration of each method is used. Therefore, first of all the three methods have been tested separately, and then, some experiments are added to compare their performance in the same conditions.

8.3.1 Correlation Coefficient

We can only test NCC performance in relation to the size of the neighbourhood it takes to compute the similarity between two Feature Points.

	3x3 patch	5x5 patch	7x7 patch	9x9 patch	15x15 patch
Correct matches	0.51%	0.82%	0.84%	0.71%	0.45%

Table 8.3: *NCC performance, 3 different neighborhood sizes (LABO sequence only frames 1 and 10 with AWGN $\sigma = 10$), mean over 10 realisations*

In Table 8.3, we can see the results of the following experiment: LABO sequence is taken. Feature points of first frame and from frame number 10 are extracted. We consider that we have a correct match when the maximal NCC is achieved when comparing one feature point of first frame with its correspondence in frame 10. A considerable amount of synthetic Gaussian noise is added to prove its robustness under noisy conditions.

As we can see on the results, 5x5 size and 7x7 size are best choices as they provide more correct matches. Confirming the decision taken in Section 7.1.1.

8.3.2 Gaussian Derivatives Descriptors

In Gaussian derivatives descriptors we can change their performance via the size of the patch, the derivatives used and the metrics chosen for comparison.

In order to select which derivatives are more adequate for matching. We have chosen randomly, for the first 10 frames of LABO sequence, one feature point and we have compared it with all the others in the next frame. When the distance between two descriptors is minimal and they are the same point, we consider a correct match.

We can test that by considering different sizes of the patch and different number of orientations.

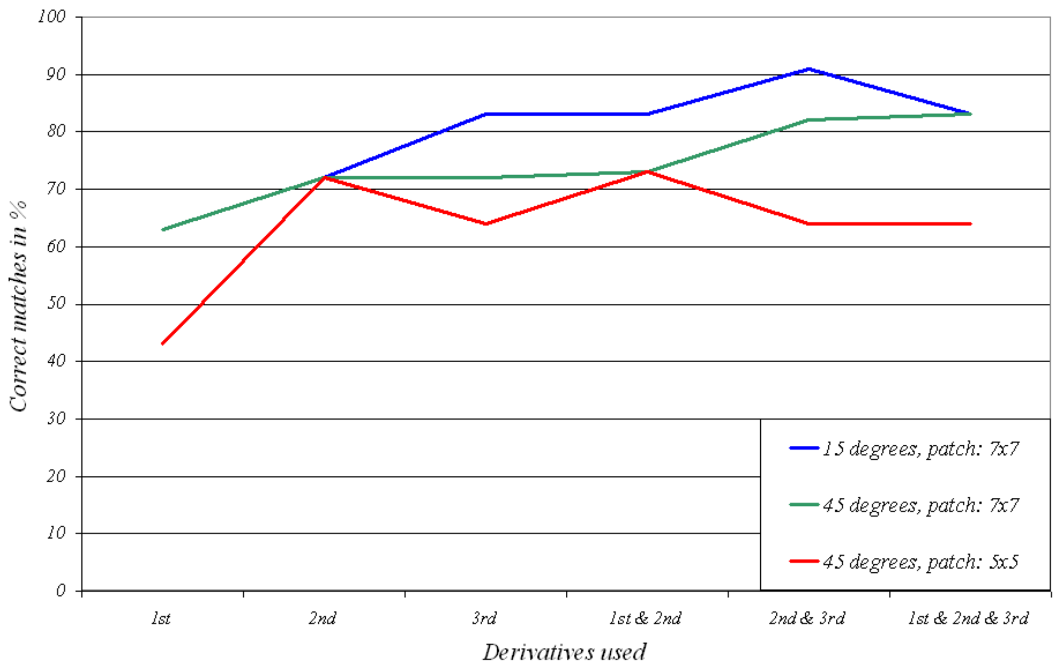


Figure 8.7: *Correct matches using Gaussian Descriptors based on different combinations of derivative orders (LABO sequence first 10 frames)*

In Figure 8.7 we can see that 2 out of 3 implementations obtain better results using the second and third derivatives together (and first one, but it doesn't affect much). Best performance is achieved with patches sized 7x7 with 8 orientations, that is 15 degree rotation each.

We characterise the neighbourhood of a feature point with a descriptor, a vector formed by the responses of different orientated Gaussian derivatives filters. In order to decide which metrics we could use to compare the descriptors, we repeat the experiment performed in Section 8.3.1 for different patch sizes and number of orientations comparing using two classification (distance) metrics, Euclidean distance and NCC. Again frames one and ten of LABO sequence with Gaussian noise added are used to compute the correct matches.

This experiment can be considered hostile (frames are very separated in time and a considerable amount of noise is being added). But it can help us to decide which size of the patch is the optimal with noisy environments, and if there is significant difference when using 4 or 8 orientations in the presence of noise.

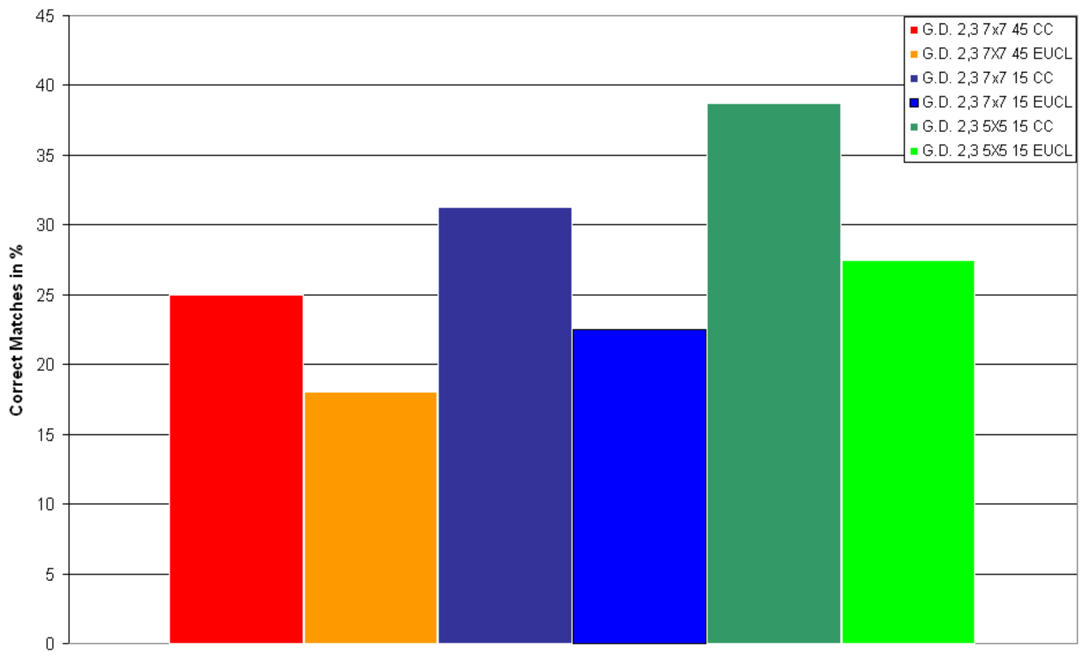


Figure 8.8: *Comparison between Correlation Coefficient and Euclidian distance for matching Gaussian derivatives descriptors (LABO sequence only frames 1 and 10 with AWGN $\sigma = 10$), 10 realisations*

In Figure 8.8 we can see the results. Normalised Cross Correlation Coefficient clearly gives better performance. On the other hand, smaller patches perform better as well as using more orientations.

8.3.3 Gabor Descriptors

For Gabor descriptors we have to perform the same kind of experiments as we did for Gaussian derivatives descriptors. In this case we can change their performance via the size of the patch, the scales used and the metrics chosen for comparison.

In order to ensure that we have taken the correct decision about the scales used in the design of the descriptor in Section 7.1.3, we perform the same test as for Gaussian derivatives. We compute the correct matches for the first 10 frames of LABO sequence for different scale sets. We also consider different sizes of the patch.

In Figure 8.9 we can see that we obtain better results with the scale sets whose

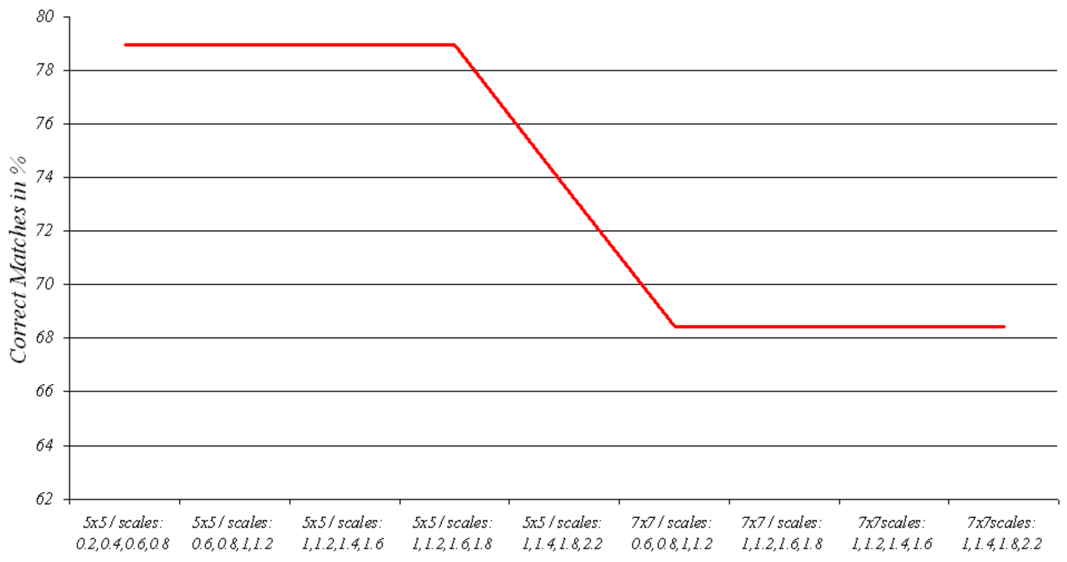


Figure 8.9: *Correct matches using Gabor descriptors for different scale sets and patch sizes (LABO sequence first 10 frames)*

maximum around 1.8. Therefore, the choice of $f = \sqrt{2}/4, \sqrt{2}/2, 3\sqrt{2}/4, \sqrt{2}$ for the scales is adequate. Best performance is achieved with patches sized 5x5.

Like in Gaussian descriptors , we characterise the neighbourhood of a feature point with a vector formed by the responses of different orientated Gabor filters. We have to decide if it is better to use NCC or Euclidean distance for comparing. We repeat the hostile experiment of comparing correct matches between frames one and ten of LABO sequence with Gaussian additive noise with both metrics.

	NCC	Euclidean Distance
Correct matches	27.5%	17.5%

Table 8.4: *Comparison between Correlation Coefficient and Euclidian distance for matching Gabor descriptors (LABO sequence only frames 1 and 10 with AWGN $\sigma = 10$), mean over 10 realisations*

In Table 8.4 we can see the results. Normalised Cross Correlation Coefficient gives better performance. But in this case both results are disappointing. Gabor descriptor seems to have a lack of robustness under noisy conditions.

8.3.4 Comparison and Final Decision

Once we have tested photometric test techniques separately, it is time to compare them in order to find which is more adequate in our system. The criteria used in this comparison are computational time and performance.

Computational Time

We cannot lose sight of we are working in an augmented reality system, where computational cost is critical. Moreover, we would reject any technique involving an unacceptable computational cost in our system. This experiment has been performed under the following conditions:

- Measure: Computational time for performing a photometric test.
- Sequence used: LABO.
- 150 photometric tests performed for each technique.
- 5 realisations.

In Figure 8.10 the results are shown. We can observe that NCC is significantly faster than the other two methods. Gaussian Descriptors computation spends nearly half of the required time for Gabor Descriptors.

Performance

In this section a summary of the performance achieved by these three techniques is exposed. Using a sequence with noise absence, results for all approaches to photometric testing are successful. Correct match rates over or near 80% in all cases, as seen in Table 8.5.

	NCC	Gaussian Derivatives	Gabor Filters
Correct matches	0.84%	83%	78.9%

Table 8.5: *Comparison of photometric test techniques performance (LABO sequence without noise)*

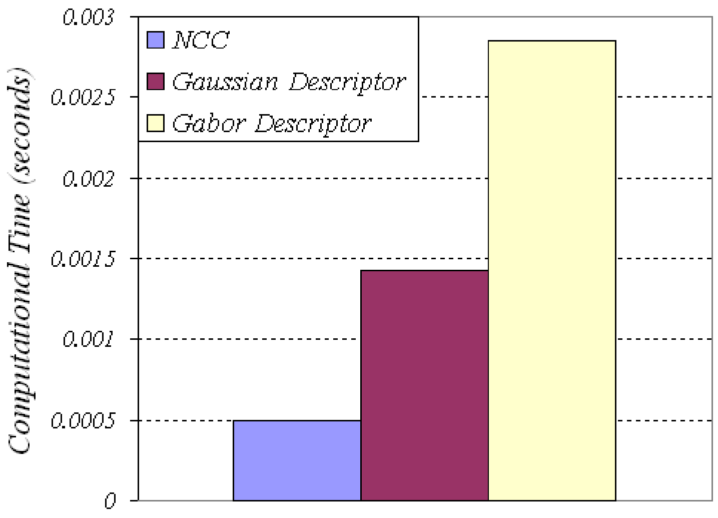


Figure 8.10: *Computational time for NCC, Gabor and Gaussian descriptors (LABO sequence, mean over 150 tests and 5 realisations).*

Nevertheless, when we add noise to prove robustness Gaussian and Gabor descriptors performance degrades dramatically. Meanwhile NCC performance remains nearly the same. We can see these results in Table 8.6.

	NCC	Gaussian Derivatives	Gabor Filters
Correct matches	0.82%	38.75%	27.5%

Table 8.6: *Comparison of photometric test techniques performance (LABO sequence only frames 1 and 10 with AWGN $\sigma = 10$), mean over 10 realisations*

Final Decision

One possible reason for this poor performance of Gaussian and Gabor descriptors could come from the distance metrics used. All current applications [26] [18] [20] [23] [35] [19] [36] of such descriptors are texture/objects recognition and/or classification. Therefore the system has to classify one image in one of the categories from its library. Thus, more complex distances can be computed such as Mahalanobis or Weighted Euclidian. In one frame per frame basis these statistic measures are not possible.

One of the drawbacks of NCC is its rotational variance, which is one of Gaussian and Gabor descriptors strengths. But in a frame per frame basis, this is not critical and obviously, there is not any better characterisation of an image than the image itself.

Therefore, having seen the fact that NCC is the fastest and most robust method in the way we will use it, we will maintain current implementation. Our Gabor and Gaussian descriptors are not a good approach to photometric test in our AR time conditions.

8.4 Track Continuation Evaluation

Track continuation evaluation is going to be tested using the labelled sequence LABO. Thus, the obtained results can be compared with the ground truth. Criteria used for evaluation are the number of births and the number of deaths. We define a birth as the event that happens when a track is started. A death is the same but for a discontinuation of a track. The average lifetime of a track or the mean number of tracks are also possible criteria, but the absolute number of births and deaths provides more information about the continuation of tracks.

In the labelled sequence LABO, 10 randomly absences lasting one or two consecutive frames have been provoked. This way, we simulate occlusions and failures of the extractor. As we have a ground truth of the sequence's tracks we can exactly measure if continuations succeed or not.

Track continuation performance using track absence indicator is shown in Figure 8.11. The behaviour is as expected. When absences last one frame, continuation decreases the number of births and deaths, consequently the number of tracks is also reduced. For two-frame absences, continuation is useful for over two-frame continuation. Obviously, if the absence lasts two frames and we only do continuation for one frame, what we do, actually, is creating new frames that will not be followed by measurements. Therefore the number of births and deaths increases. For over five-frame continuations, although performance is not always degradated, misassociations may occur.

The performance of the overall technique is shown in Figure 8.12. Births and deaths for sequence LABO with ten random 2-frames absences are shown for track absence based continuation, track quality indicator based continuation and the combination of these two. Births and deaths decrease significantly using the proposed combina-

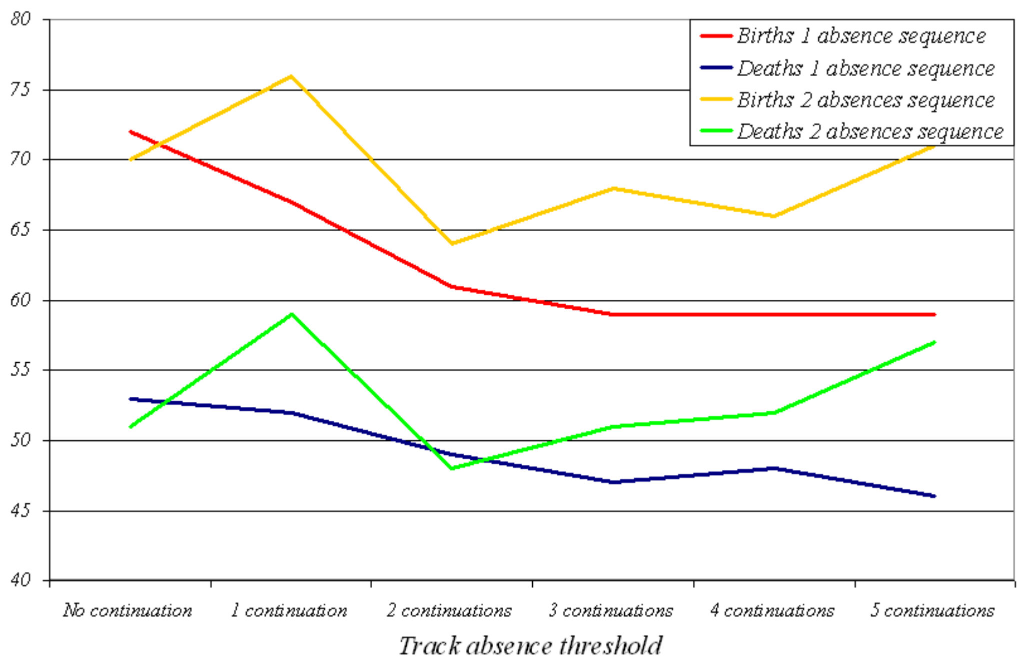


Figure 8.11: *Track absence performance*

tion.

Strategy	Births	Ground Truth	Improvement respect to TA	Improvement respect to TQI
Track Absence (TA)	71.20	37	-	1.72%
Track Quality Indicator (TQI)	71.80		1.75%	-
Combination (TA & TQI)	68.60		7.61%	9.19%

Table 8.7: *Track continuation techniques comparison, (LABO sequence, 10 realisations)*

In order to prove track continuation suitability when direction changes occur, we have run the same experiment with LABO sequence played forward and backward. Then, the sequence has twenty random 2-frame absences and double length. Results are shown in Table 8.8. Track continuation reduces the number of births. However, the number of births increases significantly from the expected with and without track continuation. This is caused as the assumed motion model persists for various frames after the change in direction. As a consequence, many extracted feature points are not associated with existing tracks during this period, provoking the increase in the

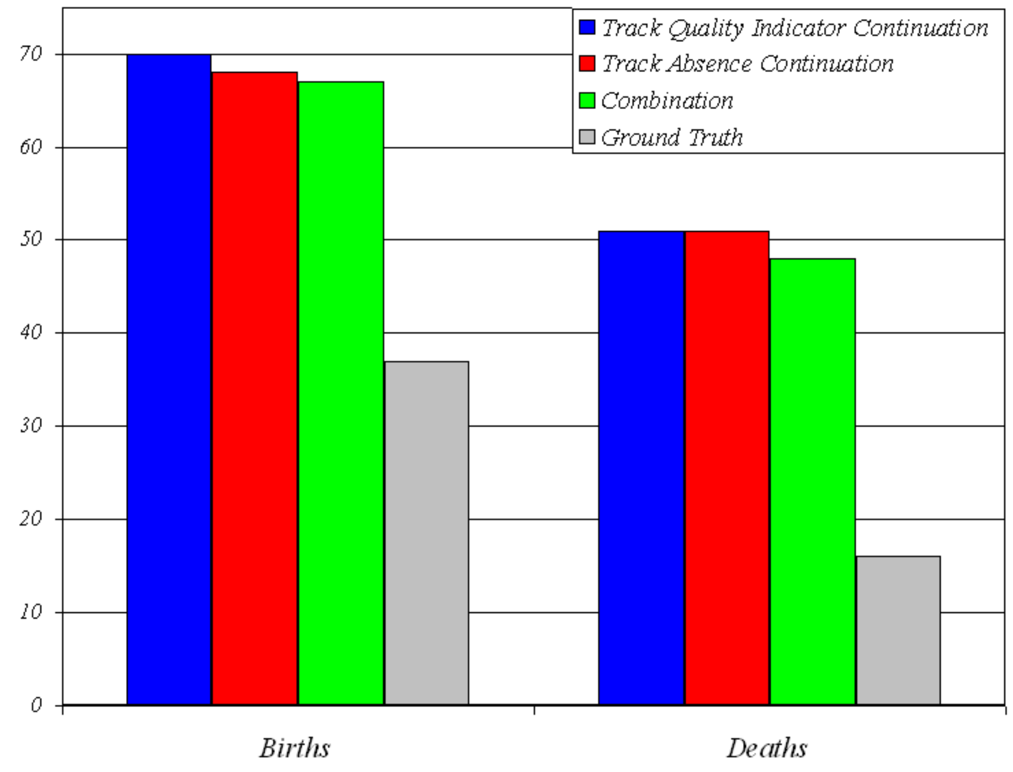


Figure 8.12: *Track continuation techniques comparison*

total number of births.

Strategy	Births	Ground Truth	Track Continuation Improvement
No track continuation	256	45	-
Combination (TA & TQI)	241		7.2%

Table 8.8: *Track continuation influence when direction changes occur, (LABO sequence forward and backward)*

8.5 Pose Estimation Applied to Tracking Evaluation

As seen in Section 4.3.2, we will use pose estimation data to predict the future position of each feature.

8.5.1 Tuning

First of all, the system has to be tuned. We have to set a threshold, a limit for the circular search area around the homography prediction. Using the whole LABO sequence, we have extracted the relative homographies between frames. From these homographies we obtain the predictions. We can compute the Mean Square Error (MSE) which is an indication of how accurate is the prediction as well as can be used to limit the search area. We compute the average MSE of each prediction with the extracted corresponding point over all the points and frames of the sequence. We obtain an MSE equal to 170, so the threshold has to be around this value in order not to discard many corresponding features.

8.5.2 Evaluation

We cannot use the labelled features in LABO sequence because they are few (around 20), the homography prediction (HP) is poor. We evaluate the performance using the Harris extracted points in LABO sequence (around 30 features per frame). Criteria we use are:

- Ratio of pose estimation update failures: The number update failures (there are not enough points for pose estimation) out of the number of frames of the sequence. The lower it is, the better the overall pose estimation becomes.
- Mean ratio of inliers: It shows us how many features out of the total, follow the same motion pattern according to the pose estimation. The higher this value is, the more reliable predictions are.
- Increase of the number of births.

Last item in the list above is taken into account to evaluate the possible worsening of this technique. As we shrink the search area, we may loose candidates which would be assigned. It is a trade off between the the tracks we loose and the global behaviour.

Tables 8.9 and 8.10 show the results of the tests using these criteria. The experiments have been performed using track continuation and without using it. In order to see the effect of track continuation (TC) itself it has been tested without homography prediction, too.

Without track continuation

Criteria	Ratio of pose Update Failures	Mean Ratio of Inliers	Increase of Number of Births
No HP	0.24	0.73	-
HP	0.16	0.75	7.6%

Table 8.9: *Effect of homography prediction*

From Table 8.9 we observe that the number of update failures decreases significantly (35%) as well as the mean ratio of inliers increases. As we had predicted, the shrink of the search region motivates new births, as we prevent matches that previously were done.

With track continuation

Criteria	Number of Update Failures	Mean Ratio of Inliers	Increase of Number of Births
No HP but TC	0.21	0.7	13.4%
HP with TC	0.11	0.81	-3.5%

Table 8.10: *Effect of homograhpy prediction and track continuation*

From Table 8.10 we observe that track continuation effect itself is nearly negligible. There is a small decrease of update failures, as well as the mean ratio of inliers. Nevertheless there is a significant increase of the number of births. Continuation is intended to provide predictions when no measurements are found as a consequence of occlusions or extraction failures. However, sometimes predictions are used instead of already extracted features which do not fit in the gating. This fact, leads to a larger number of births. However, this fact can be also an advantage, tracks that were going to die are updated and may continue in the future. This explains why there is an improvement in the ratio of update failures.

Most valuable results, however, are obtained combining homography prediction and track continuation. In Table 8.10 it is shown the significant decrease (53%) in the number of update failures. Moreover, the mean ratio of inliers is over 80% and even the number of births decreases.

Augmented Sequence

In the last two experiments, we have included an augmented object to the scene in order to see if its registration is properly done. The augmented object is a maroon square placed in the screen of a monitor in scene.

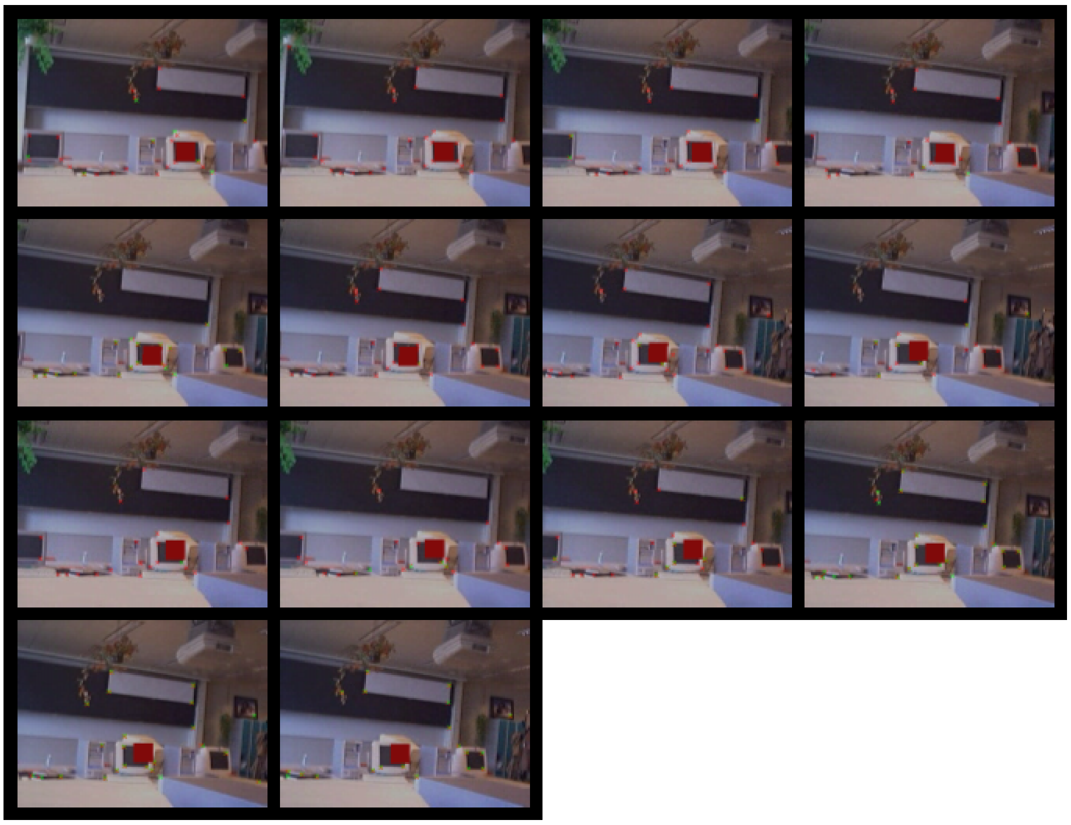


Figure 8.13: *Augmented scene without track continuation nor homography prediction*

Without track continuation neither homography prediction, the augmented object is not placed properly, especially at the end of the sequence (as the error accumulates with time).

Using track continuation together with prediction, there is a significant improvement in the registration of the augmented object, as seen in Figure 8.14.

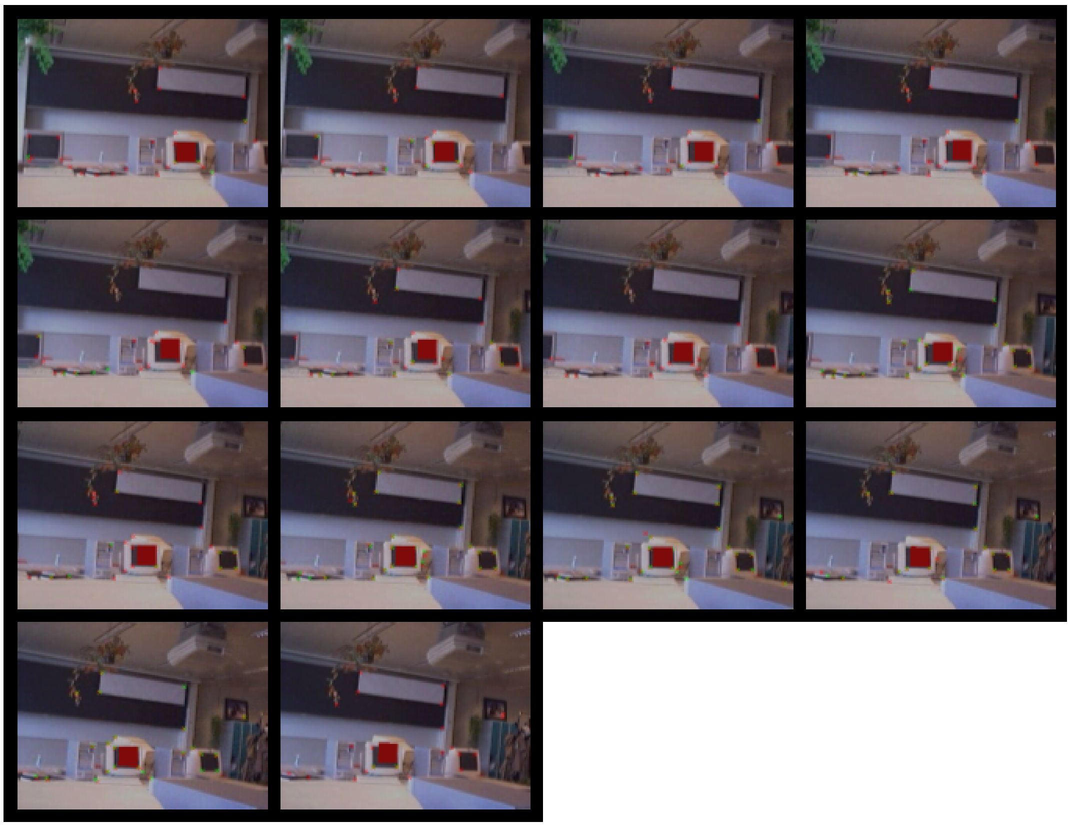


Figure 8.14: *Augmented scene with track continuation and homography prediction*

8.5.3 Conclusions

The use of pose estimation data applied to tracking has been here evaluated. The homography prediction itself is useful. However best results are obtained when combining with track continuation.

Part IV

Conclusions and future work

Chapter 9

Conclusions and Future Work

With this chapter we conclude this document. It contains the summary of the achievements and proposes extensions and improvements.

9.1 Conclusions

In this project, we have designed and implemented some new techniques in order to improve already implemented feature point extraction and tracking techniques in the core of an augmented reality system.

The improvement in feature point extraction has been successful. Harris corner detector has better performance than the former implementation based on Lindeberg detector. We pay a small increase of computational cost, but it is not significant.

Present photometric test implementation is based in cross-correlation coefficient. Two more approaches have been implemented and compared, based in Gaussian derivatives and Gabor filters. The results are deceiving in part, as new approaches are not suitable for feature point characterisation in the way of our needs. Nevertheless, we have proved a good performance of NCC under noisy conditions. Moreover, NCC is really fast and simple.

For preventing premature track deaths due to occlusions or failures of the extracting algorithm, a track continuation algorithm has been designed and implemented. A track quality indicator measuring if a track is worth-continuing or not has been found. The results prove that our continuation algorithm is capable of solving brief occlusions successfully.

The assignment technique in the tracking step has been enhanced taking advantage from pose estimation information. The new technique working together with track continuation has proven a significant improvement in pose estimation, our final goal.

9.2 Future Work

The number of extracted feature points is controlled by a threshold. This threshold has to be set according to the characteristics of the sequence (illumination, texture...). It would be interesting to estimate the right threshold to use for a desired number of extracted points in the very first frames of a sequence.

The alternatives to NCC for photometric test have been designed based on texture descriptors used in classification applications mainly. One possible improvement would be an efficient implementation of the ”Local Jet” proposed by Schmid and Mohr in [5], which consists in a more complex local invariant based in Gaussian Derivatives with good results in image retrieval.

The developed track continuation algorithm is based on using the prediction of IMM filter when no measurement is found. As proven in Section 8.5 the prediction using homographies achieves good results. It would be interesting to use it also for track continuation.

The new gating in tracking taking advantage of pose estimation uses the homography relating two frames. This is far more unstable than the absolute homograhpy, relating current frame with the initial one. We cannot relate each point in each frame with the points in the beginning of the sequence as some points may have appeared afterwards. But we could store initial homography as a track parameter for each tracked point and use absolute homography for prediction.

Bruno Palacios Serra
Lausanne, 15th September 2005

Bibliography

[1] S. J. D. Prince, K. Xu, and A. D. Cheok, "Augmented reality camera tracking with homographies," *IEEE Comput. Graph. Appl.*, vol. 22, no. 6, pp. 39–45, 2002.

[2] M. Aron, G. Simon, and M.-O. Berger, "Handling uncertain sensor data in vision-based camera tracking," in *International Symposium on Mixed and Augmented Reality - ISMAR'04, Arlington, USA*, pp. 58–67, IEEE and ACM, Nov 2004.

[3] Y. Abdeljaoued, D. M. Sanjuan, and T. Ebrahimi, *3D Videocommunication : Algorithms, concepts and real-time systems in human centred communication*, ch. Tracking and User Interface for Mixed Reality, pp. 315–332. John Wiley and Sons Ltd, July 2005.

[4] R. Azuma, Y. Baillot, R. Behringer, S. Feiner, S. Julier, and B. MacIntyre, "Recent advances in augmented reality," *Computer Graphics and Applications*, vol. 21, pp. 34–47, Nov. 2001.

[5] C. Schmidt and R. Mohr, "Local grayvalue invariants for image retrieval," *IEEE transactions on pattern analysis and machine intelligence*, vol. 19, May 1997.

[6] Y. Abdeljaoued, *Feature point extraction and tracking for video summarization and manipulation.* PhD thesis, EPFL, 2001.

[7] M. Trajkovic and M. Hedley, "Fast corner detection.," *Image Vision Comput.*, vol. 16, no. 2, pp. 75–87, 1998.

[8] B. Zitova, J. Flusser, J. Kautsky, and G. Peters, "Feature Point Detection in Multiframe Images," in *Proceedings of the Czech Pattern Recognition Workshop 2000* (T. Svoboda, ed.), pp. 117–122, 2–4, 2000.

[9] C. Schmid, R. Mohrand, and C. Bauckhage, "Comparing and evaluating interest points," in *ICCV '98: Proceedings of the Sixth International Conference on*

Computer Vision, (Washington, DC, USA), p. 230, IEEE Computer Society, 1998.

[10] C. Harris and M. Stephens, "A combined corner and edge detector.," in *Proc. 4th Alvey Vision Conf.*, pp. 189–192, 1997.

[11] N. Sebe, Q. Tian, E. Loupias, M. S. Lew, and T. S. Huang, "Evaluation of salient point techniques," in *CIVR '02: Proceedings of the International Conference on Image and Video Retrieval*, (London, UK), pp. 367–377, Springer-Verlag, 2002.

[12] N. Sebe and M. S. Lew, "Comparing salient point detectors," *Pattern Recogn. Lett.*, vol. 24, no. 1-3, pp. 89–96, 2003.

[13] K. Mikolajczyk and C. Schmid, "Indexing based on scale invariant interest points," in *International Conference on Computer Vision July 2001*, pp. 525–531, 2001.

[14] N. Ichimura, "Feature point tracking based on feature point extraction by frame," *AIST:tech. report*, 2002.

[15] E. Vincent and R. Laganière, "Detecting and matching feature points," *Visual Communications and Image Representation*, 2004.

[16] S. Ravela, B. Draper, J. Lim, and R. Weiss, "Adaptive tracking and model registration across distinct aspects," in *IROS95*, pp. 174–180, 1995.

[17] K. Bhattacharjee, *A Computational Approach to Image Retrieval.* PhD thesis, EPFL, 1999.

[18] T. N. Tan, "Rotation invariant texture features and their use in automatic script identification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 7, pp. 751–756, 1998.

[19] G. M. Haley and B. S. Manjunath, "Rotation-invariant texture classification using modified gabor filters," in *ICIP*, pp. 262–265, 1995.

[20] T. N. Tan, "Texture feature extraction via cortical channel modelling.," *In Proceedings of 11th IAPR International Conference on Pattern Recognition*, pp. 607–610, 1992.

[21] A. C. Bovik, M. Clark, and W. S. Geisler, "Multichannel texture analysis using localised spatial filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 1, pp. 55–73, 1990.

[22] M. Tuceryan and A. K. Jain, "Texture analysis," pp. 235–276, 1993.

[23] S. R. Fountain and T. N. Tan, “Efficient rotation invariant texture features for content - based image retrieval„” *Pattern Recognition*, vol. 31, no. 11, pp. 1725–1732, 1998.

[24] W. T. Freeman and E. H. Adelson, “The design and use of steerable filters,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 13, no. 9, pp. 891–906, 1991.

[25] J. J. Yokono and T. Poggio, “Oriented filters for object recognition: an empirical study.,” in *FGR*, pp. 755–760, 2004.

[26] J. J. Yokono and T. Poggio, “Rotation invariant object recognition frome one training example.,” 2004.

[27] K. Mikolajczyk and C. Schmid, “A performance evaluation of local descriptors,” 2004.

[28] Y. Bar-Shalom and X.-R. Li, *Estimation and Tracking: Principles, Techniques and Software.* Artech House, Inc., 1993.

[29] G. Welch and G. Bishop, “An introduction to the kalman filter,” tech. rep., Chapel Hill, NC, USA, 1995.

[30] I. Cox and S. Hingorani, “An efficient implementation and evaluation of reid’s multiple hypothesis tracking algorithm for visual tracking,” in *ICPR94*, pp. A:437–442, 1994.

[31] C. J. Veenman, M. J. T. Reinders, and E. Backer, “Resolving motion correspondence for densely moving points,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 1, pp. 54–72, 2001.

[32] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision.* Cambridge University Press, ISBN: 0521540518, second ed., 2004.

[33] M. A. Fischler and R. C. Bolles, “Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography,” *Comm. of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[34] A. K. Jain and F. Farrokhnia, “Unsupervised texture segmentation using gabor filters,” *Pattern Recogn.*, vol. 24, no. 12, pp. 1167–1186, 1991.

[35] R. Manthalkar, P. Biswas, and B. Chatterji, “Rotation and scale invariant texture classification using gabor wavelets,” in *Texture02*, pp. 87–90, 2002.

[36] T. Randen and J. H. Husoy, "Filtering for texture classification: A comparative study," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 4, pp. 291–310, 1999.