A general discrete-time, adaptive, multidimensional framework is introduced for estimating the motion of one or several object features from their successive non-linear projections on an image plane. The motion model consists of a set of linear difference equations with parameters estimated recursively from a non-linear observation equation. The model dimensionality corresponds to that of the original, non-projected motion space thus allowing to compensate for variable projection characteristics such as panning and zooming of the camera. Extended recursive least-squares and linear-quadratic tracking algorithms are used to adaptively adjust the model parameters and minimize the errors of either smoothing, filtering or predicting the object trajectories in the projection plane. Both algorithms are derived using a second order approximation of the projection nonlinearities. All the results presented here use a generalized vectorial notation suitable for motion estimation of any finite number of object features and various approximations of the nonlinear projection. An example of motion estimation in a sequence of video frames is given to illustrate the main concepts.
Adaptive Model-Based Motion Estimation
by
Regis J. Crinon

A THESIS
submitted
to the Oregon State University

in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

Completed May 27, 1993
Commencement June 1994
# Table of Contents

1. **BACKGROUND**
   1.1 MACHINE VISION .................................................. 1
   1.2 OPTICAL FLOW .................................................... 4
       1.2.1 OPTICAL FLOW FIELDS ....................................... 4
       1.2.2 MOTION RECONSTRUCTION ................................. 6
   1.3 FEATURE CORRESPONDENCE ...................................... 8
       1.3.1 THE CORRESPONDENCE PROBLEM .......................... 8
       1.3.2 MOTION RECONSTRUCTION ................................. 9
   1.4 DISPLACEMENT FIELDS ........................................... 11
       1.4.1 METHODS IN THE SPATIAL DOMAIN ..................... 11
       1.4.2 METHODS IN THE FREQUENCY DOMAIN .................. 13
   1.5 ORGANIZATION OF THE THESIS ................................ 15

2. **INTRODUCTION** .................................................... 18
   2.1 MOTION ESTIMATION IN LONG SEQUENCES ..................... 18
   2.2 MOTION MODEL .................................................. 19
       2.2.1 MODEL .................................................... 19
       2.2.2 ADVANTAGES ............................................... 21
       2.2.3 MULTIDIMENSIONAL SETTING ............................ 23
   2.3 PERSPECTIVE PROJECTION ...................................... 25
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>3.1.1 Non-linear Projection</td>
<td>27</td>
</tr>
<tr>
<td>3.1.2</td>
<td>Perspective Projection</td>
<td>32</td>
</tr>
<tr>
<td>3.1.3</td>
<td>Linear Projection</td>
<td>33</td>
</tr>
<tr>
<td>3.2</td>
<td>3.2.1 Non-linear Projection</td>
<td>34</td>
</tr>
<tr>
<td>3.2.2</td>
<td>Perspective Projection</td>
<td>38</td>
</tr>
<tr>
<td>3.2.3</td>
<td>Linear Projection</td>
<td>39</td>
</tr>
<tr>
<td>3.3</td>
<td>Identifiability</td>
<td>40</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Least-square Framework</td>
<td>41</td>
</tr>
<tr>
<td>3.3.2</td>
<td>General Framework</td>
<td>42</td>
</tr>
<tr>
<td>3.4</td>
<td>Observability</td>
<td>43</td>
</tr>
<tr>
<td>3.4.1</td>
<td>Least-square Framework</td>
<td>43</td>
</tr>
<tr>
<td>3.4.2</td>
<td>General Framework</td>
<td>44</td>
</tr>
<tr>
<td>3.4.3</td>
<td>Depth from Zooming</td>
<td>45</td>
</tr>
<tr>
<td>3.5</td>
<td>Constraints</td>
<td>46</td>
</tr>
<tr>
<td>3.5.1</td>
<td>Polar Decomposition</td>
<td>47</td>
</tr>
<tr>
<td>3.5.2</td>
<td>Skew Matrix Decomposition</td>
<td>48</td>
</tr>
<tr>
<td>4.1</td>
<td>General Case</td>
<td>51</td>
</tr>
<tr>
<td>4.2</td>
<td>Linear Case</td>
<td>58</td>
</tr>
<tr>
<td>4.3</td>
<td>Aliasing</td>
<td>60</td>
</tr>
<tr>
<td>4.3.1</td>
<td>Velocity Magnitude</td>
<td>60</td>
</tr>
<tr>
<td>4.3.2</td>
<td>Velocity Aliasing</td>
<td>65</td>
</tr>
</tbody>
</table>
Table of Contents, Continued

5 MOTION BANDWIDTH 80
5.1 DUALITY ......................................... 80
5.2 THE MULTIDIMENSIONAL RLS ALGORITHM .... 84
   5.2.1 LEAST-SQUARE SOLUTION ................. 84
   5.2.2 RECURSIVE LEAST-SQUARE ............... 86
5.3 BANDWIDTH OF THE RLS ALGORITHM ...... 88
   5.3.1 A-POSTERIORI ERROR ...................... 88
   5.3.2 BANDWIDTH TUNING ....................... 90

6 APPLICATIONS 98
6.1 MOTION ESTIMATION ............................. 98
   6.1.1 VISUAL MODELS .......................... 98
   6.1.2 FEATURE MATCH VALIDATION ............ 103
   6.1.3 OCCLUSIONS ............................ 103
   6.1.4 GENERAL SCHEME ...................... 104
6.2 MOTION WHITENING ........................... 106
6.3 FRACTIONAL INTERPOLATION ............... 108
6.4 PLANAR MOTION .............................. 113

7 SIMULATIONS 115
7.1 SYNTHETIC MOTION .......................... 115
7.2 DIGITAL VIDEO SEQUENCE .................. 121

8 CONCLUSION 128
8.1 CONTRIBUTION ............................... 128
8.2 FUTURE WORK ............................... 130
Table of Contents, Continued

8.2.1 HIGHER MODEL ORDERS 131
8.2.2 DESCRIPTOR SYSTEMS 131
8.2.3 STATISTICAL FRAMEWORK 133
8.2.4 INTEGRATION OF EARLY VISION 134

BIBLIOGRAPHY 136

APPENDICES

A NON-LINEAR PROJECTION APPROXIMATION 154
B NORM OF THE MODELING ERROR 158
C EXTENDED RECURSIVE LEAST-SQUARE 162
D DIFFERENTIATION WITH RESPECT TO A MATRIX 165
E STOCHASTIC APPROXIMATION 169
F A-POSTERIORI ERROR CALCULATION 171
G ESTIMATION BANDWIDTH FOR CONSTANT SIGNAL TO NOISE RATIO 176
H DESCRIPTOR SYSTEM BASED ON ROTATION MATRIX 179
List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>PERSPECTIVE PROJECTION</td>
<td>26</td>
</tr>
<tr>
<td>2.</td>
<td>SINUSOIDAL SIGNAL IN MOTION</td>
<td>62</td>
</tr>
<tr>
<td>3.</td>
<td>VELOCITY AS AN ANGLE $\Psi$</td>
<td>64</td>
</tr>
<tr>
<td>4.</td>
<td>EXAMPLE OF SPATIAL ALIASING</td>
<td>66</td>
</tr>
<tr>
<td>5.</td>
<td>EXAMPLE OF TEMPORAL ALIASING</td>
<td>67</td>
</tr>
<tr>
<td>6.</td>
<td>MAXIMUM VELOCITY FOR RADIAL HIGH-PASS FILTER</td>
<td>70</td>
</tr>
<tr>
<td>7.</td>
<td>CIRCULAR VELOCITY BANDWIDTH</td>
<td>71</td>
</tr>
<tr>
<td>8.</td>
<td>CONTOUR FOR DETERMINING VELOCITY BANDWIDTH</td>
<td>72</td>
</tr>
<tr>
<td>9.</td>
<td>CLOVER-SHAPED VELOCITY BANDWIDTH</td>
<td>74</td>
</tr>
<tr>
<td>10.</td>
<td>CONTOUR FOR DETERMINING VELOCITY BANDWIDTH</td>
<td>75</td>
</tr>
<tr>
<td>11.</td>
<td>VELOCITY BANDWIDTH FOR FIELD-INTERLACED SAMPLING GRID</td>
<td>77</td>
</tr>
<tr>
<td>12.</td>
<td>CONTOUR FOR DETERMINING VELOCITY BANDWIDTH</td>
<td>78</td>
</tr>
<tr>
<td>13.</td>
<td>ISOTROPIC WIDE-ANGLE PROJECTION</td>
<td>99</td>
</tr>
<tr>
<td>14.</td>
<td>MOTION ESTIMATION SYSTEM</td>
<td>105</td>
</tr>
</tbody>
</table>
List of Figures, Continued

15. MOTION WHITENING SYSTEM ........................................ 107
16. FRACTIONAL MOTION INTERPOLATION .......................... 111
17. HELICOIDAL MOTION MODELED WITH $\Phi$ .................. 117
18. HELICOIDAL MOTION MODELED WITH $\Phi$ and $T$ ........ 118
19. OSCILLATING MASS .................................................. 118
20. OSCILLATIONS MODELED WITH $\Phi$ ............................ 120
21. OSCILLATIONS MODELED WITH $\Phi$ and $T$ ............... 120
22. PING-PONG BALL TRAJECTORY ................................. 122
23. FRAME 1 - FIELD 1 .................................................. 123
24. FRAME 7 - FIELD 1 .................................................. 123
25. FRAME 13 - FIELD 1 .................................................. 123
26. FRAME 61 - FIELD 1 .................................................. 124
27. FRAME 64 - FIELD 2 .................................................. 124
28. FRAME 68 - FIELD 2 .................................................. 124
29. FILTERED PING-PONG BALL TRAJECTORY ..................... 125
30. PREDICTED PING-PONG BALL TRAJECTORY ................... 126
31. PREDICTION AT FRAME 7 - FIELD 2 ......................... 127
32. PREDICTION AT FRAME 13 - FIELD 1 ....................... 127
33. PREDICTION AT FRAME 69 - FIELD 2 ....................... 127
Chapter 1

BACKGROUND

1.1 MACHINE VISION

One of the tasks routinely performed by the human brain is to synthesize an understanding of surroundings from light captured by the eyes. For more than thirty years, neurophysiologists have conducted studies on animals to understand the role of each layer of neuron cells in the human visual system [56, 57]. Two main levels, low and high, can be distinguished. The low-level vision system is made of neuron layers which do not use specific a-priori knowledge about the scene being viewed. At the bottom, physical sensors on the retina called rods and cones are designed to capture light and color reflected or produced by the surrounding objects. These light receptors along with intermediate spatial/temporal detectors cells pass visual information to the ganglion cells of the retina. The outputs of these ganglion cells are fed to various parts of the brain producing mental visual fields. Each of these fields represents the response of specialized neuron layers tuned to detect either position, spatial frequency or orientation [24]. Subsequent layers of neurons ensure increasingly more abstract interpretations of the scene by combining the information coming from lower neuron layers. The resulting low-level vision system is a set of visual parame-
ters such as spatial extent, spatial position (depth), surface orientation, shape, texture and motion [59]. Each of these attributes is the result of complex combinations of more primitive information. For example, the depth information results either from an analysis of the disparities between the left and right eye image (stereo cues) [61], or from bringing the object into focus (accommodation cues) or finally from the fact that objects far away in the background seem to move slower than objects at the forefront (motion parallax). The high-level vision system brings a-priori knowledge and reasoning to complement the information produced by the low-level vision system. It synthesizes global concepts about the object structure and its motion and provides complementary information to resolve ambiguities and occlusions. The boundary between these two levels is not well defined. Their relative importance changes dynamically depending on the challenges raised by the scene being visualized or whether the eye is in tracking mode or not. In the last 15 years there has been a large research effort to emulate the functionality of the human visual system and to integrate the results either in robotic-computer vision or video compression and processing systems. The research has mainly been focused on designing early vision system mimicking the low-level human visual system. Complex coupled mechanisms have been proposed to recover simultaneously the shape, volume and motion parameters of a moving object from its perspective two-dimensional projection on the retina [96]. The recognition of the existence of spatial-frequency-tuned filter banks in the human visual system led to the development of multiresolution algorithms where spatial detail is gradually and selectively added to a coarse representation of the image [75]. Most of the tasks involved in early vision involve solving inverse optics equations and therefore face the challenge of ill-conditioned numerical problems [17, 129] and must han-
dle the nonlinear perspective projection, which occurs both in the eye and in cameras. Common modules making up a machine vision system are from low to high complexity edge detection, segmentation, optical flow field, depth recovery, surface orientation and interpolation, shape derived from shading/texture and three-dimensional motion interpretation. In edge detection, one is interested in locating edges in the image reliably [74, 25]. Segmentation involves connecting the edges together to reveal homogeneous regions in the image [92]. Optical flow is the instantaneous velocity field in the image. The terminology comes from the case where a translating sensor generates a motion flow from a point at infinity called Focus Of Expansion (FOE). Calculation of the optical flow field is one of the early tasks performed by the low-level human visual system. It was argued that it is generated from time derivatives and the slope at the zero crossings in the second derivative image intensity field [110]. These zero crossings are perceived in the receptive field as the result of a convolution of the image intensity field with a two dimensional Gaussian low pass filter followed by a Laplacian differentiation [74]. Surface interpolation is concerned with reconstructing the surface of an object from depth information [44]. A closely related topic is shape from shading where the goal is to infer the shape of the object from the intensity values registered by the camera [51]. Other modules such as contour, motion and stereo can be used to determine the shape of an object [5]. Likewise, the recovery of the three dimensional motion parameters must rely on several modules including segmentation, optical flow, depth, surface orientation and shape. It also depends on an eye tracking reflex mechanism designed to establish feature correspondences over time. This is the most complex problem to be solved by a computer vision system as it involves both spatial and temporal dimensions. A review of the work done in this domain is presented
in the next three sections. The first section deals with the estimation of the optical flow field and how it can be used to reconstruct the three dimensional motion of objects. The second section deals with the estimation of the objects three-dimensional motion parameters from feature correspondences. The third section presents motion estimations which rely uniquely on frame to frame image intensity values to derive frame to frame displacement parameters. These techniques are not related to any of the operations performed by the human visual system and have been proposed for the sole purpose of solving specific computer vision problems. For the purpose of machine vision, the images at hand are monocular or stereo sequences of frames or interleaved fields. The scope of this review is to cover the problem of motion estimation from monocular images only. This is the most important problem to solve as the advantages offered by stereo vision are lost when an object is more than twenty meters away from our eyes or from the camera [81]. The monocular sequences used in computer vision are sampled along the horizontal, vertical and temporal dimension. The sampling rate along the vertical dimension determines the number of lines per frame while the temporal sampling frequency determines the number of frames per second. Spatio-temporal sampling bring the issue of motion aliasing [28]. Neurophysiological studies indicate that the low-level human visual system display limitations similar to the ones brought by aliasing [22].

1.2 OPTICAL FLOW

1.2.1 OPTICAL FLOW FIELDS

The problem of computing the optical flow in video sequences has received wide attention in the last few years. The methods rely on the instantaneous spatial and temporal derivatives of the image luminance and chrominance values to
estimate a dense instantaneous velocity field [68]. Let \( I(x, y, t) \) be the intensity function of the image at time \( t \) and at horizontal and vertical positions \( x \) and \( y \), respectively. From the assumption that the light intensity emitted or reflected by the feature does not change in time, it can be written

\[
0 = \frac{dI(x, y, t)}{dt} = \frac{\partial I}{\partial x} \frac{\partial x}{dt} + \frac{\partial I}{\partial y} \frac{\partial y}{dt} + \frac{\partial I}{\partial t} = I_xv_x + I_yv_y + I_t \tag{1.1}
\]

where \( v_x \) and \( v_y \) denote the horizontal and vertical velocity, respectively, and \( I_x \), \( I_y \) and \( I_t \) denote the horizontal, vertical and temporal image intensity gradient, respectively. Let \( \mathbf{v} \) be the two dimensional vector \([v_x, v_y]^H\) where \( H \) denotes the Hermitian operator. Velocity vectors \( \mathbf{v} \) can be associated to every sample in the image in this manner by way of measuring the horizontal, vertical and temporal intensity gradients. Unfortunately, equation (1.1) does not determine \( \mathbf{v} \) uniquely. In fact, it can be seen from equation (1.1) that the component of the vector \( \mathbf{v} \) parallel to the vector \([I_y, -I_x]^H\) cannot be observed. Only the component parallel to the edge can be determined. This component is parallel to the vector \([I_x, I_y]^H\). This limitation is often referred to as the aperture problem [129]. Other problems arise in equation (1.1) when spatial gradients are small. Various strategies have been proposed to overcome these deficiencies. One of them is to assume that the image flow field is locally constant [39] but this assumption breaks down when the object is rotating. Another possibility is to use smoothness constraints [50, 127, 82] to propagate the velocity field inward from the boundaries of the object where spatial derivatives display large values. However, this approach uses a-priori smoothing constraints which cannot reproduce discontinuities in the flow field. Recently stochastic relaxation techniques have been proposed to overcome this limitation [11, 62]. They involve
introducing line processes which are stochastic indicator functions for modeling the discontinuities in the image. Other smoothness concepts such as oriented smoothness have been proposed to impose constraints only in the direction parallel to the edge [83, 84, 122]. Multigrid methods can then be used to propagate the solution over an entire object [38].

The equality shown in equation (1.1) no longer holds when the light reflected or emitted by an object varies from one image frame to another or when the object is rotating or when the object surface is not Lambertian [112]. In this case, either additional terms must be accounted for [85] or extended optical flow equations must be used [86, 31, 113]. Other methods for computing the optical flow have been investigated. Most of them attempt to mimic the functionality of the human visual system by relying on spatial or temporal filters [93, 100]. Zero crossings of the second derivative of a temporal Gaussian smoothing function have been used to detect moving edges and to estimate the normal velocity flow [35].

1.2.2 MOTION RECONSTRUCTION

In parallel with the problem of estimating the optical flow fields, computer vision researchers have sought to determine whether this information can be used toward reconstructing the three dimensional motion of an object under perspective projection. The instantaneous velocity field \([\hat{x}, \hat{y}, \hat{z}]\) of a point positioned at \([x, y, z]\) in a three dimensional space can be written as

\[
\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{z}
\end{bmatrix} =
\begin{bmatrix}
0 & -\omega_z & \omega_y \\
\omega_z & 0 & -\omega_x \\
-\omega_y & \omega_x & 0
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z
\end{bmatrix} +
\begin{bmatrix}
T_x \\
T_y \\
T_z
\end{bmatrix}
\]
\[ \omega = \Omega \begin{bmatrix} x \\ y \\ z \end{bmatrix} + T \]  

(1.2)

where the elements of \( \Omega \) and \( T \) are the angular and the translational velocity, respectively. It can be shown that in the case of perspective projection, the optical flow at a point in the view-plane is analytically defined by a quadratic form of its coordinates. Moreover, it was proven that \( \Omega \) and \( T \) and the depth of the object with respect to an observer can never be recovered from an optical flow field alone [95]. This result was then generalized to the concept of ambiguous curves defining ambiguities between the observed optical flow and the underlying motion of the object [15]. Similar ambiguities exist when the optical flow field is noisy [1]. It was shown that in the most general case, five optical flow vectors at five different points are required to provide a finite number of possible solutions (up to a scale factor) for \( \Omega \) and \( T \) [79, 80]. There are in fact ten possible solutions [77]. A linear algorithm was derived requiring eight optical flow vectors to reconstruct the instantaneous motion [132]. In the case of motion parallax, the \( \Omega \) and \( T \) can be determined in a more direct fashion from the focus of expansion [70]. Motion parallax occurs when the projected trajectories of two objects at different depths intersect. Likewise, it was shown that \( \Omega \) and \( T \) can also be estimated from the first derivatives of the optical flow produced by a planar patch in motion [116, 124, 70]. Only the first order spatial derivatives of the optical flow are needed when temporal variations of the flow are taken into account [103]. In [133], it is shown that for the case of a planar patch, the motion parameters can be obtained without knowing the optical flow vectors. Finally, it was recently demonstrated that the problem of inferring the motion parameters \( \Omega \) and \( T \) can be recast into calculating the eigenvalues and
eigenvectors of a $3 \times 3$ symmetric matrix built from linear combinations of the coefficients of a quadratic form [36].

1.3 FEATURE CORRESPONDENCE

1.3.1 THE CORRESPONDENCE PROBLEM

The problem of matching points or features over two or more frames has received wide attention because feature correspondences provide the basis for building a long-term token based understanding of the motion of objects from their projections. This functionality is found at the top of the low-level human visual system [13]. Psychological studies have indicated that a surprisingly small numbers of points are necessary for the human visual system to interpret motion correctly [60]. The rules used in human and computer vision to set matches among various images features can be drawn from various attributes such as similar location, line length, pixel (picture sample) intensity, line orientation, curvature or image intensity values. Features can be clusters of points, lines, corners and zero-crossings among others. The search of such attributes and features constitutes the basis for the research work done for solving the correspondence problem. It was understood early that there are in fact three different problems. First, the problem of choosing the image feature. Second, the problem of setting correct matches. Third, the problem of evaluating the quality of the match. With regard to the first problem, it was recognized that highly distinguishable tokens must be chosen in the image so ambiguities are minimized. This suggests that operators measuring variance, edge strength, amount of detail, disparity or discontinuity be used to isolate features in the image. These observations led to deriving edge detection techniques to identify image locations where zero-crossings of the Laplacian and Gaussian curvature
occur [74, 33, 32]. Gray corners were defined as the edge points where the first and second derivatives of the image intensity values are maximum along one direction and minimum in the other. Regarding the second and third question, normalized cross-correlation, absolute and square difference have all been used to grade matching candidates [2]. The first matching techniques incorporating these features were iterative relaxation algorithms where the intensity at the working pixel was compared with intensities of pixels lying in a small spatio-temporal neighborhood of the first pixel [12]. However, problem arise with variations in illumination, reflectance, shape and background as well as occlusions. Global edge description [2], the use of long sequences [98] as well as of spatial multiresolution techniques can improve the matching process [121]. Other suggestions for solving the correspondence problem include a stochastic setup for modeling the matching process as a statistical occupancy problem [45] and an iterative estimation of a continuous transformation between points on two contours using differential geometry techniques [125].

1.3.2 MOTION RECONSTRUCTION

Following the correspondence problem comes the problem of using matching features to estimate the three dimensional motion parameters of a rigid object. Due to the discrete nature of the correspondence problem, instantaneous motion parameters can no longer be estimated. Here, the motion parameters come from the following model

\[
\begin{bmatrix}
    x_{k+1} \\
y_{k+1} \\
z_{k+1}
\end{bmatrix} = \mathbf{R} \begin{bmatrix}
x_k \\
y_k \\
z_k
\end{bmatrix} + \begin{bmatrix}
    T_x \\
    T_y \\
    T_z
\end{bmatrix}
\] (1.3)
where $p_{k1} = [x_k, y_k, z_k]^H$ and $p_{k+1} = [x_{k+1}, y_{k+1}, z_{k+1}]^H$ is the position of the feature at time $k$ and $k + 1$, respectively and where $R$ is a $3 \times 3$ rotation matrix. The matrix $R$ has 6 degrees of freedom. The vector $T$ is again a translation vector. This model corresponds to the “two-view” model based on the fundamental dynamical law stating that the motion of a rigid object between two instants of time can always be decomposed into a rotation about an axis followed by a translation. Several key technical contributions addressing this problem have been made in the last ten years [105, 71, 106, 108, 118]. First it was realized that with monocular images, the translation $T$ and the depth $z_k$ can be determined only up to a scale factor because of the perspective projection. It was also shown that it is possible to recover the motion and structure parameters of a rigid planar object using four correspondences in three consecutive views. At most two solutions exist when only two views are used [53]. In the general case of a non-planar object, it was found that at least 8 such correspondence points in two consecutive camera views are required to recover $R$ and $T$ [107] except when the points lie in a particular quadric surface in which case no more than three solutions exist [72, 73]. The proof relies on building the skew symmetric matrix $S$

\[
S = \begin{bmatrix}
0 & -t_y & t_x \\
t_x & 0 & -t_z \\
-t_y & t_z & 0
\end{bmatrix}
\]  

and noticing that we always have $p_{k+1}^H S p_{k+1} = 0$ since the matrix is skew symmetric. By using equation (1.3) in equation (1.4), a matrix of essential parameters is introduced which can only be estimated up to a depth scale factor. A singular value decomposition of $S$ yields the parameters of the rotation matrix $R$ and the translation $T$. Unified approaches were then formulated whereby
noise is reduced by using an overdetermined system of equations [120, 131]. One of these approaches calls for the Rodrigues formula [18] for rotation matrices which makes explicit use of only three parameters [120]. The advantage of this method is to produce an estimated rotation matrix which is orthogonal, meaning that $RR^H = R^HR = I$ where $I$ is the $3 \times 3$ identity matrix. This condition is not automatically ensured in the other methods. Finally, it should be noted that the use of long sequences has been investigated for the particular case where frame to frame dynamics remain small. In this case, calculus of variations techniques can be applied to update the rotation matrix and the translation [109].

1.4 DISPLACEMENT FIELDS

1.4.1 METHODS IN THE SPATIAL DOMAIN

A few attempts have been made to recover the rotation matrix $R$ and the translation vector $T$ shown in equation (1.3) without the help of feature correspondences. The methods rely on the spatial and temporal intensity gradients in the image [54, 89] and recursive algorithms have been derived [89] when the rotation is infinitesimal. However, the two-view model does not hold in general because of object deformations, object acceleration, camera zoom, camera pan, occlusions and depth variations. For this reason, algorithms designed for estimating pixel motion as opposed to object motion have been developed. Ideally, a vector is calculated at every sample in the image hence providing a match and a “motion” model at the same time. Note that this is different from the correspondence problem where only a few features are chosen and where the motion parameters are three dimensional. In some sense, displacement fields can be viewed as optical flow fields in a non-instantaneous setting. However,
they do not display the same high degree of spatial smoothness. Just like in velocity flow fields, a displacement vector can be seen as the result of a linear approximation between frame difference and spatial gradient. This idea led to the development of a recursive minimization of "displaced frame difference" representing the difference in intensity between a pixel in a frame at time \( t + \Delta t \) and its original position in the previous frame at time \( t \) [87, 88]. Assume that \( \vec{d}_i^{i-1} = [d_x^{i-1}, d_y^{i-1}]^T \) is an estimate of the displacement of a pixel from frame \( t \) to \( t + \Delta t \) at the \( i \)-th iteration. The update equation is

\[
\vec{d}_i^{i+1} = \vec{d}_i^i - \varepsilon \text{DFD} \left( x_0, y_0, \vec{d}_i^{i-1} \right) \nabla I \left( x_0 - d_x^{i-1}, y_0 - d_y^{i-1}, t \right)
\]  

(1.5)

where \( [x_0, y_0]^T \) are the coordinates of the pixel in frame \( t + \Delta t \), \( \nabla I() \) is the spatial gradient and where

\[
\text{DFD} \left( x_0, y_0, \vec{d}_i^{i-1} \right) = I(x_0, y_0, t + \Delta t) - I(x_0 - d_x^{i-1}, y_0 - d_y^{i-1}, t)
\]  

(1.6)

is the displaced frame difference. The constant \( \varepsilon \) is chosen such that \( 0 \leq \varepsilon \leq 1 \) and can be replaced by the variable gain in a Newton algorithm [23]. Other similar recursive approaches were suggested but they all can be viewed as techniques aligning pixels for maximum frame to frame correlation [16, 128]. One of these methods accounts for small variations in illumination or object reflectance [102] and another proposes more than 2 successive frames to calculate the displacement field [55].

Displacement estimation techniques which do not rely on spatial gradient have also been proposed. Among the most popular ones are block-matching methods. Such methods consist in finding the best frame to frame pixel alignment by matching a small neighborhood centered about the pixel of interest in an image at frame \( t \) with similar neighborhoods in a frame at time \( t + \Delta t \). The neighborhoods are usually small square blocks of pixels. The displacement vec-
tor at the pixel of interest describes the relative position of the block in frame $t + \Delta t$ which displays the highest similarity with the original block. Normalized cross-correlation, mean-square error and sum of the absolute differences have been proposed as minimization criterion [101, 58, 40]. Stochastic frameworks methods can be used to increase the smoothness of displacement fields by building either spatial autoregressive models [37] or conditional estimates from the adjacent blocks [14]. While block-matching schemes do not provide sub-pixel motion accuracy as pel-recursive methods do, they work well for fast motions. An attempt to bring the advantages of pel-recursive and block matching algorithms together into a unified algorithm is reported in [126]. Because both pel-recursive and block-matching techniques have drawbacks, other researchers considered other alternatives. One approach consists in approximating the image intensity $I(x, y, t)$ locally by a polynomial in $x, y, t, xy, x^2, y, x^2, y^2, xt, yt$ and to derive the displacement field from the coefficients of the polynomial. Large displacements are handled through the use of a multiresolution algorithm allowing the displacement estimates to be iteratively refined [76, 63].

1.4.2 METHODS IN THE FREQUENCY DOMAIN

Besides spatio-temporal techniques, methods based on Fourier analysis have been proposed to generate displacement fields. These methods attempt to exploit the fact that the three dimensional Fourier Transform of an object in translation is confined to a plane [115]. The equation of this plane is now derived. Assume that an object is moving at constant speed over a uniform background in the time/space continuous image $I(s, t)$. The vector $v$ is a two dimensional vector $v = [v_x, v_y]^H$, where $v_x$ and $v_y$ denote the horizontal and vertical velocity, respectively, and the vector $s = [x, y]^H$ is a two dimensional vector denoting
the spatial coordinates of a point in the image. The image \( I(s, t) \) is a shifted version of the image \( I(s, 0) \). Thus their Fourier transforms are related by

\[
\mathcal{F}_{s,t} \left\{ I(s, t) \right\} = \mathcal{F}_{s,t} \left\{ I(s - vt, 0) \right\}
\]

where

\[
\mathcal{F}_{s,t} \left\{ I(s, t) \right\} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} I(s, t) e^{-j2\pi(f_s s + ft t)} \, ds \, dt
\]

is the Fourier transform with respect to the spatial coordinate \( s \) and time \( t \). The scalar \( f_t \) is the temporal frequency and the vector \( f_s = [f_x, f_y]^H \) is a two dimensional vector where \( f_x \) and \( f_y \) is the horizontal and vertical frequency, respectively. Substitute definition (1.8) into equation (1.7) to get

\[
\begin{align*}
\mathcal{F}_{s,t} \left\{ I(s, t) \right\} &= \mathcal{F}_t \left\{ I \left( f_s \right) e^{-j2\pi(f_s s + ft t)} \right\} \\
&= \mathcal{I} \left( f_s \right) \mathcal{F}_t \left\{ e^{-j2\pi(f_s s + ft t)} \right\} \\
&= \mathcal{I} \left( f_s \right) \delta \left( f_t + f_s^H v \right)
\end{align*}
\]

where \( \mathcal{I}(\cdot) \) is the Fourier transform of the image at time \( t = 0 \) and where \( \delta(\cdot) \) denotes the Delta function. The time shift property of the Fourier transform has been used to go from equation (1.7) to equation (1.9). It follows that temporal and spatial frequencies are related to each other by

\[
f_x v_x + f_y v_y + f_t = 0 \quad \text{(1.10)}
\]

Likewise, the two-dimensional (spatial) Fourier transform of the image frame at time \( t + \Delta t \) is related to the Fourier transform of image frame at time \( t \) by the following phase shift relationship

\[
\mathcal{F}_{x,y} \left\{ I(x, y, t + \Delta t) \right\} = e^{-j2\pi(f_x v_x + f_y v_y)\Delta t} \mathcal{F}_{x,y} \left\{ I(x, y, t) \right\}
\]

\text{(1.11)}
where $F_{x,y} \{I(x,y,t)\}$ is the two-dimensional Fourier transform of the image with respect to the horizontal and vertical dimension. Either equation (1.10) or (1.11) can be used to estimate displacement fields, but the latter is often preferred for its lower complexity [46, 64]. Practically, equation (1.11) is used by calculating the cross-power spectrum, dividing it by its magnitude to get a dirac at $[v_x \Delta t, v_y \Delta t]^H$ as shown below:

$$
\frac{F_{x,y} \left\{ I(x,y,t+\Delta t) \right\}}{|F_{x,y} \left\{ I(x,y,t+\Delta t) \right\}|} F_{x,y} \left\{ I(x,y,t) \right\}^H = e^{-j2\pi(f_x v_x + f_y v_y)\Delta t}
$$

The results can be extended to rotating [30] and accelerating objects. In the latter case, the Fourier transform includes terms in $f_t^2$ making the motion no longer bandlimited in the temporal dimension [27]. The most serious problem with these frequency domain techniques is the fact that images are non-stationary two dimensional signals. Short-time Fourier techniques may be used to take into account the fact that stationarity is verified only locally.

### 1.5 ORGANIZATION OF THE THESIS

Because neither of the three families of motion estimation algorithms presented above provides a general setting for long-range motion, a general multidimensional, time-varying, model is introduced in Chapter 2. The model incorporates mechanisms for capturing both slow and fast motion modes. The slow motion modes are embodied by a linear time-varying matrix and the fast modes are collected by a complementary vector. The matrix is called the transition matrix as it establishes a link among consecutive positions of the same object feature. The vector is called a translation vector. In Chapter 3, the estimation of the transition matrices $\Phi_{k+1}$ is considered. The translation $T_{k+1}$ is set to
zero so as to make the transition matrices take up as much motion as possible. The derivation are done for both the time-invariant and time-varying case in the framework of a multi-dimensional least-square minimization. Identifiability conditions are then studied for the solutions. The minimum number of feature correspondences required to estimate the transition matrix is calculated for both the time invariant and time-varying systems as well as for linear and perspective projection. The conditions for establishing the observability of the positions of an object feature are then studied. Again, the number of view-plane coordinates required to reconstruct the original feature positions is determined for both the time-invariant and time-varying case as well as for the linear and perspective projection case. At the end of Chapter 3, algebraic techniques for projecting the estimated transition matrix onto a desired class of motion are reviewed. The class of orthogonal matrices is given special attention. Chapter 4 is concerned with the derivation of the translation vector once the transition matrix is known. The method is based on a multidimensional tracking algorithm allowing the translation vector to capture and smooth the remaining motion modes. The effect of spatio-temporal sampling on the translation vectors is analyzed in term of motion aliasing. Classes of motion-aliasing free translation magnitudes are derived for various sampling lattices. Chapter 5 proceeds with establishing a duality between the problem of estimating transition matrices and the problem of estimating translation vectors. It is shown that this duality raises the issue of estimation bandwidth allocation among the two sets of motion parameters. A general bandwidth allocation policy is proposed based on making the motion model operate at a constant motion uncertainty level. Chapter 6 offers a review of several domains of applications for the multi-dimensional motion model. This includes a general setup for estimating the motion parameters from low-
level vision attributes such as optical flow and matching correspondences. It is shown that in this setup, the motion model can be made to fail gradually when occlusions occur. Non-linear projection models are also proposed to take into account the properties of the human visual system. A methodology for using the motion model to interpolate image frames is also proposed and new concepts such as motion whitening are introduced. Chapter 7 provides simulation results on both synthetic and digital video data. The performance of the motion model is analyzed for both the cases of filtering and estimating an object trajectory. Chapter 8 brings a general review of the results. It also offers a view of future extensions/developments of the model.
Chapter 2

INTRODUCTION

2.1 MOTION ESTIMATION IN LONG SEQUENCES

The problem of developing an understanding of the long range motion of an object cannot be resolved as easily as the two-view problem. In the two-view case, frame-to-frame motion can be modeled as a rotation and a translation which do not have to bear any resemblance with the actual motion of the object. For image sequences consisting of more than two frames, this model can be used as long as the object motion is constant [99]. However, in general, object dynamics over long image sequences result from the combined effect of external forces, ego-motion and inertia tensors. Non-linear projection, illumination and reflectance variations bring additional ambiguities making the task of developing a long-term motion understanding vision system even more complex. Nevertheless, simple experiments based on rotation and translation models have demonstrated some of the benefits of using long image sequences. For example, trajectories of feature points can be determined with a higher level of confidence by monitoring the smoothness of their paths [98]. Also, robustness of the estimates against noise is increased [119]. Slightly more complex motion models
have recently been investigated. In [130], it is assumed that the velocity is constant so the translation is a linear function of time and in [109, 119], recursive updates of the motion parameters are proposed for the case of small frame to frame displacements. In particular, variational techniques are used in [109].

2.2 MOTION MODEL

2.2.1 MODEL

The purpose of this section is to introduce a generalized motion model which can be used for a broader class of motion. In the real world, dynamics of objects can include sudden accelerations or decelerations, collisions, occlusions, or spinning, to name a few possibilities. To get an idea of how to design a robust motion model, take the case of Newtonian motion. Let \( \mathbf{x} \) and \( \mathbf{\dot{x}} \) be the position and the velocity of a point or feature in an \( m \) dimensional image space and assume that its motion is governed by the following continuous-time state equation:

\[
\begin{bmatrix}
\mathbf{\dot{x}} \\
\mathbf{\ddot{x}}
\end{bmatrix}
= \mathbf{A}
\begin{bmatrix}
\mathbf{x} \\
\mathbf{\dot{x}}
\end{bmatrix}
+ \begin{bmatrix}
0 \\
\mathbf{B}
\end{bmatrix}
\mathbf{y}
\]

\[
y = h(\mathbf{x}, t)
\]

(2.1)

where \( \mathbf{y} \) and \( \mathbf{y} \) are \( n \) and \( \ell \) dimensional vectors respectively and \( h() \) is an \( \ell \)-dimensional non-linear projection operator which can vary with time. The matrices \( \mathbf{A} \) and \( \mathbf{B} \) have the dimensions \( 2m \times 2m \) and \( m \times n \) respectively. The vector \( \mathbf{y} \) is a function of time and accounts for acceleration or external forces. The vector \( \mathbf{y} \) is the position of the projected point (e.g. on the screen of video frame). Introducing time-sampling, the continuous time state equation becomes
the following discrete time state equation:

\[
\begin{bmatrix}
\dot{x}_{k+1} \\
\dot{\dot{x}}_{k+1}
\end{bmatrix} = e^{AT} \begin{bmatrix}
\Phi_{11} & \Phi_{12} \\
\Phi_{21} & \Phi_{22}
\end{bmatrix} \begin{bmatrix}
x_k \\
\dot{x}_k
\end{bmatrix} + \int_{kT}^{(k+1)T} e^{A((k+1)T-\tau)} \begin{bmatrix}
0 \\
B
\end{bmatrix} u(\tau) d\tau
\begin{bmatrix}
\omega_{k,1} \\
\omega_{k,2}
\end{bmatrix}
\]

\[y_k = h_k(x_k) \quad (2.2)\]

where the sampling period is T. The projection operator at sampling time \(kT\) is the \(\ell\)-dimensional operator \(h_k()\). The matrices \(\Phi_{11}, \Phi_{12}, \Phi_{21}, \Phi_{22}\) are \(m \times m\) block matrices. The instantaneous velocity \(\dot{x}_k\) and \(\dot{x}_{k+1}\) are generally not observable. The state equation governing the position of the object feature is

\[x_{k+1} = \Phi_{11}x_k + \Phi_{12}\dot{x}_k + \omega_{k,1}\]

\[y_k = h_k(x_k) \quad (2.3)\]

Clearly, the estimation of \(\Phi_{11}\) and \(\Phi_{12}\) becomes impossible because \(\dot{x}_k\) and \(\omega_{k,1}\) are not known. Note that the object motion could have been much more complex than the simple Newtonian motion state equations presented so far. Consider, for example, the situation where the instantaneous velocity \(\dot{x}_k\) and the "acceleration" term \(u_k\) are both a function of the state \(x_k\). Since the goal is to analyze object motion in the projected space rather than to identify the true object motion, a time-varying adaptive setup is now proposed in which \(\Phi_{k+1}(y)\) is designed to compensate for the terms \(\Phi_{12}\dot{x}_k\) and \(B'\). Remaining dependence on these terms is lumped together to form a velocity field to be used for further analysis and classification of the motion. The following time-varying non-linear model results:

\[x_{k+1} = \Phi_{k+1}x_k + T_{k+1}\]
\[ y_k = h_k (x_k) \] (2.4)

where the matrix \( \Phi_{k+1} \) is a \( m \times m \) matrix and \( T_{k+1} \) is a \( m \times 1 \) column vector. The matrix \( \Phi_{k+1} \) is therefore the embodiment of any linear dependence of the object dynamics on the previous position while the vector \( T \) is the slack vector making up the difference. The \( \ell \)-dimensional projection operator \( h_k() \) is assumed to be known. It can either be a linear or non-linear operator and its dependence upon time is due to the fact that the camera position and parameters might be changing in time (panning and zooming).

### 2.2.2 ADVANTAGES

This section offers a list of advantages offered by the flexibility of the model in equation 2.4.

- The states \( x_k \) can represent more than the successive positions of an individual point or feature. For example, \( x_k \) could be made of a particular linear or non-linear combination of positions such as the centroid or the median value of a set of feature coordinates. The state \( x_k \) can also be the output of a spatial decimation or interpolation filter. This means that the model can be incorporated into hierarchical vision systems based on multi spatial resolution images [121].

- The bandwidth of the estimator for \( \Phi_{k+1} \) and \( T_{k+1} \) can be adjusted independently. This is a valuable feature as it allows the system to monitor motion aliasing in either the transition matrix or in the translation vector.

- It is possible to constrain \( \Phi_{k+1} \) to a desired class of motion without compromising the quality of the modeling as the vector \( T_{k+1} \) acts as a "slack" variable taking up the motion components that do not fit the constraint.
The matrix $\Phi_{k+1}$ is not constrained to be a rotation matrix but can also include object deformations as well as panning and zooming of the camera. If the rotation and zooming parameters of the camera are known, we can account for them by the following modification of the adaptive model:

$$
\begin{align*}
    x_{k+1} &= R_{k+1}\Phi_{k+1}x_k + R_{k+1}T_{k+1} \\
    y_k &= h_k(x_k)
\end{align*}
$$

(2.5)

where $R_{k+1}$ is a matrix denoting the rotation of the camera between sampling instants $k$ and $k+1$. The variable focal distance is included in the dependency of $h_k()$ on time. These focal distance variations must be entered as a parameter of $h_k()$ if they are not pre-programmed or known in advance.

- The system operates in a deterministic setup as we are only interested in analyzing the motion scene at hand and not trying to characterize the object motion on average over many realizations in a stochastic setup.

- When $h_k()$ is linear, a wealth of results can be tapped from linear system theory. In particular, general results regarding identifiability/observability in linear systems can be used to address questions regarding the number of correspondence points required to identify the motion parameters.

- The non-linear projection $h_k()$ can result from a sequence of projections which approximate a given optical camera mapping system. Such an approximation can be selected according to a given objective or subjective quality criterion in the motion analysis system.
2.2.3 MULTIDIMENSIONAL SETTING

Situations where several views of the same object in motion are available arise frequently in image processing. For example, stereo vision systems provide two independent views while television cameras capture three color fields (red, green and blue) of the same view simultaneously. Furthermore, it is highly desirable to develop a motion model for not one but several features of the same object to combat noise and motion ambiguities. It is therefore of prime importance to extend the definition of the model in equation (2.5) to a multidimensional setting allowing concurrent tracking of $p$ points/features. Let $x^{(j)}_k$, $1 \leq j \leq p$, denote the position of the $j$th feature at sampling time $k$. Let $y^{(j)}_k$, $1 \leq j \leq p$ be their respective projected coordinates on the screen. Build the $m \times p$ matrix $X_k$ and $X_{k+1}$, the the $\ell \times p$ matrix $Y_k$ and the $m \times v$ matrix $T_{k+1}$ as

$$X_{k+1} = [x^{(1)}_{k+1}, \ldots, x^{(p)}_{k+1}]$$
$$X_k = [x^{(1)}_k, \ldots, x^{(p)}_k]$$
$$Y_k = [y^{(1)}_k, \ldots, y^{(p)}_k]$$
$$T_{k+1} = [T^{(1)}_{k+1}, \ldots, T^{(1)}_{k+1}]$$

so equation (2.5) can be re-written as

$$X_{k+1} = R_{k+1} \Phi_{k+1} X_k + R_{k+1} T_{k+1}$$
$$Y_k = [h_k(x^{(1)}_k), \ldots, h_k(x^{(p)}_k)]$$

(2.7)

To make equation (2.7) look like equation (2.5), define the $vec()$ operator [90]. Given the matrix $X_k = [x^{(j)}_k, \ldots, x^{(p)}_k]$, the action of this operator is such that

$$vec(X_k) = \begin{bmatrix} x^{(1)}_k \\ \vdots \\ x^{(p)}_k \end{bmatrix}$$

(2.8)
that is \( \text{vec}(X_k) \) is the vector obtained from stacking the consecutive columns of the matrix \( X_k \) to form an \( mp \times 1 \) vector. Now, by taking a \( p \times q \) matrix \( M \) and a \( s \times r \) matrix \( N \) define the Kronecker product of these two matrices as the \( ps \times qr \) matrix

\[
M \otimes N = \begin{bmatrix}
m_{11} & \ldots & m_{1q} \\
\vdots & \vdots & \vdots \\
m_{p1} & \ldots & m_{pq}
\end{bmatrix} \otimes \begin{bmatrix}
n_{11} & \ldots & n_{1r} \\
\vdots & \vdots & \vdots \\
n_{s1} & \ldots & n_{sr}
\end{bmatrix} = \begin{bmatrix}
m_{11}N & \ldots & m_{1q}N \\
\vdots & \vdots & \vdots \\
m_{p1}N & \ldots & m_{pq}N
\end{bmatrix}
\] (2.9)

Let \( 1 = [1, \ldots, 1]^H \) be the \( p \) dimensional column vector where all the entries are equal to one. From equation (2.9), the matrix \( T_{k+1} \) can be re-written as

\[
T_{k+1} = T_{k+1} 1^H
\] (2.10)

Furthermore, a fundamental equality involving the \( \text{vec}() \) and the \( \otimes \) operator is

\[
\text{vec}(MPN) = (N^H \otimes M) \text{vec}(P)
\] (2.11)

where \( P \) is a \( q \times s \) matrix [19]. Now apply the linear operator \( \text{vec}() \) to both sides of the equality shown in equation (2.6) and use equations (2.10),(2.11) to get

\[
\begin{align*}
\text{vec}(X_{k+1}) &= (I_{pxp} \otimes R_{k+1} \Phi_{k+1}) \text{vec}(X_k) + (1 \otimes R_{k+1}) T_{k+1} \\
\text{vec}(Y_k) &= \left[ h_k(x_k^{(1)}), \ldots, h_k(x_k^{(p)}) \right]^H
\end{align*}
\] (2.12)

The motion model described in equation (2.12) now features column vectors for the states and observations just as in equation (2.5). The Kronecker product along with the column stacking operator \( \text{vec}() \) make the motion model operate in a multidimensional setting without any particular constraint. These two operators have also been found to be a valuable design tool for massively parallel image processing systems [7].
2.3 PERSPECTIVE PROJECTION

The motion model in equation (2.12) features a time-varying non-linear projection operator \( h_k() \) which in general is a perspective projection operator. Perspective projection is the non-linear projection occurring in the human visual system as well as in cameras. Assume that the fixed system of axis \( \{X_1, X_2, X_3\} \) is chosen so the depth axis \( X_3 \) is aligned with the axis of the camera as shown in Figure 1. Let \( F_0 \) be the focal distance of the camera and let \( x_3 = 0 \) be the projection plane so the focus point coordinates are \((0,0,-F_0)\), \(F_0 > 0\).

The perspective transformation mapping the point at location \( x = [x_1, x_2, x_3]^H \) to the point at location \( y = [y_1, y_2, 0]^H \) is

\[
y = \frac{Cx + d}{\ell^H x + g}
\]  

(2.13)

where

\[
C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad d = 0, \quad g = 1, \quad \ell = \begin{bmatrix} 0, 0, \frac{1}{F_0} \end{bmatrix}^H
\]  

(2.14)

or equivalently

\[
y_1 = \frac{x_1F_0}{x_3 + F_0} \quad y_2 = \frac{x_2F_0}{x_3 + F_0}
\]  

(2.15)

It follows that in the motion model, the dimensions must be set to \( m = 3 \) and \( \ell = 2 \). Because perspective projection is the most common non-linear projection, it will be used extensively in the following chapters to illustrate some of the key results.
Figure 1. PERSPECTIVE PROJECTION

\[ x = [x_1, x_2, x_3]^H \]

\[ y = [y_1, y_2, 0]^H \]
Chapter 3

ESTIMATION OF THE TRANSITION MATRIX

The motion model introduced in equation (2.4) includes two sets of motion parameters, the transition matrix and the translation vector. The matrix $\Phi_{k+1}$ captures the slow varying modes of the motion and it is a legitimate goal to try to explain as much motion as possible with this matrix alone. The first step toward estimating the parameters of the motion is therefore to set all the entries of the vector $T_{k+1}$ to zero and derive algorithms for estimating $\Phi_{k+1}$.

3.1 TIME-INARIANT MODEL

3.1.1 NON-LINEAR PROJECTION

The situation where the motion parameters are time-invariant is studied first. In this case, the estimation of the transition matrix can be done by using successive positions of the feature in the image frames. In particular, assume that the transition matrix $\Phi$ has been constant over the last $N$ image frames. The motion model is therefore

$$x_{k+1} = R_{k+1} \Phi x_k$$
\[ y_k = h_k(x_k) \] (3.16)

The object motion is assumed to follow this model exactly. Apply the \(\text{vec}(\cdot)\) operator to both sides of the first equality in equation (3.16) and notice that \(\text{vec}(x_{k+1}) = x_{k+1}\). With the help of equation (2.11), equation (3.16) can be re-written as

\[
x_{k+1} = (x_k^H \otimes R_{k+1}) \text{vec}(\Phi)
\]

\[
y_k = h_k(x_k)
\] (3.17)

The estimation of \(\Phi\) is now in order. To achieve this, perform a least-square fit by minimizing the functional

\[
\mathcal{F}_N(\Phi) = \text{trace}\left\{ \sum_{i=1}^{N-1} \beta_i^{[1:N]} E_i^H E_i \right\} = \sum_{i=1}^{N-1} \beta_i^{[1:N]} \text{vec}(E_i)^H \text{vec}(E_i)
\]

\[
= \sum_{i=1}^{N-1} \beta_i^{[1:N]} \text{vec}(Y_{i+1} - h_{i+1} (R_{i+1} \Phi X_i))\text{vec}(Y_{i+1} - h_{i+1} (R_{i+1} \Phi X_i))^H
\] (3.18)

where

\[
Y_{i+1} = \begin{bmatrix} y_{i+1}^{(1)} & \cdots & y_{i+1}^{(p)} \end{bmatrix}
\]

\[
E_i = \begin{bmatrix} e_i^{(1)} & \cdots & e_i^{(p)} \end{bmatrix}
\]

\[
= \begin{bmatrix} y_{i+1}^{(1)} - h_{i+1} (R_{i+1} \Phi x_i^{(1)}) & \cdots & y_{i+1}^{(p)} - h_{i+1} (R_{i+1} \Phi x_i^{(p)}) \end{bmatrix}
\] (3.19)

are both \(\ell \times p\) matrices. The \(l^2\) metric is used as the minimization criterion because of convenience. Since learning the behavior of dynamics is the goal here, it is important to keep in mind that optimization in the \(l^2\) norm leads to simple gradient-based search method widely used in learning algorithms. Therefore, explicit recursive algorithms for estimating the motion model parameters should
be attainable. The values $\beta_i^{[1:N]}$ weight the errors. The expression $\text{vec}(E_i)$ can be further expanded as

$$
\text{vec}(E_i) = \begin{bmatrix}
y_{i+1}^{(1)} - h_{i+1} \left( R_{i+1} \Phi x_i^{(1)} \right) \\
\vdots \\
y_{i+1}^{(p)} - h_{i+1} \left( R_{i+1} \Phi x_i^{(p)} \right)
\end{bmatrix}
$$

$$
= \text{vec}(Y_{i+1}) - \begin{bmatrix}
h_{i+1} \left( (x_i^{(1)})^H \otimes R_{i+1} \right) \text{vec}(\Phi) \\
\vdots \\
h_{i+1} \left( (x_i^{(p)})^H \otimes R_{i+1} \right) \text{vec}(\Phi)
\end{bmatrix}
$$

$$
= \text{vec}(Y_{i+1}) - g_{i+1} \begin{bmatrix}
\left( (x_i^{(1)})^H \otimes R_{i+1} \right) \\
\vdots \\
\left( (x_i^{(p)})^H \otimes R_{i+1} \right)
\end{bmatrix} \text{vec}(\Phi)
$$

(3.20)

where the new projection $g_{i+1}()$ is a $pl$-dimensional non-linear function obtained by stacking $p \ell$-dimensional $h_{i+1}()$ projections. Take the derivative of $F_N$ with respect to the vector $\text{vec}(\Phi)$, set it to $\mathbf{0}$ to obtain

$$
\sum_{i=1}^{N-1} \beta_i^{[1:N]} \begin{bmatrix}
x_i^{(1)} \otimes (J_{i+1}^{(1)} R_{i+1})^H \\
\vdots \\
x_i^{(p)} \otimes (J_{i+1}^{(p)} R_{i+1})^H
\end{bmatrix}_{m^2 \times pl} \text{vec}(E_i) = \mathbf{0} \quad (3.21)
$$

where $\mathbf{0}$ is a $m^2 \times 1$ column vector and where $J_{i+1}^{(j)}$, $1 \leq j \leq p$, denotes the $\ell \times m$ Jacobian of the $\ell$-dimensional projection function $h_{i+1}() = [h_{i+1,1}(), \ldots, h_{i+1,\ell}]^H$
evaluated at $x_{i+1}^{(j)} = R_{i+1} \Phi^{(j)}_N$. More explicitly,

$$J_{i+1}^{(j)} = \begin{bmatrix}
\frac{\partial h_{i+1,1}(\cdot)}{\partial x_1} & \cdots & \frac{\partial h_{i+1,1}(\cdot)}{\partial x_m} \\
\vdots & \ddots & \vdots \\
\frac{\partial h_{i+1,m}(\cdot)}{\partial x_1} & \cdots & \frac{\partial h_{i+1,m}(\cdot)}{\partial x_m}
\end{bmatrix}
$$

(3.22)

where $x_1, \ldots, x_m$ are the $m$ entries of the position vector $x_{i+1}^{(j)}$. To make the result in equation (3.21) more compact, the following relationships were used:

$$\begin{align*}
\left( (x_i^{(j)})^T \otimes R_{i+1} \right)^T J_{i+1}^{(j)} &= \left( x_i^{(j)} \otimes R_{i+1}^T \right) \left( 1 \otimes J_{i+1}^{(j)} \right)^T \\
&= \left( x_i^{(j)} \otimes R_{i+1} J_{i+1}^{(j)} \right) \\
&= \left( x_i^{(j)} \otimes (J_{i+1}^T R_{i+1}) \right)^T \quad (3.23)
\end{align*}$$

Linearizing the error vector $\text{vec}(E_i)$ about a given vector $\Phi_N^*$ yields

$$\begin{align*}
\text{vec}(E_i) &= \text{vec}(Y_{i+1}) - \\
&= \text{vec}(Y_{i+1}) - \\
&= \text{vec}(E_{i+1}) - \\
&= \begin{bmatrix}
\frac{\partial h_{i+1,1}(\cdot)}{\partial x_1} (x_i^{(1)})^T J_{i+1}^{(1)} R_{i+1} + J_{i+1}^{(1)} (\Phi - \Phi_N^*) x_i^{(1)} \\
\vdots \\
\frac{\partial h_{i+1,m}(\cdot)}{\partial x_1} (x_i^{(p)})^T J_{i+1}^{(p)} R_{i+1} + J_{i+1}^{(p)} (\Phi - \Phi_N^*) x_i^{(p)} \\
\end{bmatrix}
\end{align*}

(3.24)

where $J_{i+1}^{*^{(j)}}$ denotes the Jacobian evaluated at $R_{i+1} \Phi_N^{*^{(j)}}$. See Appendix A for the proof of the first order approximation. Notice that the $m^2 \times p\ell$ matrix in
equation (3.21) can be written as

\[
\begin{bmatrix}
 z_i^{(1)} \otimes (J_{i+1}^{(1)} R_{i+1})^H, \ldots, z_i^{(p)} \otimes (J_{i+1}^{(p)} R_{i+1})^H
\end{bmatrix}^H = \begin{bmatrix}
 (z_i^{(1)})^H \otimes J_{i+1}^{(1)} R_{i+1} \\
 \vdots \\
 (z_i^{(p)})^H \otimes J_{i+1}^{(p)} R_{i+1}
\end{bmatrix}^H
\]

(3.25)

The least-square solution \( \Phi_N \) to equation (3.18) is

\[
\sum_{i=1}^{N-1} \beta_i^{[1:N]} \begin{bmatrix}
 (z_i^{(1)})^H \otimes J_{i+1}^{(1)} R_{i+1} \\
 \vdots \\
 (z_i^{(p)})^H \otimes J_{i+1}^{(p)} R_{i+1}
\end{bmatrix}^H \begin{bmatrix}
 vec(\Phi_N - \Phi_N^*) \\
 vec(Y_{i+1} - Y_{i+1}^*)
\end{bmatrix}
\]

where \( Y_{i+1}^* = [h_{i+1}(R_{i+1} \Phi_N^{*z_i^{(1)}}), \ldots, h_{i+1}(R_{i+1} \Phi_N^{*z_i^{(p)}})] \) is a \( \ell \times p \) matrix. In the sequel the particular case \( \Phi_N^* = 0 \) is considered. Using the properties of the Kronecker product operator \( \otimes \) [19], the above equality can be further simplified to

\[
\sum_{i=1}^{N-1} \beta_i^{[1:N]} \sum_{j=1}^p \begin{bmatrix}
 (z_i^{(j)})^H \otimes R_{i+1}^H (J_{i+1}^{(j)})^H J_{i+1}^{*(j)} R_{i+1}
\end{bmatrix} vec(\Phi_N)
\]

(3.27)
Regroup all the terms of equation (3.27) into an equality of two vec() expressions and establish the equality between the two arguments to obtain:

\[
\sum_{i=1}^{N-1} \beta_i^{[1:N]} \sum_{j=1}^{P} R_{i+1}^H (J_{i+1}^{(j)})^H J_{i+1}^{*(j)} R_{i+1} \Phi_N \bar{x}_i^{(j)} (\bar{x}_i^{(j)})^H \\
= \sum_{i=1}^{N-1} \beta_i^{[1:N]} \sum_{j=1}^{P} R_{i+1}^H (J_{i+1}^{(j)})^H y_{i+1}^{(j)} (\bar{x}_i^{(j)})^H
\]

(3.28)

For \( N = 2 \) and \( p = 1 \), the solution \( \Phi_N \) is in general not unique since the rank of the Jacobians does not exceed \( \ell \).

### 3.1.2 PERSPECTIVE PROJECTION

In this section, some of the quantities that appeared in the preceding section are explicitly computed. With help of the definitions in section (2.3), the Jacobian at \( \bar{x}_{i+1} = R_{i+1} \Phi_N \bar{x}_i^{(j)} \) can be found to be

\[
J_{i+1}^{(j)} = \begin{bmatrix}
\frac{F_0}{\bar{x}_{i+1,3} + F_0} & 0 & -\bar{y}_{i+1,1} F_0 \\ 0 & \frac{F_0}{\bar{x}_{i+1,3} + F_0} & -\bar{y}_{i+1,2} F_0 \\
\frac{F_0}{\bar{x}_{i+1,3} + F_0} & 0 & -\bar{y}_{i+1,2} F_0
\end{bmatrix}
\]

(3.29)

where \( \bar{y}_{i+1} = h(\bar{x}_{i+1}) \) and where \( \bar{x}_{i+1} = [\bar{x}_{i+1,1}, \bar{x}_{i+1,2}, \bar{x}_{i+1,2}]^H \). The Jacobian \( J_{i+1}^{*(j)} \) takes a similar expression at \( R_{i+1} \Phi_N^* \bar{x}_i^{(j)} \).
3.1.3 LINEAR PROJECTION

When the projection is linear, equation (3.28) can be simplified as follows. Let \( h_k() \) be the linear projection onto the \( \ell \)-dimensional, \( \ell \leq m \), subspace spanned by the vectors \( \{v_1, \ldots, v_\ell\} \). Denote by \( B_k \) the \( m \times \ell \) matrix \([v_1, \ldots, v_\ell]\). The projection \( h_k() \) is represented by the \( m \times m \) matrix \( B_k B_k^H \) whose rank is at most \( \ell \). If in addition the vectors \( v_1, \ldots, v_\ell \) are linearly independent and mutually orthogonal so that \( \text{rank}(B_k B_k^H) = \ell \) and \( B_k^H B_k = I_{\ell \times \ell} \), equation (3.28) becomes

\[
\sum_{i=1}^{N-1} \beta_i^{[1:N]} R_{i+1}^H B_{i+1} B_{i+1}^H R_{i+1} \Phi_N \sum_{i=1}^{p} y_i^{(j)} (x_i^{(j)})^H = \sum_{i=1}^{N-1} \beta_i^{[1:N]} R_{i+1}^H B_{i+1} B_{i+1}^H \sum_{j=1}^{p} y_{i+1}^{(j)} (x_i^{(j)})^H
\]

(3.30)

which results from replacing the position dependent Jacobians \( J_{i+1}^{[j]} \) and \( J_{i+1}^{*[j]} \) by the position independent projection symmetric operator \( B_{i+1} B_{i+1}^H \). This loss of position dependency suggests that non-linear projections contain more information about the underlying motion of the object than linear projections.

The result shown in equation (3.30) is general and, in particular, \( \Phi_N \) does not need to be close to \( \Phi_N^* = 0 \). Equation (3.30) can be further simplified when both the camera motion and the projection \( h() \) are time invariant:

\[
R^H B B^H R \Phi_N \sum_{i=1}^{N-1} \beta_i^{[1:N]} \sum_{i=1}^{p} y_i^{(j)} (x_i^{(j)})^H = R^H B B^H \sum_{i=1}^{N-1} \beta_i^{[1:N]} \sum_{j=1}^{p} y_{i+1}^{(j)} (x_i^{(j)})^H
\]

(3.31)

The results shown in equation (3.28), (3.30) and (3.31) are important as they represent the "batch" solution of a recursive algorithm. As such, they provide the basis for studying the identifiability of \( \Phi_N \) in the framework of least-square minimization. The problem of identifiability is to be addressed in an oncoming section.
3.2 TIME-VARYING MODEL

3.2.1 NON-LINEAR PROJECTION

The motion of objects is seldom time-invariant and therefore time dependency must be introduced into the motion estimator. This is done by computing the motion parameters recursively. In other words, minimize $\mathcal{F}_{N+1}(\Phi)$ with the assumption that $\mathcal{F}_N(\Phi)$ has already been minimized. Denote by $\Phi_{N+1}$ and $\Phi_N$ the solutions to minimizing $\mathcal{F}_{N+1}(\Phi)$ and $\mathcal{F}_N(\Phi)$, respectively. Assume for the time being that the object motion follows the time invariant model $x_{i+1} = \Phi x_i$ so that

$$\Phi_N + \Delta \Phi_N = \Phi_{N+1} + \Delta \Phi_{N+1} = \Phi$$ \hspace{1cm} (3.32)

Assuming that $\|\text{vec}(\Delta \Phi)\|^3$ is small and can be neglected, the second order Taylor series expansion of $\mathcal{F}_{N+1}(\Phi)$ yields [114]:

$$\mathcal{F}_{N+1}(\Phi) = \mathcal{F}_{N+1}(\Phi_{N+1}) + D_{\text{vec}(\Phi)^H} \{\mathcal{F}_{N+1}(\Phi = \Phi_{N+1})\} \text{vec}(\Delta \Phi_{N+1})$$

$$+ \frac{1}{2} \text{vec}(\Delta \Phi_{N+1})^H D_{\text{vec}(\Phi)\text{vec}(\Phi)^H} \{\mathcal{F}_{N+1}(\Phi = \Phi_{N+1})\} \text{vec}(\Delta \Phi_{N+1})$$

$$\underbrace{= 0^H}_{\Gamma_{N+1}}$$

$$= \mathcal{F}_{N+1}(\Phi_{N+1}) + \frac{1}{2} \text{vec}(\Phi_N - \Phi_{N+1} + \Delta \Phi_N)^H \Gamma_{N+1} \text{vec}(\Phi_N - \Phi_{N+1} + \Delta \Phi_N)$$ \hspace{1cm} (3.33)

where $D_{\text{vec}(\Phi)^H}$ and $D_{\text{vec}(\Phi)\text{vec}(\Phi)^H}$ denote the derivative of $\mathcal{F}_{N+1}(\Phi)$ with respect to the vector $\text{vec}(\Phi)$ and the matrix $\text{vec}(\Phi)\text{vec}(\Phi)^H$, respectively, and where $\Gamma_{N+1}$ is a $m^2 \times m^2$ symmetric matrix. The fact that the first order term is equal to $0^H$ follows from the minimization of $\mathcal{F}_{N+1}(\Phi)$ at $\Phi_{N+1}$. 
Alternatively, it can be written

\[
\mathcal{F}_{N+1}(\Phi) = 
\sum_{i=1}^{N} \beta_{i}^{[1:N+1]} \sum_{j=1}^{p} (y_{i+1}^{(j)} - h_{i+1} \left( R_{i+1} \Phi x_{i}^{(j)} \right))^{H} \left( y_{i+1}^{(j)} - h_{i+1} \left( R_{i+1} \Phi x_{i}^{(j)} \right) \right)
\]

(3.34)

To control the bandwidth of the time-varying model, use a stochastic Newton approximation method [69]. Such scheme is well suited for tracking time-varying systems as it includes data forgetting factors. Given the sequence \( \{\beta_{1}^{[1:2]}, \beta_{2}^{[1:3]}, \ldots, \beta_{k}^{[1:k+1]} \} \), the forgetting profile is governed by another sequence \( \{\lambda^{[1:2]}, \lambda^{[1:3]}, \ldots, \lambda^{[1:k]} \} \) such that

\[
\beta_{i}^{[1:k+1]} = \lambda^{[1:k]} \beta_{i}^{[1:k]}
\]

\[
= \left( \prod_{j=i+1}^{k} \lambda^{[1:j]} \right) \beta_{i}^{[1:i+1]} \quad \text{for} \quad 1 \leq i \leq k - 1
\]

(3.35)

The recursion in equation (3.35) can be related to the gain sequence \( \{\gamma^{[1:2]}, \ldots, \gamma^{[1:k+1]} \} \) of a conventional Robbins-Monro scheme by setting \( \beta_{i}^{[1:i+1]} = 1 \) and \( \lambda^{[1:i]} = \gamma^{[1:k]}(1 - \gamma^{[1:i+1]})/\gamma^{[1:i+1]} \). Equation (3.32) can now be re-written as follows:

\[
\mathcal{F}_{N+1}(\Phi) = \lambda^{[1:N]} \mathcal{F}_{N}(\Phi)
\]

\[
+ \beta_{N}^{[1:N+1]} \sum_{j=1}^{p} (y_{N+1}^{(j)} - h_{N+1} \left( R_{N+1} \Phi x_{N}^{(j)} \right))^{H} \left( y_{N+1}^{(j)} - h_{N+1} \left( R_{N+1} \Phi x_{N}^{(j)} \right) \right)
\]

(3.36)

or using equation (3.32),

\[
\mathcal{F}_{N+1}(\Phi) = \lambda^{[1:N]} \mathcal{F}_{N}(\Phi) + \lambda^{[1:N]} \frac{1}{2} vec(\Delta \Phi N)^{H} \Gamma_{N} vec(\Delta \Phi N)
\]

\[
+ \beta_{N}^{[1:N+1]} \sum_{j=1}^{p} (y_{N+1}^{(j)} - h_{N+1} \left( R_{N+1} \Phi x_{N}^{(j)} \right))^{H} \left( y_{N+1}^{(j)} - h_{N+1} \left( R_{N+1} \Phi x_{N}^{(j)} \right) \right)
\]

(3.37)
Let \( \hat{x}_{N+1}^{(j)} = R_{N+1} \Phi_N x_N^{(j)} \). This term can be interpreted as a prediction of object position \( \hat{x}_{N+1}^{(j)} \) based on the motion model at time \( N \). In Appendix A, it is shown that for a second order approximation, the following equality holds:

\[
\begin{align*}
\hat{x}_{N+1}^{(j)} - h_{N+1} \left( R_{N+1} \Phi_N x_N^{(j)} \right) &= y_{N+1}^{(j)} - h_{N+1} \left( R_{N+1} (\Phi_N + \Delta \Phi_N) x_N^{(j)} \right) \\
&= y_{N+1}^{(j)} - h (\hat{x}_N^{(j)}) - J_{N+1}^{(j)} R_{N+1} \Delta \Phi_N x_N^{(j)} \\
&\quad - \frac{1}{2} \left( R_{N+1} \Delta \Phi_N x_N^{(j)} \otimes I_{\ell \times \ell} \right) H_{N+1}^{(j)} R_{N+1} \Delta \Phi_N x_N^{(j)} 
\end{align*}
\]

(3.38)

where \( I_{\ell \times \ell} \) is the \( \ell \times \ell \) identity matrix and \( J_{N+1}^{(j)} \) is the Jacobian of \( h_{N+1}() \) evaluated at \( \hat{x}_{N+1}^{(j)} \). The matrix \( H_{N+1}^{(j)} \) is the \( m \ell \times m \) re-shuffled Hessian matrix

\[
H_{N+1}^{(j)} = \begin{bmatrix}
\frac{\partial^2 h_{N+1,1}}{\partial x_1 x_1} (\hat{x}_N^{(j)}) & \cdots & \frac{\partial^2 h_{N+1,1}}{\partial x_1 x_m} (\hat{x}_N^{(j)}) \\
\vdots & \ddots & \vdots \\
\frac{\partial^2 h_{N+1,\ell}}{\partial x_1 x_1} (\hat{x}_N^{(j)}) & \cdots & \frac{\partial^2 h_{N+1,\ell}}{\partial x_1 x_m} (\hat{x}_N^{(j)}) \\
\frac{\partial^2 h_{N+1,1}}{\partial x_m x_1} (\hat{x}_N^{(j)}) & \cdots & \frac{\partial^2 h_{N+1,1}}{\partial x_m x_m} (\hat{x}_N^{(j)}) \\
\vdots & \ddots & \vdots \\
\frac{\partial^2 h_{N+1,\ell}}{\partial x_m x_1} (\hat{x}_N^{(j)}) & \cdots & \frac{\partial^2 h_{N+1,\ell}}{\partial x_m x_m} (\hat{x}_N^{(j)})
\end{bmatrix}
\]

(3.39)

where \( h_{N+1}() = [h_{N+1,1}(), \ldots, h_{N+1,\ell}()]^H \). Equations (3.38) and (3.39) yield
\[ \| y_{N+1}^{(j)} - h_{N+1}^{(j)} (R_{N+1} \Phi_N^{(j)}) \|^2 \]
\[ = \left( y_{N+1}^{(j)} - h_{N+1}^{(j)} (R_{N+1} \Phi_N^{(j)}) \right)^H \left( y_{N+1}^{(j)} - h_{N+1}^{(j)} (R_{N+1} \Phi_N^{(j)}) \right) \]
\[ = (\varepsilon_{N+1}^{(j)})^H (\varepsilon_{N+1}^{(j)})^H 2(\varepsilon_{N+1}^{(j)})^H \left( (\varepsilon_{N}^{(j)})^H \otimes J_{N+1}^{(j)} R_{N+1}^{(j)} \right) vec(\Delta \Phi_N) + \]
\[ \text{vec}(\Delta \Phi_N)^H (\varepsilon_{N}^{(j)})^H \otimes R_{N+1}^H \left( (J_{N+1}^{(j)})^H J_{N+1}^{(j)} - \mathcal{H}_{N+1}^{(j)} \right) R_{N+1} vec(\Delta \Phi_N) \]

(3.40)

where

\[ \varepsilon_{N+1}^{(j)} = y_{N+1}^{(j)} - h_{N+1}^{(j)} (\varepsilon_{N+1}^{(j)}) \]
\[ \mathcal{H}_{N+1}^{(j)} = \left( I_{m \times m} \otimes \left( \varepsilon_{N+1}^{(j)} \right)^H \right) H_{N+1}^{(j)} \]

(3.41)

are an innovation vector and an \( m \times m \) matrix derived from the \( m \ell \times m \) reshuffled Hessian matrix \( H_{N+1}^{(j)} \), respectively. See Appendix B for a complete derivation. Substituting equation (3.40) into (3.37) and equating the latter with the expression in equation (3.33), the following recursive update equation is obtained

\[ \text{vec}(\Phi_{N+1}) = \text{vec}(\Phi_N) + 2\beta_{[1:N+1]}^{[1:N]} P_{N+1} \text{vec} \left( \sum_{j=1}^{p} R_{N+1}^H (J_{N+1}^{(j)})^H \varepsilon_{N+1}^{(j)} (\varepsilon_{N+1}^{(j)})^H \right) \]

(3.42)

where \( P_{N+1} = \Gamma_{N+1}^{-1} \) is a \( m^2 \times m^2 \) matrix. This equation is similar to the update equation found in a standard recursive least-square algorithm. The second update equation is

\[ P_{N+1} = \frac{1}{\lambda [1:N]} \left\{ I_{m^2 \times m^2} + \right\} \]
\[
\frac{2\beta_{N+1}}{\lambda_{N+1}} P_N \sum_{j=1}^{P} \left( \begin{pmatrix} x_N^{(j)} \\ x_N^{(j)} \end{pmatrix}^H \otimes R_{N+1}^H \left( (J_{N+1}^{(j)})^H J_{N+1}^{(j)} - \mathcal{H}_{N+1}^{(j)} \right) R_{N+1} \right)^{-1} P_N \right) 
\] 

\[
(3.43)
\]

This equation is similar to the covariance update equation obtained in a conventional Recursive Least-Square algorithm before the inversion lemma is applied. It is a Riccati equation. The above equations define the Extended Recursive Least-Squares algorithm. For a complete derivation of the results in equations (3.41) and (3.42), refer to Appendix C. In equation (3.42), the matrix \( P_1 \) must be initialized to \( \delta^{-1} I_{m^2 \times m^2} \) where \( \delta < 1 \) is a small value and \( I_{m^2 \times m^2} \) is the \( m^2 \times m^2 \) identity matrix [47].

### 3.2.2 PERSPECTIVE PROJECTION

The results obtained in the previous subsection are illustrated by calculating some of the quantities used in the iterations for the particular case of perspective projection. The Jacobian at \( \tilde{x}^{(j)}_{N+1} = R_{N+1} \Phi_N \bar{z}^{(j)}_N \) is

\[
J_{N+1}^{(j)} = \begin{bmatrix}
\frac{F_0}{\tilde{x}_{N+1,3} + F_0} & 0 & \frac{-\tilde{x}^{(j)}_{N+1,1} F_0}{\tilde{x}_{N+1,3} + F_0} \\
0 & \frac{F_0}{\tilde{x}_{N+1,2} + F_0} & \frac{-\tilde{x}^{(j)}_{N+1,2} F_0}{\tilde{x}_{N+1,2} + F_0}
\end{bmatrix}
\]

\[
= \frac{1}{\tilde{x}_{N+1,3} + F_0} \begin{bmatrix}
F_0 & 0 & \tilde{y}^{(j)}_{N+1,1} \\
0 & F_0 & \tilde{y}^{(j)}_{N+1,2}
\end{bmatrix}
\]

\[
(3.44)
\]
where \( \tilde{z}^{(j)}_{N+1} = h(\tilde{z}^{(j)}_{N+1}) = [\tilde{y}^{(j)}_{N+1,1}, \tilde{y}^{(j)}_{N+1,2}]^H \) and \( \tilde{z}^{(j)}_{N+1} = [\tilde{x}^{(j)}_{N+1,1}, \tilde{x}^{(j)}_{N+1,2}, \tilde{x}^{(j)}_{N+1,3}]^H \). The re-shuffled 6 x 3 Hessian matrix at \( \tilde{z}^{(j)}_{N+1} \) is

\[
H^{(j)}_{N+1} = \frac{1}{(\tilde{x}^{(j)}_{N+1,3} + F_0)^2} \begin{bmatrix}
0 & 0 & -F_0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & -F_0 \\
-\epsilon^{(j)}_{N+1,1} & 0 & 2\tilde{y}^{(j)}_{N+1,1} \\
0 & -F_0 & 2\tilde{y}^{(j)}_{N+1,2}
\end{bmatrix}
\]

and the matrix \( \mathcal{J}^{(j)}_{N+1} \) is the 3 x 3 symmetric matrix

\[
\mathcal{J}^{(j)}_{N+1} = \frac{1}{(\tilde{x}^{(j)}_{N+1,3} + F_0)^2} \begin{bmatrix}
0 & 0 & -\epsilon^{(j)}_{N+1,1}F_0 \\
0 & 0 & -\epsilon^{(j)}_{N+1,2}F_0 \\
-\epsilon^{(j)}_{N+1,1}F_0 & -\epsilon^{(j)}_{N+1,2}F_0 & 2(\epsilon^{(j)}_{N+1})^H \tilde{y}^{(j)}_{N+1}
\end{bmatrix}
\]

where the innovation \( \epsilon^{(j)}_{N+1} = y^{(j)}_{N+1} - \tilde{y}^{(j)}_{N+1} = y^{(j)}_{N+1} - h(\tilde{z}^{(j)}_{N+1}) \) is the two dimensional column vector \( [\epsilon^{(j)}_{N+1,1}, \epsilon^{(j)}_{N+1,2}]^H \).

### 3.2.3 LINEAR PROJECTION

The general results are now re-written for the specific case when the projections \( h_k(), k = 1, \ldots, N + 1 \), are linear. Equations (3.42) and (3.43) simplify as
follows. Equation (3.42) becomes

$$vec(\Phi_{N+1}) = vec(\Phi_N) + 2\beta_N^{[1:N+1]}P_{N+1}vec \left( R_{N+1}^H B_{N+1} B_{N+1}^H \sum_{j=1}^P \varepsilon_{N+1}^{(j)}(x_N^{(j)})^H \right)$$

(3.47)

which is obtained by replacing the position dependent Jacobians by the position independent symmetric matrix $B_{N+1}B_{N+1}^H$. Likewise, equation (3.43) becomes

$$P_{N+1} = \frac{1}{\lambda_{[1:N]}} \left\{ I_{m^2 \times m^2} + \frac{2\beta_N^{[1:N+1]}}{\lambda_{[1:N]}} P_N \sum_{j=1}^P \left( x_N^{(j)}(x_N^{(j)})^H \otimes R_{N+1}^H B_{N+1} B_{N+1}^H R_{N+1} \right) \right\}^{-1} P_N$$

(3.48)

since the Hessian $H_{N+1}^{(j)}$ is the $m \ell \times m$ 0 matrix. If the $m^2 \times m^2$ matrix $P_1$ is initialized as a $m \times m$ identical blocks diagonal matrix, equations (3.43) and (3.44) reduce to a conventional Recursive Least Square algorithm, which involves only $m \times m$ matrices.

### 3.3 IDENTIFIABILITY

Now that estimates of the transition matrix have been derived, conditions for the existence of a unique solution must be found in both the non-linear and linear case. In order to dissociate the problem of estimating the unknown $m$-dimensional feature positions from the problem of estimating the motion parameters, it is assumed that the $x_i^{(j)}$s, $1 \leq i \leq N$, $1 \leq j \leq p$, are known. Identifiability in the framework of least-square estimation is considered first. A more general identifiability criterion is then proposed for the case of perspective projection.
3.3.1 LEAST-SQUARE FRAMEWORK

In this first subsection, identifiability is studied in the framework of the least-square estimation schemes presented in sections 3.1 and 3.2. From equation (3.27), it can be seen that $\Phi_N$ will be a unique solution when

$$\text{rank} \left\{ \sum_{i=1}^{N-1} \beta_i^{[1:N]} \sum_{j=1}^{p} (q_i^{(j)}(q_i^{(j)}))^H \otimes R_i^{H}(J_{i+1}^{(j)}H J_{i+1}^{*}(j)R_{i+1}) \right\} = m^2$$

(3.49)

where $\text{rank}\{M\}$ denotes the rank of a matrix $M$. This comes from the fact that the $m \times m$ matrix $\Phi_N$ features $m^2$ unknown parameters. To find out the number of image frames and feature correspondence required to satisfy this condition, recall that the rank of a matrix resulting from the Kronecker product of two matrices cannot be greater than the product of the individual matrix ranks [19]. Moreover, in equation (3.49),

$$\text{rank} \left\{ q_i^{(j)}(q_i^{(j)})^H \right\} = 1$$

$$\text{rank} \left\{ R_i^{H}(J_{i+1}^{(j)}H J_{i+1}^{*}(j)R_{i+1}) \right\} = \ell$$

(3.50)

It follows that the number of images frames $N$ combined with the number of features $p$ must be such that $(N - 1)p\ell \geq m^2$. The results hold for the linear case as well. However, the loss of position dependency in the Jacobians means that in some situations, more positions and frames are needed. To realize this, consider equation (3.30) and assume that two vectors $q_i^j$ and $q_i^k$ are such that $q_i^j = s q_i^k$ where $s$ is a scalar, $s \neq 0$. Clearly, the contribution made by $q_i^j$ to identify $\Phi_N$ is the same as the one made by $q_i^k$ since the factor $s^2$ appears on both sides of the equality and cancels out. This cancellation will not happen for a non-linear projection in general because the Jacobians at $q_i^j$ and $q_i^k$ are different. Finally, note that successive image frames cannot be used to identify
the matrix $\Phi_N$ when the motion is time-varying. In this case, the number of corresponding features $p$ must be chosen such that $p \ell \geq m^2$.

### 3.3.2 GENERAL FRAMEWORK

The identifiability of $\Phi_N$ is now studied in the more general framework of non-linear projection. However, due to the lack of general results about identifiability in non-linear systems, only perspective projection is considered. The question at hand is what is the number of corresponding features required to identify the transition matrices. Again, both the projected positions $y^{(j)}_i$ and the original positions $x^{(j)}_i$ are assumed to be known. Consider $Np$ projected positions of corresponding features $y^{(j)}_i = [y^{(j)}_{i,1}, y^{(j)}_{i,2}]^T$, $1 \leq j \leq p$, $1 \leq i \leq N$, with their respective original positions $x^{(j)}_i = [x^{(j)}_{i,1}, x^{(j)}_{i,2}, x^{(j)}_{i,3}]^T$. Assume that the motion follows the model $y^{(j)}_{k+1} = h(\Phi_{k+1} x^{(j)}_k)$, where $h()$ here represents the perspective projection operator. With the help of equation (2.15), one can write

\[
\begin{bmatrix}
  y^{(j)}_{i+1,1} \\
  y^{(j)}_{i+1,2}
\end{bmatrix} = \begin{bmatrix}
  \frac{x^{(j)}_{i+1,1}F_0}{x^{(j)}_{i+1,3} + F_0} \\
  \frac{x^{(j)}_{i+1,2}F_0}{x^{(j)}_{i+1,3} + F_0}
\end{bmatrix} = \begin{bmatrix}
  \frac{\phi_{1,1}x^{(j)}_{i,1} + \phi_{1,2}x^{(j)}_{i,2} + \phi_{1,3}x^{(j)}_{i,3}}{F_0} \\
  \frac{\phi_{3,1}x^{(j)}_{i,1} + \phi_{3,2}x^{(j)}_{i,2} + \phi_{3,3}x^{(j)}_{i,3} + F_0}{F_0}
\end{bmatrix}
\]

(3.51)

where $\phi_{ij}$ denote the elements of the matrix $\Phi_{i+1}$. It can readily be established that equation (3.51) can be reformulated as

\[
y^{(j)}_{i+1} = \frac{x_{i,3} + F_0}{F_0}
\]

\[
\begin{bmatrix}
  y^{(j)}_{i,1} & 0 & -\frac{y^{(j)}_{i+1,1}v^{y^{(j)}_{i,1}}}{F_0} & y^{(j)}_{i,2} & 0 & -\frac{y^{(j)}_{i+1,2}v^{y^{(j)}_{i,2}}}{F_0} & 0 & -\frac{y^{(j)}_{i+1,3}v^{y^{(j)}_{i,3}}}{F_0} & 0 & -\frac{y^{(j)}_{i+1,4}v^{y^{(j)}_{i,4}}}{F_0}
\end{bmatrix} \cdot vec(\Phi_{i+1})
\]

(3.52)
where $\mathbf{z}_i^{(j)} = \mathbf{x}_i^{(j)} \mathbf{F}_0 / (\mathbf{x}_i^{(j)} + \mathbf{F}_0)$. The $2 \times 9$ matrix shown above will be in general of rank two. To uniquely identify the nine motion parameters in the transition matrix $\Phi_{i+1}$, a total of at least five corresponding features are needed. If the motion is time invariant, these feature positions can come from the same point tracked over six successive image frames. If the motion is time-varying, they must come from five different features in the same image frame.

### 3.4 OBSERVABILITY

In section 3.3, it was assumed that the positions of the features were known. It seems natural at this point to look at the complementary problem which is concerned with the reconstruction of the feature positions from their projections assuming that the object motion is known. This is an observability problem. An approach similar to the one used for studying identifiability is followed. First, observability is studied in the framework of the least-square solution. The problem of observability is then studied in the particular setting of perspective projection. Finally, an alternative method, depth from zooming, is proposed.

#### 3.4.1 LEAST-SQUARE FRAMEWORK

The least-square solution $\Phi_N$ obtained in equation (3.28) suggests that observability of a given feature position $\mathbf{x}_i^{(j)}$ can be established with the help of the following equality:

$$
\mathbf{y}_i^{(j)} = \mathbf{J}_{i+1}^{(j)} \mathbf{R}_{i+1} \Phi_N \mathbf{x}_i^{(j)}
$$

which is valid for small motions only since the linearization was performed at $\Phi_N = \mathbf{0}$. For the sake of simplicity, assume that the rotation matrices $\mathbf{R}_{i+1}$, $i = 1, \ldots, N$, are all equal to a constant matrix $\mathbf{R}_N$. Since the motion model is
time invariant from frame $i = 1$ to frame $i = N$, past records can be combined to observe the feature position $\mathbf{z}_{N-m-1}^{(j)}$ as shown below:

\[
\begin{bmatrix}
\mathbf{y}_{N-m-1}^{(j)} \\
\mathbf{y}_{N-m}^{(j)} \\
\mathbf{y}_{N-m+1}^{(j)} \\
\vdots \\
\mathbf{y}_N^{(j)}
\end{bmatrix}
= 
\begin{bmatrix}
\mathbf{J}_{N-m-1}^{*(j)} & \mathbf{I}_{m \times m} \mathbf{J}_{N-m}^{*(j)} \\
\mathbf{J}_{N-m}^{*(j)} & (\mathbf{R}_N \Phi_N)^2 \\
\vdots & \vdots \\
\mathbf{J}_N^{*(j)} & (\mathbf{R}_N \Phi_N)^{m-1}
\end{bmatrix}
\mathbf{z}_{N-m-1}^{(j)}
\tag{3.54}
\]

This equality takes a form similar to the one used for observability in linear systems [20]. The position $\mathbf{z}_{N-m-1}^{(j)}$ will be observable if the matrix shown in the right member of equation (3.54) is at least of rank $m$. Powers of $\mathbf{R}_N \Phi_N$ greater than $m - 1$ are not considered as they are linear combinations of the first $m$ powers (0 to $m - 1$). This fact is a consequence of the Cayley-Hamilton theorem.

3.4.2 GENERAL FRAMEWORK

Just as in the case of identifiability, the results are now extended to a more general framework where the transition matrix no longer results from a least-square minimization. Because of the lack of general results about observability in non-linear systems, the study of observability is restricted to the case of perspective projection. Consider again equation (3.51). Divide both numerators and denominators by $\mathbf{x}_{1,3}^{(j)} + F_0$ which is assumed positive (object in front of the
camera or the eye) to obtain

\[
\begin{bmatrix}
  y_{i+1,1}^{(j)} \\
  y_{i+1,2}^{(j)}
\end{bmatrix} = \begin{bmatrix}
  \frac{F_0 (\phi_{1,1} y_{i,1}^{(j)} + \phi_{1,2} y_{i,2}^{(j)} + \phi_{1,3} z_i^{(j)})}{\phi_{3,1} y_{i,1}^{(j)} + \phi_{3,2} y_{i,2}^{(j)} + \phi_{3,3} z_i^{(j)} + \frac{F_0}{x_{i,3} + F_0}} \\
  \frac{F_0 (\phi_{2,1} y_{i,1}^{(j)} + \phi_{2,2} y_{i,2}^{(j)} + \phi_{2,3} z_i^{(j)})}{\phi_{3,1} y_{i,1}^{(j)} + \phi_{3,2} y_{i,2}^{(j)} + \phi_{3,3} z_i^{(j)} + \frac{F_0}{x_{i,3} + F_0}}
\end{bmatrix}
\]

This result can be used in different ways. If \( y_{i+1,1}^{(j)} \neq 0 \), the ratio of the two observed coordinates is

\[
\frac{y_{i+1,2}^{(j)}}{y_{i+1,1}^{(j)}} = \frac{\phi_{2,1} y_{i,1}^{(j)} + \phi_{2,2} y_{i,2}^{(j)} + \phi_{2,3} z_i^{(j)}}{\phi_{1,1} y_{i,1}^{(j)} + \phi_{1,2} y_{i,2}^{(j)} + \phi_{1,3} z_i^{(j)}}
\]

(3.56)

where \( z_i^{(j)} = x_{i,3} F_0 / (x_{i,3} + F_0) \) is the only unknown. Hence, the depth \( x_{i,3} \) can be calculated and the remaining coordinates \( x_{i,1} \) and \( x_{i,2} \) are readily obtained from equation (2.15). Notice that the observability cannot be established when \( \phi_{2,3} = 0 \) and \( \phi_{1,3} = 0 \) or when \( y_{i+1,1}^{(j)} = 0 \). This last case corresponds to the fact that the perspective projection behaves like an orthogonal projection at the center of the screen. Another way to use equation (3.55) is to assume that \( z_i^{(j)} \) tends to \( F_0 \). This happens when the feature depth value \( x_{i,3}^{(j)} \) is very large compared to the focal distance \( F_0 \). In this case, \( z_i^{(j)} \) can be replaced by \( F_0 \) in equation (3.55) leaving two expressions where \( x_{i,3}^{(j)} \) is the only unknown. This kind of observability can be viewed as observability at the horizon or observability at large.

### 3.4.3 DEPTH FROM ZOOMING

The goal of this section is to suggest that in particular situations, other approaches can be used to study observability. The example considered here deals with studying observability from known variations of the camera focal lens.
To simplify the expressions, it is assumed that the transition matrix $\Phi$ is the identity matrix $I_{m \times m}$. In the image frame at time $i$, the feature position is

$$
\begin{bmatrix}
y_{i,1} \\
y_{i,2}
\end{bmatrix} = \begin{bmatrix}
x_{i,1}F_0 \\
x_{i,2}F_0
\end{bmatrix}
$$

(3.57)

Assume that at time $i + 1$, the focal distance has changed from $F_0$ to $F_0 + \delta F_0$. The projected positions become

$$
\begin{bmatrix}
y_{i+1,1} \\
y_{i+1,2}
\end{bmatrix} = \begin{bmatrix}
x_{i,1}(F_0 + \delta F_0) \\
x_{i,2}(F_0 + \delta F_0)
\end{bmatrix}
$$

(3.58)

If the variation $\delta F_0$ is small, a first order approximation yields:

$$
\begin{bmatrix}
y_{i+1,1} - y_{i,1} \\
y_{i+1,2} - y_{i,2}
\end{bmatrix} = \frac{\delta F_0}{F_0} \left( \frac{x_{i,3}}{x_{i,3} + F_0} \right) \begin{bmatrix}
y_{i,1} \\
y_{i,2}
\end{bmatrix}
$$

(3.59)

The only unknown in equation (3.59) is $x_{i,3}$. Once this value is determined, $x_{i,1}$ and $x_{i,2}$ can be determined from equation (3.57).

### 3.5 CONSTRAINTS

The recursive algorithm obtained in sections 3.1 and 3.2 for estimating the transition matrix assumes that all of its $m^2$ parameters are independent. This assumption was also used in the identifiability study of the transition matrix. In some situations, it is desirable to make the transition matrices belong to a particular class of motion. This can be accomplished by projecting the solution $\Phi_k$ onto the desired class and leaving to the translation vector the additional responsibility of modeling the dynamics that do not fit this class. From the review provided in Chapter 1, it can be seen that the most popular class of transition matrices is the class of orthogonal matrices. First, a review of the
most common technique for projecting a matrix onto this class is provided. It is called polar decomposition. Next, a new approach is proposed. This new approach offers the advantage of producing estimates of the axis of rotation and rotation angle. To conclude, the particular case of small rotations is considered.

3.5.1 POLAR DECOMPOSITION

The polar decomposition of a matrix calls for decomposing the matrix into the product of an orthogonal and a positive semidefinite matrix. The polar decomposition of a nonsingular $m \times m$ matrix $\Phi_k$ is

$$
\Phi_k = \Phi_k \left( \Phi_k^H \Phi_k \right)^{-1/2} \left( \Phi_k^H \Phi_k \right)^{1/2}
$$

(3.60)

It can be verified that $QQ^H = Q^HQ = I_{m \times m}$ and $S = S^H$. The matrix $Q$ is the closest orthogonal matrix to $\Phi_k$ in the sense that

$$
Q = \min_M \left\{ \text{trace} \left\{ (M - \Phi_k)^H (M - \Phi_k) \right\} \right\}
$$

(3.61)

See [52] and for a more general form of the polar decomposition known as the “Orthogonal Procrustes Problem”, see [97]. Briefly, the proof of the result shown in equation (3.60) relies on the singular value decomposition of the positive matrices $\Phi_k \Phi_k^H$ and $\Phi_k^H \Phi_k$:

$$
\Phi_k \Phi_k^H = U_k \Lambda_k U_k^H
$$

$$
\Phi_k^H \Phi_k = V_k \Lambda_k V_k^H
$$

(3.62)

Both $U_k$ and $V_k$ are $m \times m$ orthogonal matrices and $\Lambda_k$ is a $m \times m$ diagonal matrix. It can be proved that $Q = U_k V_k^H$ [41]. This result or the result shown in equation (3.60) suggests that the computation of $Q$ requires either a singular value decomposition or an inversion of the square root of a matrix. A somewhat simpler algorithm is proposed in the next section.
3.5.2 SKEW MATRIX DECOMPOSITION

This section provides an alternative to the polar decomposition technique. The main advantage of this new decomposition is the fact that it leads to a direct determination of the rotation axis and rotation angle. Consider the motion model

\[ \mathbf{x}_{i+1} = \mathbf{R}_{i+1} \mathbf{x}_i \]  

(3.63)

where \( \mathbf{R}_{i+1} \) is a \( m \times m \) rotation matrix. It can be shown that provided none of the eigenvalue of \( \mathbf{R}_{i+1} \) is equal to \(-1\) (a rotation of \( \pi \)), one can write

\[ \mathbf{R}_{i+1} = (\mathbf{I}_{m \times m} - \mathbf{S})^{-1} (\mathbf{I}_{m \times m} + \mathbf{S}) \]  

(3.64)

where \( \mathbf{S} \) is a \( m \times m \) skew symmetric matrix. Hence \( \mathbf{S}^H = -\mathbf{S} \) or, equivalently, \( \mathbf{S} = \mathbf{F} - \mathbf{F}^H \) for some \( m \times m \) matrix \( \mathbf{F} \) (the choice of \( \mathbf{F} \) is not unique). Equation (3.64) is known as the Cayley formula [18]. This formula is now used in the problem of finding the skew matrix \( \mathbf{S} \) generating the rotation matrix closest to the transition matrix \( \mathbf{\Phi}_{i+1} \). The minimization is performed on the trace of the matrix \( \mathbf{E}^H \mathbf{E} \) where

\[ (\mathbf{I}_{m \times m} - \mathbf{S}) \mathbf{\Phi}_{i+1} = \mathbf{I}_{m \times m} + \mathbf{S} + \mathbf{E} \]  

(3.65)

or, since \( \mathbf{S} \) is a skew symmetric matrix,

\[ (\mathbf{I}_{m \times m} - \mathbf{F} + \mathbf{F}^H) \mathbf{\Phi}_{i+1} = \mathbf{I}_{m \times m} + \mathbf{F} - \mathbf{F}^H + \mathbf{E} \]  

(3.66)

A solution for \( \mathbf{S} \) is obtained by using equation (3.66) and carrying out the minimization of trace \( \{\mathbf{E}^H \mathbf{E}\} \) with respect to \( \mathbf{F} \). Using the rules of derivations provided in Appendix D, the following result is obtained

\[
\begin{align*}
\left[ (\mathbf{F} - \mathbf{F}^H) \mathbf{\Phi}_{i+1} \mathbf{\Phi}_{i+1}^H - \mathbf{\Phi}_{i+1} \mathbf{\Phi}_{i+1}^H (\mathbf{F} - \mathbf{F}^H) \right] \\
+ \left[ (\mathbf{F} - \mathbf{F}^H) (\mathbf{\Phi}_{i+1} + \mathbf{\Phi}_{i+1}^H) + (\mathbf{\Phi}_{i+1} + \mathbf{\Phi}_{i+1}^H) (\mathbf{F} - \mathbf{F}^H) \right] \\
+ 2 (\mathbf{F} - \mathbf{F}^H) - 2 (\mathbf{\Phi}_{i+1} - \mathbf{\Phi}_{i+1}^H) = 0
\end{align*}
\]  

(3.67)
Since \( F - F^H = S \), a more compact formulation of this result reads

\[
S\Psi + \Psi S = 2\left(\Phi_{i+1} - \Phi_{i+1}^H\right) \tag{3.68}
\]

where the \( m \times m \) matrix \( \Psi \) is defined as

\[
\Psi = (I_{m \times m} + \Phi_{i+1})(I_{m \times m} + \Phi_{i+1})^H \tag{3.69}
\]

Equation (3.68) can now be re-written as

\[
\left[(\Psi \otimes I_{m \times m}) + (I_{m \times m} \otimes \Psi)\right] \text{vec}(S) = 2\text{vec}\left(\Phi_{i+1} - \Phi_{i+1}^H\right) \tag{3.70}
\]

Define the Kronecker sum operator \( \oplus \) [19] as

\[
\Psi \oplus \Psi = \left[(\Psi \otimes I_{m \times m}) + (I_{m \times m} \otimes \Psi)\right] \tag{3.71}
\]

If the \( m^2 \times m^2 \) Kronecker sum \( (\Psi \oplus \Psi) \) is nonsingular, the solution is

\[
S = 2(\Psi \oplus \Psi)^{-1}\text{vec}\left(\Phi_{i+1} - \Phi_{i+1}^H\right) \tag{3.72}
\]

The solution \( S \) shown in equation (3.72) is a skew symmetric matrix because \( \Psi \) is symmetric and \( (\Phi_{i+1} - \Phi_{i+1}^H) \) is skew symmetric. Given the matrix \( S \), the rotation matrix can be calculated with equation (3.64). In the case \( m = 3 \), the matrix \( S \) takes the form

\[
S = \begin{bmatrix}
0 & -s_3 & s_2 \\
 s_3 & 0 & -s_1 \\
- s_2 & s_1 & 0
\end{bmatrix} \tag{3.73}
\]

where the \( s_i \)'s, \( 1 \leq i \leq 3 \), are called the Rodrigues parameters. The Cayley formula (3.64) yields

\[
R_{i+1} = \frac{1}{1 + \sum_{i=1}^3 s_i^2} \begin{bmatrix}
1 + s_1^2 - s_2^2 - s_3^2 & 2(s_1s_2 - s_3) & 2(s_1s_3 - s_2) \\
2(s_2s_1 + s_3) & 1 - s_1^2 + s_2^2 - s_3^2 & 2(s_2s_3 - s_1) \\
2(s_3s_1 - s_2) & 2(s_3s_2 + s_1) & 1 - s_1^2 - s_2^2 + s_3^2
\end{bmatrix} \tag{3.74}
\]
It can be proved that the axis of rotation is parallel to the vector \([s_1, s_2, s_3]^H\) and the rotation angle \(\varphi\) satisfies \(\tan(\varphi/2) = s_1^2 + s_2^2 + s_3^2\) \([18]\). These parameters can therefore be calculated directly from the matrix \(S\).

When the frame-to-frame rotation is small, a first order approximation of the rotation matrix \(R_{i+1}\) can be made. In this case, the motion model becomes

\[
\mathbf{x}_{i+1} = (I_{m \times m} + \Omega_i) \mathbf{x}_i
\]  

(3.75)

where \(\Omega_i\) is a \(m \times m\) skew-symmetric matrix representing the instantaneous angular velocity at time \(i\). The first order approximation in equation (3.75) results from the fact that \(R_{i+1} = \exp(\Omega_i \Delta t)\) when the matrix \(\Omega_i\) is constant over the time interval \(\Delta t\) \([18]\). Given the transition matrix \(\Phi_{i+1}\), it can be verified with the derivation rules in Appendix D that

\[
\min_{\Omega_i} \left\{ \text{trace} \left\{ (\Phi_{i+1} - I_{m \times m} - \Omega_i)^H (\Phi_{i+1} - I_{m \times m} - \Omega_i) \right\} \right\} = \frac{1}{2} (\Phi_{i+1} - \Phi_{i+1}^H)
\]

(3.76)

Equation (3.76) shows that the closest angular velocity matrix \(\Omega_i\) for the motion model in equation (3.75) is simply \(1/2(\Phi_{i+1} - \Phi_{i+1}^H)\).
Chapter 4

ESTIMATION OF THE TRANSLATION

The fast modes of the object motion are represented in the motion model by the time-varying translation vector \( T_{k+1} \). This translation vector acts as a slack variable taking up the motion modes which the transition matrix is unable to capture. The role of the translation vector is therefore to compensate for the difference between the motion generated by the transition matrix and the true motion of the object. However, its role is also to smooth the motion of the object so only the fast motion trends and not the noise are incorporated in the model. To this end, an optimum quadratic tracking algorithm is derived to estimate the translation vectors.

4.1 GENERAL CASE

The general case of non-linear projection is first considered. Recall the multi-dimensional motion model of equation (2.12) and modify it slightly as follows

\[
vec(X_{i+1}) = (I_{pxp} \otimes R_{i+1} \Phi_{i+1}) vec(X_i) + (1 \otimes R_{i+1}) T_{i+1}
\]
\[ \text{vec}(Y_i) = \begin{bmatrix} \mathbf{h}_{i,1}(\text{vec}(X_i)) \\ \vdots \\ \mathbf{h}_{i,p}(\text{vec}(X_i)) \end{bmatrix} = \mathbf{h}_i(\text{vec}(X_i)) \quad (4.77) \]

where \( \mathbf{h}_{i,j}(\text{vec}(X_i)) = \mathbf{h}_i(x_i^{(j)}) \). The vectors \( \text{vec}(X_i) \) and \( \text{vec}(Y_i) \) are \( mp \times 1 \) and \( pl \times 1 \) column vectors, respectively. The \( p \times 1 \) column vector \( \mathbf{1} \) is \([1, \ldots, 1]^H\) and the translation vector \( T_{i+1} \) is a \( m \times 1 \) column vector. This translation vector is common to all the \( p \) features. For \( N \) successive frames, determine the motion vectors \( T_i \)'s, \( 1 \leq i \leq N - 1 \), by minimizing

\[
P = \frac{1}{2}(\text{vec}(Y_N) - \mathbf{h}_N(\text{vec}(X_N)))^H \mathbf{L}_N (\text{vec}(Y_N) - \mathbf{h}_N(\text{vec}(X_N)))
+ \frac{1}{2} \sum_{i=1}^{N-1} (\text{vec}(Y_i) - \mathbf{h}_i(\text{vec}(X_i)))^H \mathbf{L}_i (\text{vec}(Y_i) - \mathbf{h}_i(\text{vec}(X_i)))
+ \frac{1}{2} \sum_{i=1}^{N-1} T_{i+1}^H \mathbf{Q}_{i+1} T_{i+1}
+ \sum_{i=1}^{N-1} \lambda_{i+1}^H ((I_{pxp} \otimes \mathbf{R}_{i+1} \mathbf{D}_{i+1}) \text{vec}(X_i) + (1 \otimes \mathbf{R}_{i+1}) T_{i+1} - \text{vec}(X_{i+1}))
\quad (4.78) \]

where the Lagrangian \( \lambda_{i+1} \) is a \( mp \times 1 \) column vector, \( \mathbf{L}_i, 1 \leq i \leq N \), is a \( pl \times pl \) symmetric matrix and \( \mathbf{Q}_{i+1}, 1 \leq i \leq N - 1 \), is a \( m \times m \) symmetric matrix. To this end define the Hamiltonian \( \mathcal{G}_i \)

\[
\mathcal{G}_i = \frac{1}{2} (\text{vec}(Y_i) - \mathbf{h}_i(\text{vec}(X_i)))^H \mathbf{L}_i (\text{vec}(Y_i) - \mathbf{h}_i(\text{vec}(X_i)))
+ \frac{1}{2} T_{i+1}^H \mathbf{Q}_{i+1} T_{i+1}
+ \lambda_{i+1}^H ((I_{pxp} \otimes \mathbf{R}_{i+1} \mathbf{D}_{i+1}) \text{vec}(X_i) + (1 \otimes \mathbf{R}_{i+1}) T_{i+1} - \text{vec}(X_{i+1}))
\quad (4.79) \]
The optimum \( T_{i+1}, 1 \leq i \leq N - 1 \), for this quadratic tracking system satisfies [67]:

\[
\text{State System: ( for } 1 \leq i \leq N - 1 \text{ )}
\]
\[
0 = \frac{\partial G_i}{\partial \lambda_{i+1}}
\]
\[
= -\text{vec}(X_{i+1}) + (I_{p \times p} \otimes R_{i+1} \Phi_{i+1}) \text{vec}(X_i) + (I \otimes R_{i+1}) T_{i+1}
\]

\[
\text{Costate System: ( for } 2 \leq i \leq N - 1 \text{ )}
\]
\[
\lambda_i = \frac{\partial G_i}{\partial \text{vec}(X_i)}
\]
\[
= (I_{p \times p} \otimes \Phi^H_{i+1} R^H_{i+1}) \lambda_{i+1} - \delta_i^H L_i \text{vec}(Y_i - Y_i^*) + \delta_i^H L_i \delta_i^* \text{vec}(X_i - X_i^*)
\]

\[
\text{Stationarity Condition: ( for } 1 \leq i \leq N - 2 \text{ )}
\]
\[
0 = \frac{\partial G_i}{\partial T_{i+1}} = (I^H \otimes R^H_{i+1}) \lambda_{i+1} + Q_{i+1} T_{i+1}
\]

\[
\text{Boundary Condition :}
\]
\[
\lambda_N = \frac{\partial G_N}{\partial \text{vec}(X_N)}
\]
\[
= -\delta_N^H L_N \text{vec}(Y_N - Y_N^*) + \delta_N^H L_N \delta_N^* \text{vec}(X_N - X_N^*)
\]

(4.80)

where the \( \ell \times p \) matrix \( Y_i^* = h_i(\text{vec}(X_i^*)) \), \( 2 \leq i \leq N \). In the costate and boundary equations shown above, the non-linear projection is linearized about the given positions \( X_i^* \)'s , \( 1 \leq i \leq N \). As an example, these positions can be chosen to be the filtered positions \( \bar{z}_i^{(j)} = \Phi_i \hat{z}_{i-1}^{(j)} \) where \( \Phi_i \) is the output of the recursive least-square algorithm described in Chapter 2 and where \( \bar{z}_1^{(j)} = \hat{z}_1^{(j)} \).
In equation (4.80), \( \mathcal{J}_i \) is the \( p \ell \times mp \) block diagonal matrix:

\[
\mathcal{J}_i = \begin{bmatrix}
J^{(1)}_i & 0 & \ldots & 0 & 0 \\
0 & J^{(2)}_i & 0 & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & \cdots & 0 & J^{(p-1)}_i & 0 \\
0 & 0 & \cdots & 0 & J^{(p)}_i
\end{bmatrix}
\] (4.81)

where \( J^{(j)}_i \) is the Jacobian at \( x^{(j)}_i \). Likewise, the matrix \( \mathcal{J}_i \) has the same block diagonal form except that the Jacobians are evaluated this time at the given positions \( x^{(j)*}_i \). Using the sweep method [21], set

\[
\lambda_i = S_i vec(X_i) + F_i
\] (4.82)

where \( S_i \) is a \( mp \times mp \) matrix and \( F_i \) is a \( mp \times 1 \) vector. Let the \( p \times p \) symmetric matrix \( 1^H \) be denoted by \( 1 \) (all columns and rows filled with 1). Substitute the stationarity condition (4.80) and equation (4.82) in equation (4.77) to obtain

\[
vec(X_{i+1}) = (I_{pxp} \otimes R_{i+1} \Phi_{i+1}) vec(X_i) + (1 \otimes R_{i+1}) T_{i+1}
\]

\[
- (1 \otimes R_{i+1} (Q_{i+1})^{-1} (1^H \otimes R_{i+1}^H) \lambda_{i+1})
\]

\[
= (I_{pxp} \otimes R_{i+1} \Phi_{i+1}) vec(X_i)
\]

\[
- (1 \otimes R_{i+1} (Q_{i+1})^{-1} R_{i+1}^H) (S_{i+1} vec(X_{i+1}) + F_{i+1})
\]

\[
= (I_{mp \times mp} + (1 \otimes R_{i+1} (Q_{i+1})^{-1} R_{i+1}^H) S_{i+1})^{-1}
\]

\[
(I_{pxp} \otimes R_{i+1} \Phi_{i+1}) vec(X_i) - (1 \otimes R_{i+1} (Q_{i+1})^{-1} R_{i+1}^H) F_{i+1}
\]

(4.83)

The Inversion Lemma cannot be used here as the matrix \( (1 \otimes R_{i+1} Q_{i+1}^{-1} R_{i+1}^H) \) is singular. Moreover, the costate system in equation (4.80) becomes

\[
S_i vec(X_i) + F_i = (I_{pxp} \otimes \Phi_{i+1}^H R_{i+1}^H) (S_{i+1} vec(X_{i+1}) + F_{i+1})
\]
Substituting equation (4.83) into equation (4.84) and observing that the equality must hold for any state $\text{vec}(X_i)$, one obtains

$$S_i = (I_{p \times p} \otimes \Phi_{i+1}^H R_{i+1}^H) S_{i+1} \left( I_{mp \times mp} \right)$$

$$+ \left( I \otimes R_{i+1} (Q_{i+1})^{-1} R_{i+1}^H \right) S_{i+1}^{-1} (I_{p \times p} \otimes R_{i+1} \Phi_{i+1}) + \beta_i^H L_i \beta_i^*$$

$$F_i = (I_{p \times p} \otimes \Phi_{i+1}^H R_{i+1}^H) \left( I_{mp \times mp} - \right)$$

$$S_{i+1} \left( I_{mp \times mp} + (I \otimes R_{i+1} Q_{i+1}^{-1} R_{i+1}^H) S_{i+1} \right)^{-1} (I \otimes R_{i+1} Q_{i+1}^{-1} R_{i+1}^H) \right) F_{i+1}$$

$$- \beta_i^H L_i \text{vec}(Y_i - Y_i^*) - \beta_i^H L_i \beta_i^* \text{vec}(X_i^*)$$

(4.85)

The recursions shown in equation (4.85) are similar to the conventional Riccati equations appearing in the solution of an optimum tracking problem. However, here the Jacobian depends on the reconstructed positions $x_i^{(j)}$ which are not known a-priori. It is assumed that the Jacobian $J_i^{(j)}$ at $x_i^{(j)}$ can be replaced by an approximate Jacobian obtained from the data at hand. Then, equation (4.85) can be used to determine the matrices $\{S_{N-1}, \ldots, S_2\}$ and the vectors $\{F_{N-1}, \ldots, F_2\}$ recursively from the boundary condition in equation (4.80) which gives

$$S_N = \beta_N^H L_N \beta_N^*$$

$$F_N = -\beta_N^H L_N \text{vec}(Y_N - Y_N^*) - \beta_N^H L_N \beta_N^* \text{vec}(X_N^*)$$

(4.86)
Once these sequences are known, the Lagrangian $\lambda_{i+1}$, $1 \leq i \leq N - 1$, can be calculated with the help of equation (4.82). Following this, the translation vectors are given by

$$T_{i+1} = -(Q_{i+1})^{-1} \left( 1^H \otimes R_{i+1}^H \right) \lambda_{i+1}$$

$$= -(Q_{i+1})^{-1} \left( 1^H \otimes R_{i+1}^H \right) \left( S_{i+1} vec(X_{i+1}) + F_{i+1} \right)$$

$$= -(Q_{i+1})^{-1} \left( 1^H \otimes R_{i+1}^H \right)$$

$$\left( S_{i+1} \left( I_{p \times p} \otimes R_{i+1} \Phi_{i+1} \right) vec(X_i) + (1 \otimes R_{i+1}) T_{i+1} \right) + F_{i+1} \right)$$

(4.87)

The last equality shown in equation (4.87) follows from the fact that the state $vec(X_{i+1})$ cannot be reconstructed before $T_{i+1}$ is known. Furthermore, the same equality can be re-arranged to yield the following expression for $T_{i+1}$:

$$T_{i+1} = -\left( I_{m \times m} + (Q_{i+1})^{-1} \left( 1^H \otimes R_{i+1}^H \right) \right)^{-1}$$

$$\left( Q_{i+1}^{-1} \left( 1^H \otimes R_{i+1}^H \right) \right)$$

$$\left( S_{i+1} \left( I_{p \times p} \otimes R_{i+1} \Phi_{i+1} \right) vec(X_i) + F_{i+1} \right)$$

(4.88)

The algorithm can be summarized as follows. First determine the sequences $\{S_N, \ldots, S_2\}$ and $\{F_N, \ldots, F_2\}$. Next use equation (4.88) to determine $T_2$ starting with a known or estimated initial state $vec(X_1)$. Once this is done, use the state system equation in (4.80) to build the state $vec(X_2)$ and repeat until the translation $T_N$ and the state $vec(X_N)$ are determined. When the number of image frames $N$ is large, concepts borrowed from receding horizon techniques
in control theory may be used. Receding horizon techniques refer to finite-time optimum control laws designed to look at a terminal time which is constantly $n$ samples away [26]. The technique involves sliding a finite temporal window from sample to sample while generating an optimum control signal for the first sample in the window. It has been shown that receding horizon feedback control laws guarantee asymptotic stability. As such, they can be used to replace the steady state optimum control over an infinite time interval [65]. For the case of estimating the translation vector in the motion model, this concept translates into the following procedure:

- **Step 1**: Replace $N$ by a smaller number of image frames $n$ in the translation estimation algorithm.
- **Step 2**: Set $j=2$
- **Step 3**: Estimate the vectors $T_{n-2+j}, \ldots, T_j$
- **Step 4**: Keep $T_j$ only
- **Step 5**: if $j = N$ go to end
- **Step 6**: Slide the $n$ frames temporal window by one position
- **Step 7**: $j=j+1$
- **Step 8**: Go to step 3
- **Step 9**: end

The advantage of such a procedure is to provide additional smoothing among the translation vectors.
4.2 LINEAR CASE

When the projection $h_k()$ is linear, the algorithm derived in the previous section no longer needs an approximation of the Jacobians as they become position invariant matrices. The goal of this section is to reveal the simplifications that result. Consider again the time-varying linear projection $B_kB_k^H$ introduced earlier in the derivation of the recursive least-square algorithm. The multidimensional model shown in equation (4.77) becomes

$$
vec(X_{i+1}) = (I_{pxp} \otimes R_{i+1}\Phi_{i+1}) vec(X_i) + (1 \otimes R_{i+1}) T_{i+1}
$$

$$
vec(Y_i) = vec(B_iB_i^HX_i) = (I_{pxp} \otimes B_iB_i^H) vec(X_i)
$$

and the system of equations (4.80) becomes

**State System**: ( for $1 \leq i \leq N - 1$ )

$$
0 = \frac{\partial S_i}{\partial \lambda_{i+1}}
= -vec(X_{i+1}) + (I_{pxp} \otimes R_{i+1}\Phi_{i+1}) vec(X_i) + (1 \otimes R_{i+1}) T_{i+1}
$$

**Costate System**: ( for $2 \leq i \leq N - 1$ )

$$
\lambda_i = \frac{\partial S_i}{\partial vec(X_i)}
= \left( I_{pxp} \otimes \Phi_i^H R_i^H \right) \lambda_{i+1} - \left( I_{pxp} \otimes B_iB_i^H \right) L_i vec(Y_i)
+ \left( I_{pxp} \otimes B_iB_i^H \right) L_i (I_{pxp} \otimes B_iB_i^H) vec(X_i)
$$

**Stationarity Condition**: ( for $1 \leq i \leq N - 2$ )

$$
0 = \frac{\partial S_i}{\partial T_{i+1}} = \left( 1^H \otimes R_{i+1}^H \right) \lambda_{i+1} + Q_{i+1} T_{i+1}
$$

**Boundary Condition**: 


\[ \lambda_N = \frac{\partial \mathcal{G}_N}{\partial \text{vec}(X_N)} \]
\[ = -(I_{pxp} \otimes B_NB_N^H)L_N \text{vec}(Y_N) \]
\[ + (I_{pxp} \otimes B_NB_N^H)L_N(I_{pxp} \otimes B_NB_N^H)\text{vec}(X_N) \]

yielding the following Riccati equations:

\[
S_i = (I_{pxp} \otimes \Phi_{i+1}^H R_{i+1}^H) \\
S_{i+1} \left( I_{mpxm} + (1 \otimes R_{i+1}(Q_{i+1})^{-1} R_{i+1}^H) S_{i+1} \right)^{-1} (I_{pxp} \otimes R_{i+1} \Phi_{i+1}) \\
+ (I_{pxp} \otimes B_i B_i^H) L_i (I_{pxp} \otimes B_i B_i^H)
\]

\[
F_i = (I_{pxp} \otimes \Phi_{i+1}^H R_{i+1}^H) \left( I_{mpxm} - \\
S_{i+1} \left( I_{mpxm} + (1 \otimes R_{i+1}(Q_{i+1})^{-1} R_{i+1}^H) S_{i+1} \right)^{-1} (1 \otimes R_{i+1} Q_{i+1}^{-1} R_{i+1}^H) \right) F_{i+1} \\
- (I_{pxp} \otimes B_i B_i^H) L_i \text{vec}(Y_i)
\]

with the initial conditions

\[
S_N = (I_{pxp} \otimes B_NB_N^H)L_N(I_{pxp} \otimes B_NB_N^H)
\]
\[
F_N = -(I_{pxp} \otimes B_NB_N^H)L_N \text{vec}(Y_N)
\]

where

\[
(I_{pxp} \otimes B_i B_i^H) = \begin{bmatrix}
B_i B_i^H & 0 & \ldots & 0 & 0 \\
0 & B_i B_i^H & 0 & \ldots & 0 \\
:\ & : & : & : & : \\
0 & \ldots & 0 & B_i B_i^H & 0 \\
0 & 0 & \ldots & 0 & B_i B_i^H
\end{bmatrix}
\]
As in the non-linear case, equation (4.88) can be used to estimate the vectors $T_{i+1}$ and rebuild the states $\text{vec}(X_{i+1})$ iteratively.

4.3 ALIASING

In the algorithms described in Section 4.1 and 4.2, the action of the matrices $L_i$, $1 \leq i \leq N$, is to weight the $p$ feature positions not only with respect to each other but also relatively to time. The choice of their values can reflect confidence levels in the feature matching (correspondence ambiguities, partial occlusion) or simple changes in the noise level. The action of the matrices $Q_i$, $1 \leq i \leq N$, is to introduce weights among the various translation components. It is suggested in this section that the $Q_i$'s be used to control motion aliasing. Indeed, it is shown that the motion of a translating object can be unambiguously estimated up to a certain range. Beyond this limit, motion aliasing occurs meaning that the true translation of the object can no longer be recovered. The $Q_i$ can therefore be used to help reduce the amount of aliasing by giving larger weights to the translation components exceeding motion bandwidth. The cost of such weighting is a loss of tracking ability. The derivation of the domains where the motion parameters are free of aliasing is in order. The study is limited to the case $m = 2$ (two dimensional motion) so the results apply to any algorithm for estimating two dimensional displacement fields (see section 1.4).

4.3.1 VELOCITY MAGNITUDE

Conditions under which velocity aliasing occurs are now derived. It is shown that velocity aliasing takes its origin from either spatial or temporal aliasing. Consider the space and time continuous image $I(s, t)$, where $s = [x, y]^H$ is the position vector, to be a two dimensional "sinusoidal" signal moving at constant
speed $v = [v_x, v_y]^H$, that is

$$I(s, t) = \sin \left[ 2\pi (f_0^0)^H (s - vt) \right]$$  \hspace{1cm} (4.94)$$

where $f_0 = [f_0^0, f_0^0]^H$ is the vector denoting the horizontal and vertical frequency of the sinusoid, respectively. With the help of equation (1.10), the signal $I()$ can also be expressed as

$$I(s, t) = \sin \left[ 2\pi \left( (f_0^0)^H s + f_0^0 t \right) \right]$$  \hspace{1cm} (4.95)$$

where $f_0^0 = -(f_0^0)^H v$. At this point, decompose the velocity vector $v$ into $v_{\parallel}$ and $v_{\perp}$, a vector parallel and perpendicular to the sinusoidal wave, respectively, as shown in Figure 2 below. Observe that in the image $I()$, the set of points $s_0$ sharing a given phase $\theta$ at time $t = t_0$ satisfies $(f_0^0)^H (s_0 - vt_0) = \theta$. If $v = v_{\parallel}$, the motion is parallel to the sinusoidal wave. This means that at time $t = t_0 + \Delta t$, the points $s_0$ will maintain the same phase $\theta$, that is $(f_0^0)^H (s_0 - v_{\parallel}(t_0 + \Delta t)) = \theta$. Subtracting the phase of a point $s_0$ at time $t_0$ from the phase of the same point at time $t_0 + \Delta t$, and noticing that $\Delta t$ is arbitrary, it can be concluded that

$$(f_0^0)^H v_{\parallel} = 0$$  \hspace{1cm} (4.96)$$

In the case when $v = v_{\parallel}$, equation (1.10) becomes $f_0^0 = -(f_0^0)^H v_{\parallel}$ and, from equation (4.96), it can be concluded that $f_0^0 = 0$. In other words, the component $v_{\parallel}$ of the velocity vector $v$ does not contribute to the temporal frequency. This results since, as shown in equation (4.96), these two vectors are perpendicular to each other. However, since $v_{\parallel}$ is by definition perpendicular to $v_{\perp}$, it follows that the two dimensional vector $v_{\perp}$ is parallel to the two dimensional vector $f_0^0$. Therefore, there exists a real number $k_{\perp}$ such that $v_{\perp} = k_{\perp} f_0^0$. From equation
Figure 2. SINUSOIDAL SIGNAL IN MOTION

\[
\sin \left[ 2\pi f_s \left( H_0 (s - vt) \right) \right]
\]
(1.10), it can now be written

\[
\begin{aligned}
    f_t^0 &= -(f_s^0)^H v = -(f_s^0)^H (v_\parallel + v_\perp) \\
    &= -(f_s^0)^H v_\perp = -k_\perp (f_s^0)^H f_s^0 \\
    &= -k_\perp \|f_s^0\|^2 \quad (4.97)
\end{aligned}
\]

Equation (4.97) yields \( k_\perp = -f_t^0 \|f_s^0\|^{-2} \) (assuming that \( f_s \neq 0 \). If \( f_s^0 = 0 \), take \( k_\perp = 0 \). Equation (4.97) also shows that only \( v_\perp \), the velocity vector perpendicular to the sinusoidal wave, contributes to the temporal frequency \( f_t^0 \). This proves the aperture problem which states that motion parallel to an edge cannot be recovered unambiguously [111]. Equation (4.97) leads to the additional observation that in the spatio-temporal frequency domain, the velocity magnitude can be defined as the tangent of the angle between vector \([f_s^0, f_t^0]^H\) and its projection \([f_s^0, 0]^H\). See Figure 3.

The tangent of \( \Psi \) is

\[
\tan(\Psi) = \frac{f_t^0}{\|f_s^0\|} = -\frac{(f_s^0)^H v_\perp}{\|f_s^0\|} \quad (4.98)
\]

Moreover, \( |(f_s^0)^H v_\perp| = \|f_s^0\| \|v_\perp\| \) since the vectors \( f_s^0 \) and \( v_\perp \) are parallel. With the help of equation (4.98), one can now write

\[
|\tan(\Psi)| = \frac{\|f_s^0\| \|v_\perp\|}{\|f_s^0\|} = \|v_\perp\| = \frac{f_t^0}{\|f_s^0\|} \quad (4.99)
\]

Equation (4.99) is the key to analyzing conditions under which velocity aliasing occurs. To study velocity aliasing along each spatial dimension, calculate the \( j \)th, \( j = 1, 2 \), component \( v_{\perp j} \) of the vector \( v_\perp \). Recalling that

\[
v_\perp = k_\perp f_s^0 = -\frac{f_t^0}{\|f_s^0\|^2} f_s^0 \quad (4.100)
\]

it follows from equation (4.100) that

\[
v_{\perp j} = -\frac{f_t^0 f_{sj}^0}{\|f_s^0\|^2} \quad (4.101)
\]
Figure 3. VELOCITY AS AN ANGLE $\Psi$
4.3.2 VELOCITY ALIASING

The results derived in the previous section are now used to determine the conditions under which velocity aliasing occurs. To this end, assume that the image \( I() \) has been sampled according to a given spatio-temporal lattice. To identify the cause of velocity aliasing, spatial and temporal aliasing is analyzed in the particular case of a three dimensional rectangular grid. Velocity bandwidth will be defined afterwards. Let \( f_x^0 \) and \( f_y^0 \) denote the horizontal and vertical frequency of the sinusoidal signal \( \sin (2\pi (f_x^0 x + f_y^0 y + f_t^0 t)) \). Furthermore denote the horizontal, vertical and temporal sampling frequency as \( F_x \), \( F_y \) and \( F_t \), respectively. Spatial aliasing occurs if either \( f_x^0 \geq F_x/2 \) or \( f_y^0 \geq F_y/2 \), or equivalently if the vector \([f_x^0, f_y^0]^H\) is no longer inside the three dimensional Nyquist cube shown in Figure 4.

In this case the velocity \( v_\perp \) magnitude verifies

\[
\|v_\perp\| = \frac{f_t^0}{\sqrt{(f_x^0)^2 + (f_y^0)^2}} \leq \frac{4f_t^0}{\sqrt{F_x^2 + F_y^2}} \tag{4.102}
\]

Thus, the value \( \|v_\perp\| \) is smaller than \( 4f_t^0/\sqrt{F_x^2 + F_y^2} \) which is the minimum admissible velocity magnitude at the given temporal frequency \( f_t^0 \) (which is assumed to strictly less than \( F_t/2 \)). The result is the aliased vector shown in Figure 4. Observe that the component of the aliased vector along the temporal frequencies remains \( f_t^0 \) while the components along the spatial frequencies are the result of mapping \( f_x^0 \) and \( f_y^0 \) back into the allowed Nyquist square. The aliased velocity vector is now defined by an angle \( \varphi \neq \Psi \). Concentrate now on temporal aliasing which occurs when \( f_t^0 \geq F_t/2 \). In this case, \( f_t^0 \) induces a velocity whose magnitude is too large as depicted in Figure 5. The resulting
Figure 4. EXAMPLE OF SPATIAL ALIASING
Figure 5. EXAMPLE OF TEMPORAL ALIASING
velocity magnitude is
\[
\|v_\perp\| = \frac{f_0^0}{\sqrt{(f_x^0)^2 + (f_y^0)^2}} \geq \frac{F_t}{2\sqrt{(f_x^0)^2 + (f_y^0)^2}} \tag{4.103}
\]
where \(F_t/2\sqrt{(f_x^0)^2 + (f_y^0)^2}\) is the maximum admissible velocity magnitude given \(f_x^0\) and \(f_y^0\). From Figure 5, it can be seen that the aliased vector results from mapping the vector \(f_t^0\) back into the allowed Nyquist range \([0, F_t/2]\) while both \(f_x^0\) and \(f_y^0\) remain unchanged (because \(f_x^0 < F_x/2\) and \(f_y^0 < F_y/2\)). The aliased vector is defined by an angle \(\phi \neq \Psi\). When the aliased \(f_t^0\) is of opposite direction, the effect is similar to the well known backward rotation effect sometimes observed with a rotating spoke pattern illuminated by stroboscope flashes.

The inequalities in equation (4.102) and (4.103) suggest that given either \(f_0^0\) or \((f_x^0, f_y^0)\), there are bounds on the velocity magnitude. In particular, the inequality in equation (4.103) involves an upper bound which can be interpreted as the maximum bandwidth for \(\|v_\perp\|\). This definition of velocity is well suited to most applications as it is usually easier to use spatial filters (to fix \(f_x^0\) and \(f_y^0\)) as opposed to linear temporal filters. For the latter to perform well, the constant velocity assumption is critical. To define precisely the velocity bandwidth, first observe that the maximum allowable velocity magnitude is inversely proportional to \(\sqrt{(f_x^0)^2 + (f_y^0)^2}\). This prompts the velocity bandwidth definition at a given radial frequency \(\sqrt{(f_x^0)^2 + (f_y^0)^2}\) to be the domain described by horizontal and vertical frequencies \(f_x\) and \(f_y\), respectively, such that
\[
(f_x^0)^2 + (f_y^0)^2 = (k^0)^2 \frac{F_x^2 + F_y^2}{4} \leq f_x^2 + f_y^2 < \frac{F_x^2 + F_y^2}{4} \tag{4.104}
\]
where the lowest bound for \(f_x^2 + f_y^2\) has been expressed as a fractional part of the spatial Nyquist rates \(F_x/2\) and \(F_y/2\) square \((0 \leq k^0 < 1)\). The domain defined by equation (4.104) can be viewed as the frequency range filtered out by an ideal circular two dimensional filter with cut-off frequency
Clearly, from equation (4.99), the largest $|\tan(\Psi)|$ value (the velocity magnitude) is found when $f_t^0 = F_t/2$ (maximum temporal frequency) and $f_x^2 + f_y^2 = (f_x^0)^2 + (f_y^0)^2$ (minimum spatial frequency). This is shown in Figure 6 where the maximum angle is denoted by $\Psi_{\text{max}}$.

On the other hand, the zero velocity magnitude is reached when $f_t^0 = 0$. Hence

$$0 \leq \|v\| \leq \frac{F_t}{k \sqrt{F_x^2 + F_y^2}}$$  \hspace{1cm} (4.105)$$

The inequalities in equation (4.105) describe a circular domain defining a velocity bandwidth at the radial frequency $\sqrt{(f_x^0)^2 + (f_y^0)^2}$. Temporal velocity aliasing cannot occur in this domain. Figure 7 shows the circular velocity band for $k_r = 1/2$ and $F_x = F_y = F_t = 1$ (normalized sampling frequencies).

Note that the radius of the velocity baseband increases as $k_r^0$ decreases. This means that the velocity baseband is small in image areas of high spatial frequency content and very large in uniform regions. As opposed to evaluating the velocity bandwidth at $\sqrt{(f_x^0)^2 + (f_y^0)^2}$, it is now established by selecting the conditions $f_x^0 \geq k_x F_x/2$ or $f_y^0 \geq k_y F_y/2$ where $0 \leq k_x, k_y < 1$. The resulting domain can be seen as the frequency range filtered out by a separable two dimensional filter with horizontal and vertical cut-off frequencies $f_x^0$ and $f_y^0$, respectively. From equation (4.101)

$$v_{\perp x} = -\frac{f_t^0 f_x^0}{(f_x^0)^2 + (f_y^0)^2} \hspace{1cm} \text{and} \hspace{1cm} v_{\perp y} = -\frac{f_t^0 f_y^0}{(f_x^0)^2 + (f_y^0)^2}$$  \hspace{1cm} (4.106)$$

By setting $f_t^0 = F_t/2$ and the vertical frequency to $f_y^0 = k_y^0 F_y/2$ and by varying the horizontal frequency $f_x^0$ from 0 to $k_x^0 F_x/2$, move along segment A as shown in Figure 8. Likewise, setting $f_t^0 = F_t/2$ and the horizontal frequency to $f_x^0 = k_x^0 F_x/2$ and varying the vertical frequency $f_y^0$ from 0 to $k_y^0 F_y/2$, move along segment B as shown in Figure 8.
Figure 6. MAXIMUM VELOCITY FOR RADIAL HIGH-PASS FILTER
Figure 7. CIRCULAR VELOCITY BANDWIDTH
Figure 8. CONTOUR FOR DETERMINING VELOCITY BANDWIDTH
Along segment A and segment B, equations (4.106) become

\[
\begin{align*}
\text{SEGMENT A} & & \text{SEGMENT B} \\

v_{x} &= \frac{F_{1}a_{x}F_{x}}{a_{x}^{2}F_{x}^{2}+(k_{y}^{0})^{2}F_{y}^{2}} & v_{x} &= -\frac{F_{1}k_{x}^{0}F_{x}}{(k_{x}^{0})^{2}F_{x}^{2}+a_{x}^{2}F_{y}^{2}} \\

v_{y} &= \frac{F_{1}k_{y}^{0}F_{y}}{a_{x}^{2}F_{x}^{2}+(k_{y}^{0})^{2}F_{y}^{2}} & v_{y} &= -\frac{F_{1}a_{y}F_{y}}{(k_{x}^{0})^{2}F_{x}^{2}+a_{y}^{2}F_{y}^{2}} \\

& & (4.107)
\end{align*}
\]

where \(0 \leq a_{x} \leq k_{x}^{0}\) and \(0 \leq a_{y} \leq k_{y}^{0}\). Equations (4.108) are parametrized by \(a_{x}\) and \(a_{y}\). Figure 9 shows the “clover-shaped” velocity band obtained from equation (4.108) when \(k_{x}^{0} = k_{y}^{0} = 1/2\) and \(F_{1} = F_{x} = F_{y} = 1\). Similar shapes are obtained whenever \(k_{x}^{0} = k_{y}^{0}\). As expected, the velocity bandwidth increases as \(k_{x}^{0}\) and \(k_{y}^{0}\) decrease (smaller spatial frequencies). Similar procedures can be used to define permissible velocity domains for sampling grids other than the rectangular grid. Figure 10 shows the boundaries that one must work with to calculate the velocity bandwidth at \((f_{x}^{0}, f_{y}^{0})\) for a field-interlaced rectangular sampling lattice.

The interlacing of the lines makes the vertical-temporal frequency alias-free domain to assume a diamond shape, the result of undergoing a \(\pi/4\) rotation [34]. As a consequence, the maximum possible temporal frequency \(k_{y}^{0}F_{1}/2\) allowed along the vertical frequency axis depends on the chosen value \(f_{y}^{0}\) and it can be verified that \(f_{y}^{0} = k_{y}^{0}F_{y}/2 = (1 - k_{1}^{0})F_{y}/2\). Analytically, the velocity bandwidth
Figure 9. CLOVER-SHAPED VELOCITY BANDWIDTH
Figure 10. CONTOUR FOR DETERMINING VELOCITY BANDWIDTH

\[ k \frac{F}{t} \frac{0.5 F_t}{2} \]

\[ F \frac{v}{2} (1 - k \frac{0}{t}) \]

Segment A

Segment B

\[ k \frac{F_x}{x} \frac{0.5 F_x}{2} \]

\[ 0.5 F_y \]

\[ 0.5 F_x \]
is defined by

\[ v_{\perp x} = \frac{(1-k_y^0)F_xa_xF_x}{a_x^2F_x^2 + (k_y^0)^2F_y^2} \quad \text{SEGMENT A} \]

\[ v_{\perp x} = \frac{(1-a_y)F_yk_y^0F_x}{(k_y^0)^2F_x^2 + a_y^2F_y^2} \quad \text{SEGMENT B} \]

\[ v_{\perp y} = \frac{(1-k_x^0)F_yc_yF_y}{a_y^2F_y^2 + (k_x^0)^2F_y^2} \quad \text{SEGMENT A} \]

\[ v_{\perp y} = \frac{(1-a_y)F_ya_yF_y}{(k_x^0)^2F_x^2 + a_y^2F_y^2} \quad \text{SEGMENT B} \]

(4.108)

where \(0 \leq a_x \leq k_x^0\) and \(0 \leq a_y \leq k_y^0\). Figure 11 shows the corresponding velocity bandwidth for \(k_x^0 = k_y^0 = 1 - k_x^0 = 1/2\).

As before, the sampling frequencies are assumed to be \(F_x = F_y = F_t = 1\). In the case of a quincunx sampling grid [34], the Nyquist volume takes the shape of a diamond both in the vertical-temporal and the horizontal-temporal plane and constraints similar to those between \(f_x^0\) and \(f_y^0\) also exist between \(f_t^0\) and \(f_x^0\). As a special case, consider the velocity bandwidth when \(k_x^0 = k_y^0\) so that the temporal frequency upper limit is identical for both, the horizontal and vertical, frequency axis. The limit is \(f_t^0 = k_t^0F_t/2\) with \(1 - k_t^0 = k_x^0 = k_y^0\). Figure 12 shows different segments that must be considered to determine the velocity bandwidth for images sampled with a quincunx sampling grid.

Three segments are required to determine the velocity bandwidth if \(k_x^0 > 1/2\). The velocity bandwidth along segment C is determined by replacing \(f_x^0\) and \(f_y^0\) in equation (4.106) by \(a_xF_x/2\) and \(a_yF_y/2\), respectively with \(a_x = 1 - a_y\).

It can be shown that the effect of segment C on the shape of the alias-free domain is to bring additional leaves to the clover shape shown in Figure 9. For a quincunx sampling grid, it might be desirable in some situations to determine
Figure 11. VELOCITY BANDWIDTH FOR FIELD-INTERLACED SAMPLING GRID
Figure 12. CONTOUR FOR DETERMINING VELOCITY BANDWIDTH
the velocity bandwidth corresponding to the frequency domain filtered out by a non-separable diamond-shaped two dimensional filter. In this case, only one segment connecting \( f_x^0 \) on the horizontal frequency axis to \( f_y^0 \) on the vertical frequency axis needs to be considered. Along this segment, the variations of the parameters \( a_x \) and \( a_y \) are governed by \( a_y/k_y^0 = 1 - (a_x/k_x^0) \) and the resulting velocity bandwidth takes an oval shape.

The results derived in this section, although restricted to the two dimensional case, indicate that motion aliasing occurs as a by-product of either temporal or spatial aliasing. Estimated motion parameters which do not comply with the aliasing-free domains provide an indicator that the feature correspondence might be subject to ambiguities. In this case, the corresponding weights in the matrix \( L_i \) should be adjusted to reflect this lack of confidence. An alternative solution is to develop the model so as to minimize aliasing problem by shifting dynamically the amount of motion in the transition matrix and the translation vector. This can be done by monitoring and tuning the bandwidth of both the transition matrix and the translation vector estimators.
Chapter 5

MOTION BANDWIDTH

In this chapter, the flexibility of the stochastic approximation framework introduced in Chapter 3 is used to derive a general policy for distributing estimation bandwidth among the transition matrix estimator and the translation vector estimator. The discussion starts by showing that these two estimation algorithms share common algorithmic features at the heart of the bandwidth allocation problem.

5.1 DUALITY

This section reveals the similarities between the problem of estimating the transition matrices $\Phi_{k+1}$ and the problem of estimating the translation vectors $T_{k+1}$. In particular, it is shown that these problems are dual problems. This duality reveals the complementary roles played by the transition matrix estimator and the translation vector estimator.

The calculations of the time-varying transition matrices and translation vectors in Chapter 3 and Chapter 4 both include recursive Riccati equations. This similarity is not a coincidence but the result of the duality existing between
estimation and control problems as shown now. First, recall that a solution for the matrix $\Phi_{k+1}$ was obtained in Chapter 3 by re-writing the multi-feature model

$$\Phi_{k+1}X_k = X_{k+1}$$  \hspace{1cm} (5.109)

as

$$\left(X_k^H \otimes I_{m \times m}\right) \Phi_{k+1} = vec(X_{k+1})$$  \hspace{1cm} (5.110)

Because the estimated transition matrix $\Phi_{k+1}$ does not necessarily describe the points trajectory perfectly, one could raise the following question. Given the transition matrix $\Phi_{k+1}$ and the positions $X_{k+1}$, find all the possible positions $Z_k$ verifying

$$\Phi_{k+1}Z_k = X_{k+1}$$  \hspace{1cm} (5.111)

or, equivalently,

$$(I_{p \times p} \otimes \Phi_{k+1}) vec(Z_k) = vec(X_{k+1})$$  \hspace{1cm} (5.112)

so the transition matrix explains perfectly the motion from $Z_k$ to $X_{k+1}$. The $mp \times mp$ matrix $(I_{p \times p} \otimes \Phi_{k+1})$ in equation (5.112) can be viewed as an endomorphism (linear function) $\varphi$ from $\mathbb{R}^{mp}$ to $\mathbb{R}^{mp}$. Let $\text{Im}(\varphi)$ denote the range of $\varphi$ and let $\text{Ker}(\varphi)$ be the null space of $\varphi$, where

$$\text{Im}(\varphi) = \left\{ \varphi(vec(Z_k)) \text{ with } vec(Z_k) \in \mathbb{R}^{mp} \right\}$$

$$\text{Ker}(\varphi) = \left\{ vec(Z_k) \in \mathbb{R}^{mp} \text{ such that } \varphi(vec(Z_k)) = 0 \right\}$$  \hspace{1cm} (5.113)

Let $\lambda_{k+1}$ be a vector in $\mathbb{R}^{mp}$. Introduce the scalar product from $\mathbb{R}^{mp} \times \mathbb{R}^{mp}$ to $\mathbb{R}$:

$$\mathbb{R}^{mp} \times \mathbb{R}^{mp} \rightarrow \mathbb{R}$$  \hspace{1cm} (5.114)

$$\left(\lambda_{k+1}, \text{vec}(X_{k+1})\right) \rightarrow \lambda_{k+1}^H \text{vec}(X_{k+1})$$
A well known result in linear algebra states that

$$\text{Im}(\varphi) = \{\text{Ker}(\varphi^H)\}^\perp$$  \hspace{1cm} (5.115)

where $\varphi^H$ is the transpose of $\varphi$ and where

$$\{\text{Ker}(\varphi^H)\}^\perp = \left\{\text{vec}(X_{k+1}) \text{ such that } \forall \lambda_{k+1} \in \text{Ker}(\varphi^H), \lambda_{k+1}^H \text{vec}(X_{k+1}) = 0 \right\}$$  \hspace{1cm} (5.116)

The result in equation (5.115) can be used as follows. Assume that $\lambda_{k+1}$ is a vector in $\text{Ker}(\varphi^H)$. Take the transpose on both sides of equation (5.112) and then multiply by $\lambda_{k+1}$ to obtain

$$\text{vec}(Z_k)^H \left( I_{p \times p} \otimes \Phi_{k+1}^H \right) \lambda_{k+1} = \text{vec}(X_{k+1})^H \lambda_{k+1}$$

$$= \lambda_{k+1}^H \text{vec}(X_{k+1})$$  \hspace{1cm} (5.117)

From equation (5.115), it can be realized that the inner product $\lambda_{k+1}^H \text{vec}(X_{k+1})$ in equation (5.117) is equal to zero if and only if the vector $\text{vec}(X_{k+1})$ is an element of the set $\text{Im}(\varphi)$. In other words, there exists a solution $\text{vec}(Z_k)$ to the system in equation (5.112) if and only if

$$\lambda_{k+1}^H \text{vec}(X_{k+1}) = 0$$  \hspace{1cm} (5.118)

for any vector $\lambda_{k+1}$ in the set $\text{Ker}(\varphi^H)$, or equivalently, for any vector $\lambda_{k+1}$ such that

$$(I_{p \times p} \otimes \Phi_{k+1}^H) \lambda_{k+1} = 0$$  \hspace{1cm} (5.119)

Now, observe that equation (5.118) represents the state system constraint in equation (4.80). Furthermore, by substituting equation (5.119) in the costate
system in equation (4.80), notice that equation (5.119) represents the sweep method postulated in equation (4.82). It follows that the problem of finding the solution $Z_k$ in equation (5.112) can be transposed to the problem of estimating the translation vector $T_{k+1}$ as done in Chapter 4. Because of this duality, the estimation of $\Phi_{k+1}$ in the motion model

$$
X_{k+1} = \Phi_{k+1}X_k + (1^H \otimes T_{k+1})
$$

(5.120)
can be done by using the positions $X_k$ and $X_{k+1}$. The discrepancies between the positions produced by the transition matrix and the observed positions are handled through the estimation of the auxiliary vector $T_{k+1}$. In some sense, the vector $T_{k+1}$ brings controllability in the motion estimation system. Clearly, the transition matrix and the translation vector play complementary roles and the problem of their relative importance must now be addressed. For example, the bandwidth of the transition matrix estimator can be made large to make the matrix $\Phi_{k+1}$ take up as much motion as possible. On the other hand, variations bounds could be put on $\Phi_{k+1}$ thereby leaving to the translation vectors the responsibility of modeling the rapidly changing motion modes. This can be achieved by choosing an exponential forgetting profile where the $\beta_i^{[1:i+1]}$s are set to one and the $\lambda^{[i:i]}$'s are set to an arbitrary constant close to but less than one.

A more complex, optimal, methodology is proposed in the next two sections whereby estimation bandwidth is determined from the a-posteriori errors of a multidimensional recursive least-square algorithm operating in parallel to the true non-linear motion estimation algorithm. This additional algorithm operates on the estimated original features positions only and its sole purpose is to provide the bandwidth information the model-based motion estimator needs for the next iteration.
5.2 THE MULTIDIMENSIONAL RLS ALGORITHM

In this section, the conventional least-square estimate of the transition matrix is derived in a multidimensional setting. The projection operator is the identity operator. As a result, the motion model is simply

\[ x_{k+1} = \Phi_{k+1}x_k \]  \hspace{1cm} (5.121)

The Kronecker operator \( \otimes \) and the stacking operator \( \text{vec}(\cdot) \) are again extensively used to develop the solution. The derivations begin with the calculation of the least-square solution for the case where the motion model is time invariant. The recursive least-square solution is then calculated by computing this solution recursively.

5.2.1 LEAST-SQUARE SOLUTION

To calculate the least-square solution of the motion model, it is assumed as in Section 3.1 that the transition matrix \( \Phi \) is constant over image frames \( i = 1 \) to \( i = N \). The positions of the \( p \) features are \( x_i^{(1)}, \ldots, x_i^{(p)} \) at time \( i \) and \( x_{i+1}^{(1)}, \ldots, x_{i+1}^{(p)} \) at time \( i + 1 \). Let \( e_i^{(j)} = x_{i+1}^{(j)} - \Phi x_i^{(j)}, 1 \leq j \leq p \), and define the \( m \times p \) matrices

\[
\begin{align*}
X_{i+1} &= \begin{bmatrix} x_{i+1}^{(1)}, & \ldots, & x_{i+1}^{(p)} \end{bmatrix} \\
X_i &= \begin{bmatrix} x_i^{(1)}, & \ldots, & x_i^{(p)} \end{bmatrix} \\
E_i &= \begin{bmatrix} e_i^{(1)}, & \ldots, & e_i^{(p)} \end{bmatrix}
\end{align*}
\]  \hspace{1cm} (5.122)

The functional to be minimized is

\[
\left\{ \sum_{i=1}^{i=N-1} \beta_i^{[1:i:N]} \left( \sum_{j=1}^{j=p} e_i^{(j)} H e_i^{(j)} \right) \right\} = \text{trace} \left\{ \sum_{i=1}^{i=N-1} \beta_i^{[1:i:N]} E_i^H E_i \right\}
\]  \hspace{1cm} (5.123)
Since \( E_i = X_{i+1} - X_i \)

\[
\text{trace} \left\{ \sum_{i=1}^{i=N-1} \beta_i^{[1:N]} E_i^H E_i \right\} = \text{trace} \left\{ \sum_{i=1}^{i=N-1} \beta_i^{[1:N]} [X_{i+1} - \Phi X_i]^H [X_{i+1} - \Phi X_i] \right\}
\]

\[
= \text{trace} \left\{ \sum_{i=1}^{i=N-1} \beta_i^{[1:N]} X_{i+1}^H X_{i+1} \right\}
- \text{trace} \left\{ \sum_{i=1}^{i=N-1} \beta_i^{[1:N]} \Phi X_i X_i^H \right\}
- \text{trace} \left\{ \sum_{i=1}^{i=N-1} \beta_i^{[1:N]} \Phi^H X_{i+1} X_i^H \right\}
+ \text{trace} \left\{ \sum_{i=1}^{i=N-1} \beta_i^{[1:N]} \Phi X_i X_i^H \Phi^H \right\}
\]

(5.124)

The minimization is carried out by finding the transition matrix \( \Phi_N \) satisfying

\[
\frac{\partial \text{trace} \left\{ \sum_{i=1}^{i=N-1} \beta_i^{[1:N]} E_i^H E_i \right\}}{\partial \Phi} = \sum_{i=1}^{i=N-1} \beta_i^{[1:N]} \frac{\partial \text{trace} \left\{ E_i^H E_i \right\}}{\partial \Phi} = 0_{m \times m} \quad (5.125)
\]

or, equivalently,

\[
\sum_{i=1}^{i=N-1} \beta_i^{[1:N]} \frac{\partial \text{trace} \left\{ E_i^H E_i \right\}}{\partial \text{vec}(\Phi)} = \text{vec}(0_{m \times m}) \quad (5.126)
\]

Using the derivation rules of Appendix D, equation (5.126) leads to

\[
\text{vec}(0_{m \times m}) = \sum_{i=1}^{i=N-1} \beta_i^{[1:N]} \text{vec}(\Phi_N X_i X_i^H) - \sum_{i=1}^{i=N-1} \beta_i^{[1:N]} \text{vec}(X_{i+1} X_i^H)
\]

(5.127)

It follows that the solution \( \Phi_N \) obtained with \( p \) distinct features and \( N \) image frames is given by

\[
\Phi_N = \left( \sum_{i=1}^{i=N-1} \beta_i^{[1:N]} X_{i+1} X_i^H \right)^{-1} \left( \sum_{i=1}^{i=N-1} \beta_i^{[1:N]} X_i X_i^H \right) \Theta_N \Gamma_N^{-1}
\]

(5.128)

where \( \Gamma_N \) and \( \Theta_N \) are \( m \times m \) matrices.
5.2.2 RECURSIVE LEAST-SQUARE

The recursive implementation of the Least-Square solution leads to a time-varying estimate of the transition matrix. In particular, a recursive algorithm is sought such that given $\Phi_N$ in equation (5.128), the solution at time $N + 1$ is

$$
\Phi_{N+1} = \left( \sum_{i=1}^{i=N} \beta_i^{[1:N+1]}X_{i+1}X_i^H \right)^{-1} \Theta_{N+1} \Gamma_{N+1}^{-1} \left( \sum_{i=1}^{i=N} \beta_i^{[1:N+1]}X_iX_i^H \right) ^{-1}
$$

To achieve this, a stochastic Newton approximation framework is chosen to define the evolution of the weights $\beta_i^{[1:j]}$, $1 \leq i \leq N + 1$, $1 \leq j \leq i - 1$ in time. See equation (3.35). The matrices $\Gamma_{N+1}$ and $\Theta_{N+1}$ can then be computed as

$$
\Gamma_{N+1} = \beta_N^{[1:N+1]}X_NX_N^H + \lambda^{[1:N]}\Gamma_N
$$

(5.130)

$$
\Theta_{N+1} = \beta_N^{[1:N+1]}X_{N+1}X_N^H + \lambda^{[1:N]}\Theta_N
$$

(5.131)

$\Gamma_{N+1}^{-1}$ can be calculated from equation (5.130) and the Inversion Lemma:

$$
\Gamma_{N+1}^{-1} = (\lambda^{[1:N]})^{-1}\Gamma_N^{-1} \left\{ I_{m \times m} - \beta_N^{[1:N+1]}X_N( I_{p \times p} + \beta_N^{[1:N+1]}(\lambda^{[1:N]})^{-1}X_N\Gamma_N^{-1}X_N)^{-1}X_N^H(\lambda^{[1:N]})^{-1}\Gamma_N^{-1} \right\}
$$

(5.132)

Define the $m \times m$ matrices $P_N = \Gamma_N^{-1}$ and $P_{N+1} = \Gamma_{N+1}^{-1}$. Equation (5.132) becomes

$$
P_{N+1} = (\lambda^{[1:N]})^{-1}P_N \left\{ I_{m \times m} - \beta_N^{[1:N+1]}X_N( I_{p \times p} + \beta_N^{[1:N+1]}(\lambda^{[1:N]})^{-1}X_N^HP_NX_N)^{-1}X_N^H(\lambda^{[1:N]})^{-1}P_N \right\}
$$

(5.133)
Also, define the Kalman gain matrix to be the $p \times m$ matrix

\[
K_N = \left( I_{pxp} + \beta_N^{[1:N+1]}(\lambda^{[1:N]})^{-1}X_N^H P_N X_N \right)^{-1} \sqrt{\beta_N^{[1:N+1]}X_N^H (\lambda^{[1:N]})^{-1}P_N}
\]

\[
= \left( \lambda^{[1:N]} I_{pxp} + \beta_N^{[1:N+1]}X_N^H P_N X_N \right)^{-1} \sqrt{\beta_N^{[1:N+1]}X_N^H P_N}
\]

(5.134)

The matrix $K_N$ is called the Kalman gain matrix because it plays a role similar to the gain factor in a Kalman filter. Equation (5.133) can be re-written as

\[
P_{N+1} = \left( \lambda^{[1:N]} \right)^{-1}P_N \left( I_{m \times m} - \sqrt{\beta_N^{[1:N+1]}X_N K_N} \right)
\]

(5.135)

Moreover, from equation (5.134), it can be written

\[
\left( I_{pxp} + \left( \lambda^{[1:N]} \right)^{-1} \beta_N^{[1:N+1]}X_N^H P_N X_N \right) K_N = \left( \lambda^{[1:N]} \right)^{-1} \sqrt{\beta_N^{[1:N+1]}X_N^H P_N}
\]

(5.136)

Equation (5.136) can in turn be written

\[
K_N = \left( \lambda^{[1:N]} \right)^{-1} \sqrt{\beta_N^{[1:N+1]}X_N^H P_N} \left( I_{m \times m} - \sqrt{\beta_N^{[1:N+1]}X_N K_N} \right)
\]

\[
= \sqrt{\beta_N^{[1:N+1]}X_N^H P_{N+1}}
\]

(5.137)

For the matrix $\Phi_{N+1}$ to be the Least-Square solution when $N + 1$ positions are available, the following need to be verified

\[
\Phi_{N+1} = \Theta_{N+1} P_{N+1}
\]

\[
= \lambda^{[1:N]} \Theta_N P_{N+1} + \beta_N^{[1:N+1]}X_{N+1}^H P_{N+1}
\]

\[
= \lambda^{[1:N]} \Theta_N P_{N+1} + \sqrt{\beta_N^{[1:N+1]}X_{N+1} K_N}
\]

\[
= \Theta_N P_N \left( I_{m \times m} - \sqrt{\beta_N^{[1:N+1]}X_N K_N} \right) + \sqrt{\beta_N^{[1:N+1]}X_{N+1} K_N}
\]

\[
= \Theta_N P_N + \sqrt{\beta_N^{[1:N+1]} \left( X_{N+1} - \Theta_N X_N \right) K_N}
\]

\[
= \Phi_N + \sqrt{\beta_N^{[1:N+1]} \left( X_{N+1} - \Phi_N X_N \right) K_N}
\]

(5.138)
The resulting Recursive Least-Square algorithm works by initially setting $\Phi_N = 0_{m \times m}$ and $P_1 = \delta^{-1} I_{m \times m}$ where $\delta$ is a small value and then iterating on equations (in order) (5.134), (5.138) and (5.135). The similarity between this algorithm and a conventional stochastic approximation algorithm is studied in Appendix E.

5.3 BANDWIDTH OF THE RLS ALGORITHM

The goal of this section is to propose a general setup for tuning the bandwidth of the motion estimator. The rules for changing the bandwidth of the estimator are based on a close monitoring of the a-posteriori errors of the Recursive Least-Square algorithm derived in the previous section.

5.3.1 A-POSTERIORI ERROR

The power of the a-posteriori error produced by the multidimensional recursive least-square algorithm derived in section 5.2 is calculated. This value is used by the motion estimation system to tune its estimation bandwidth. Start with the motion model obtained from $N$ and $N + 1$ consecutive feature positions:

$$
\Phi_N \left( \sum_{i=1}^{i=N-1} \beta_i^{[1:N]} X_i X_i^H \right) = \left( \sum_{i=1}^{i=N-1} \beta_i^{[1:N]} X_{i+1} X_{i+1}^H \right)
$$

$$
\Phi_{N+1} \left( \sum_{i=1}^{i=N} \beta_i^{[1:N+1]} X_i X_i^H \right) = \left( \sum_{i=1}^{i=N} \beta_i^{[1:N+1]} X_{i+1} X_{i+1}^H \right)
$$

Equation (5.139) shows both $\Phi_N$ and $\Phi_{N+1}$ as Least-Square solutions. It was shown in Section 5.2 that the latter can also be obtained from a Recursive Least Square algorithm. At this point, introduce the $m \times p(N - 1)$ matrices $C_N$ and $D_N$ defined by

$$
C_N = \left[ \sqrt{\beta_1^{[1:N]} X_1}, \cdots, \sqrt{\beta_{N-1}^{[1:N]} X_{N-1}} \right]
$$
\[ D_N = \left[ \sqrt{\beta_1^{[1:N]}}, \cdots, \sqrt{\beta_{N-1}^{[1:N]}}, \sqrt{\beta_{N}^{[1:N]}}, \sqrt{\beta_{N+1}^{[1:N]}}, \cdots, \sqrt{\beta_{N+p}^{[1:N]}}, \cdots \right] \] (5.140)

so equation (5.139) can be re-written as

\[ \Phi_N \left( C_N C_N^H \right) = (D_N C_N^H) \]

\[ \Phi_{N+1} \left( C_{N+1} C_{N+1}^H \right) = (D_{N+1} C_{N+1}^H) \] (5.141)

and using equation (3.35),

\[ C_{N+1} = \left[ \sqrt{\beta_1^{[1:N+1]}}, \cdots, \sqrt{\beta_{N-1}^{[1:N+1]}}, \sqrt{\beta_{N}^{[1:N+1]}}, \sqrt{\beta_{N+1}^{[1:N+1]}}, \cdots \right] \]
\[ = \left[ \sqrt{\lambda^{[1:N]}}, \sqrt{\beta_{N+1}^{[1:N+1]}}, \sqrt{\beta_{N+2}^{[1:N+1]}}, \cdots \right] \]

\[ D_{N+1} = \left[ \sqrt{\beta_1^{[1:N+1]}}, \cdots, \sqrt{\beta_{N-1}^{[1:N+1]}}, \sqrt{\beta_{N}^{[1:N+1]}}, \sqrt{\beta_{N+1}^{[1:N+1]}}, \cdots \right] \]
\[ = \left[ \sqrt{\lambda^{[1:N]}}, \sqrt{\beta_{N+1}^{[1:N+1]}}, \sqrt{\beta_{N+2}^{[1:N+1]}}, \cdots \right] \] (5.142)

The a-posteriori error \( \xi_{N+1}^2 \) obtained from the model developed with \( N+1 \) image frames is

\[ \xi_{N+1}^2 = \sum_{i=1}^{i=N} \beta_i^{[1:N+1]} \left( \sum_{j=1}^{j=p} (\xi_i^{(j)} - \Phi_{N+1} \xi_i^{(j)})^H (\xi_i^{(j)} - \Phi_{N+1} \xi_i^{(j)}) \right) \]
\[ = \sum_{i=1}^{i=N} \sum_{j=1}^{j=p} \beta_i^{[1:N+1]} (\xi_i^{(j)} - \Phi_{N+1} \xi_i^{(j)})^H (\xi_i^{(j)} - \Phi_{N+1} \xi_i^{(j)}) \]
\[ = \text{trace} \left\{ (D_{N+1} - \Phi_{N+1} C_{N+1})^H (D_{N+1} - \Phi_{N+1} C_{N+1}) \right\} \]
\[ = \text{trace} \left\{ (D_{N+1} - \Phi_{N+1} C_{N+1}) (D_{N+1} - \Phi_{N+1} C_{N+1})^H \right\} \] (5.143)

Define the \( m \times p \) innovation matrix

\[ \Xi_{N+1} = X_{N+1} - \Phi_N X_N \]
\[ = \left[ \xi_{N+1}^{(1)} - \Phi_N x_N^{(1)}, \cdots, \xi_{N+1}^{(p)} - \Phi_N x_N^{(p)} \right] \]
\[ = \left[ \xi_{N+1}^{(1)}, \cdots, \xi_{N+1}^{(p)} \right] \] (5.144)
so the RLS update equation (5.138) can be re-written

$$\Phi_{N+1} = \Phi_N + \sqrt{\beta_N^{[1:N+1]}} \Xi_{N+1} K_N$$  \hspace{1cm} (5.145)

Using equations (5.145) and (5.142), it is shown in Appendix F that the a-posteriori error power in equation (5.143) is

$$\xi_{N+1}^2 = \lambda^{[1:N]} \xi_N^2 + \beta_N^{[1:N+1]} \lambda^{[1:N]} \text{tr} \left\{ \Xi_{N+1} \left( \lambda^{[1:N]} I + \beta_N^{[1:N+1]} X_N^H P_N X_N \right)^{-1} \Xi_{N+1}^H \right\}$$

$$= \lambda^{[1:N]} \xi_N^2 + \beta_N^{[1:N+1]} \text{tr} \left\{ \Xi_{N+1} \left( I - \sqrt{\beta_N^{[1:N+1]}} K_N X_N \right) \Xi_{N+1}^H \right\}$$ \hspace{1cm} (5.146)

When \( p = 1 \), this result simplifies to

$$\xi_{N+1}^2 = \lambda^{[1:N]} \xi_N^2 + \frac{\lambda^{[1:N]} \beta_N^{[1:N+1]}}{\lambda^{[1:N]} + \beta_N^{[1:N+1]} \Xi_{N+1}^H P_N \Xi_{N+1}} \xi_{N+1}^2$$ \hspace{1cm} (5.147)

The results in equation (5.146) and (5.147) suggest that given the a-posteriori error \( \xi_N^2 \), the weights \( \lambda^{[1:N]} \) and \( \beta_N^{[1:N+1]} \) determine the a-posteriori errors \( \xi_{N+1}^2 \).

5.3.2 BANDWIDTH TUNING

In this section, the value of the a-posteriori errors are used to develop a self-tuning bandwidth capability for the motion estimator.

EXPONENTIAL FORGETTING PROFILE

A forgetting exponential profile is considered first. The forgetting factor is denoted by \( \alpha \). This corresponds to \( \beta_i^{[1:i]} = 1 \) and \( \lambda^{[1:i]} = \alpha \), \( 0 \leq \alpha \leq 1 \). Hence

$$\beta_i^{[1:k]} = \alpha^{k-i} \quad \text{with} \quad i \leq k - 2$$ \hspace{1cm} (5.148)
The a-posteriori errors at times $N+1$ and $N$ are, respectively,

\[
\xi_{N+1}^2 = \sum_{i=1}^{i=N} \alpha^{N-i}(x_{i+1} - \Phi_{N+1} x_i)^H (x_{i+1} - \Phi_{N+1} x_i) \\
\xi_N^2 = \sum_{i=1}^{i=N-1} \alpha^{N-1-i}(x_{i+1} - \Phi_N x_i)^H (x_{i+1} - \Phi_N x_i)
\]

(5.149)

To normalize these quantities, notice that

\[
\sum_{i=1}^{i=N} \alpha^{N-i} = \frac{1 - \alpha^N}{1 - \alpha}
\]

\[
\sum_{i=1}^{i=N-1} \alpha^{N-1-i} = \frac{1 - \alpha^{N-1}}{1 - \alpha}
\]

(5.150)

so the normalized errors, denoted by \(\tilde{\xi}_{N+1}^2\) and \(\tilde{\xi}_N^2\) are, respectively,

\[
\tilde{\xi}_{N+1}^2 = \frac{1 - \alpha}{1 - \alpha^N} \xi_{[1:N+1]^2} \\
\tilde{\xi}_N^2 = \frac{1 - \alpha}{1 - \alpha^{N-1}} \xi_{[1:N]^2}
\]

(5.151)

The equality obtained in equation (5.147) can therefore be re-written as

\[
\frac{1 - \alpha^N}{1 - \alpha} \xi_{N+1}^2 = \alpha - \frac{1 - \alpha^{N-1}}{1 - \alpha} \xi_N^2 + \frac{\alpha}{\alpha + x_N^H P_N x_N} \xi_{N+1}^2
\]

(5.152)

This last equality can be re-written as

\[
\tilde{\xi}_{N+1}^2 = \frac{1 - \alpha^{N-1}}{1 - \alpha^N} \tilde{\xi}_N^2 + \frac{1 - \alpha}{1 - \alpha^N} \frac{\alpha}{\alpha + x_N^H P_N x_N} \xi_{N+1}^2
\]

\[
= \left(1 - \frac{1 - \alpha}{1 - \alpha^N}\right) \tilde{\xi}_N^2 + \frac{1 - \alpha}{1 - \alpha^N} \frac{\alpha}{\alpha + x_N^H P_N x_N} \xi_{N+1}^2
\]

(5.153)

and finally,

\[
\tilde{\xi}_{N+1}^2 - \tilde{\xi}_N^2 = -\frac{1 - \alpha}{1 - \alpha^N} \tilde{\xi}_N^2 + \frac{1 - \alpha}{1 - \alpha^N} \frac{\alpha}{\alpha + x_N^H P_N x_N} \xi_{N+1}^2
\]

\[
= \frac{1 - \alpha}{1 - \alpha^N} \left[ -\tilde{\xi}_N^2 + \frac{\alpha}{\alpha + x_N^H P_N x_N} \xi_{N+1}^2 \right]
\]

(5.154)
To make the normalized error constant \( (\xi_{N+1}^2 = \xi_N^2) \), the following equality needs to be satisfied:

\[
\xi_N^2 = \frac{\alpha}{\alpha + x_N^H P_N x_N} \xi_{N+1}^H \xi_{N+1}
\] (5.155)

Therefore, the forgetting factor should verify the following:

\[
\alpha = \frac{\xi_N^2 x_N^H P_N x_N}{\xi_{N+1}^H \xi_{N+1} - \xi_N^2}
\] (5.156)

The result in equation (5.156) states the condition required for keeping the a-posteriori error power constant. The required forgetting factor \( \alpha \) as defined by equation (5.156) could be compared with the value \( \alpha \) used in the algorithm. If previous values of \( \alpha \) are close to one and the value produced by equation (5.156) is significantly lower than one, then it is reasonable to expect a noticeable change in one or several modes of the motion. In this situation, the bandwidth of the motion estimation system must be made larger. In fact, the lower the value of \( \alpha \) is, the larger the estimation bandwidth must be. Equation (5.156) reveals the relationship between estimation bandwidth and innovation. To satisfy \( \alpha \geq 0 \) (negative memory is not allowed), the innovation must be greater than the normalized a-posteriori error. This can be interpreted as a local persistent excitation condition. Note that bandwidth increases \( (\alpha \) gets smaller) as innovation increases. As the innovation error gets closer to the a-posteriori error, \( \alpha \) becomes very large and the contribution of past data is reinforced.

At this point, it becomes clear that estimation bandwidth can be changed on line to keep a-posteriori error power constant. Now, extend this analysis to the more general stochastic approximation framework.
STOCHASTIC APPROXIMATION

The study presented in Section 5.3.2 is now extended to a stochastic approximation framework. It is shown that this general framework brings the capability for the motion estimator to self-tune its bandwidth from a-posteriori errors. Let

\[ S^{[1:N+1]} = \sum_{i=1}^{i=N} \beta_i^{[1:N+1]} \]  

(5.157)

The recursions on the weights \( \beta_i^{[1:N]} \) dictate that

\[ S^{[1:N+1]} = \beta_N^{[1:N+1]} + \lambda^{[1:N]} \sum_{i=1}^{i=N-1} \beta_i^{[1:N]} = \beta_N^{[1:N+1]} + \lambda^{[1:N]} S^{[1:N]} \]  

(5.158)

Using equation (5.158) to normalize the a-posteriori errors, equation (5.147) can now be re-written as

\[ \left( \beta_N^{[1:N+1]} + \lambda^{[1:N]} S^{[1:N]} \right) \bar{\xi}_{N+1}^2 = \lambda^{[1:N]} S^{[1:N]} \xi_N^2 + \frac{\lambda^{[1:N]} \beta_N^{[1:N+1]} \xi_{N+1}^H \xi_{N+1}}{\lambda^{[1:N]} + \beta_N^{[1:N+1]} \xi_N^H P_N \xi_N} \]  

(5.159)

After a few manipulations, the following equality is obtained

\[ \bar{\xi}_{N+1}^2 - \xi_N^2 = \frac{1}{\beta_N^{[1:N+1]} + \lambda^{[1:N]} S^{[1:N]}} \left\{ \frac{\lambda^{[1:N]} \beta_N^{[1:N+1]} \xi_{N+1}^H \xi_{N+1}}{\lambda^{[1:N]} + \beta_N^{[1:N+1]} \xi_N^H P_N \xi_N} - \beta_N^{[1:N+1]} \xi_N^2 \right\} \]  

(5.160)

Given a value \( \beta_N^{[1:N+1]} \) (usually equal to one in a stochastic approximation framework), and from the equation shown above, it can be realized that to maintain the same level of normalized a-posteriori error, the following must be verified:

\[ \beta_N^{[1:N+1]} \xi_N^2 = \frac{\lambda^{[1:N]} \beta_N^{[1:N+1]} \xi_{N+1}^H \xi_{N+1}}{\lambda^{[1:N]} + \beta_N^{[1:N+1]} \xi_N^H P_N \xi_N} \]  

(5.161)
This can be re-arranged as follows:

\[
\beta_N^{[1:N+1]} \xi_N^{[1:N]} + (\beta_N^{[1:N+1]})^2 \xi_N^{[1:N]} H P_N x_N = \lambda^{[1:N]} \beta_N^{[1:N+1]} \xi_N^{[1:N]} H \xi_{N+1}^{[1:N]} + 1
\]

which leads to

\[
(\beta_N^{[1:N+1]})^2 \xi_N^{[1:N]} H P_N x_N = \lambda^{[1:N]} \beta_N^{[1:N+1]} (\xi_N^{[1:N]} H \xi_{N+1}^{[1:N]} + 1 - \xi_N^{2})
\]

Assuming that \( \beta_N^{[1:N+1]} \neq 0, \)

\[
\frac{\lambda^{[1:N]}}{\beta_N^{[1:N+1]}} = \frac{\xi_N^{[1:N]} H P_N x_N}{\xi_{N+1}^{[1:N]} \xi_{N+1} - \xi_N^{2}}
\]

Denote the inverse of the motion estimator bandwidth by \( \gamma_{N+1}, \) that is \( \gamma_{N+1} = \lambda^{[1:N]} / \beta_N^{[1:N+1]} \). The value of \( \gamma_{N+1} \) must be adjusted according to the equality in equation (5.164) to keep the normalized a-posteriori error constant. It is desired to keep the value of \( \gamma_{N+1} \) within the interval \((0, 1]\). At this point, notice that in equation (5.130), the matrices \( P_k, k = 2, 3, \ldots, N, \) are positive because \( P_1 = \delta^{-1} I_{m \times m} \) was chosen positive. It follows that \( \xi_N^{[1:N]} H P_N x_N \) is a non-negative quantity. Since \( \xi_{N+1}^{[1:N]} \) is also a non-negative quantity, \( \gamma_{N+1} \leq 0 \) whenever the denominator is negative, that is when \( \xi_{N+1}^{[1:N]} \xi_{N+1} < \xi_N^{2} \). To summarize, \( \gamma_{N+1} \leq 0 \) or \( \gamma_{N+1} > 1 \) if

\[
\xi_{N+1}^{[1:N]} \xi_{N+1} < \xi_N^{2} \left( 1 + \frac{\xi_N^{[1:N]} H P_N x_N}{\xi_{N+1}^{[1:N]} \xi_{N+1}} \right)
\]

In this case, the innovation appears too small to excite the filter properly. The value \( \gamma_{N+1} \) brings either amplification of the past data (\( \lambda^{[1:N]} \) tending to infinity) or attenuation of the most recent data (\( \beta_N^{[1:N+1]} \) tending to zero). As a consequence the Kalman gain becomes small. The situation described by equation (5.165) indicates that the amount of new information (innovation) is too low to
keep the given normalized a-posteriori error level constant. On the other hand, 0 ≤ γ_{N+1} ≤ 1 if

$$\xi_{N+1} \leq \frac{\xi^2}{\gamma_{N+1}} (1 + x_N^HP_Nx_N)$$  

(5.166)

In this case the filter is properly locally excited and the amount of new information coming in appears to be adequate. Practically, β_{1:1:N+1} should be set to one and λ_{1:1:N} is set to the value making the normalized a-posteriori error remain constant. However, notice in equations (5.131) and (5.133) that the matrix inversion becomes singular as λ_{1:1:N} gets close to zero. One might view this phenomenon as a sign that the amount of new information (innovation) brought by each new iteration is too high. In this case, the motion estimator is constantly trying to adjust to the instantaneous dynamics governing the motion of the object. To avoid this extreme situation, introduce a lower limit on γ_{N+1}, call it γ_{min} with 0 < γ_{min} < 1. This has the effect of rejecting innovation terms that are too large thereby limiting the amount of information brought by the most recent data. Specifically, the following should be verified:

$$\frac{\xi^2}{\gamma_{min}} (1 + x_N^HP_Nx_N) \leq \xi_{N+1} \leq \frac{\xi^2}{\gamma_{min}} \left( \frac{\gamma_{min} + x_N^HP_Nx_N}{\gamma_{min}} \right)$$  

(5.167)

Clearly, stochastic approximation algorithms offer a framework for monitoring and analyzing the sampling rate locally. The inequality in equation (5.167) can even be generalized further by introducing a maximum value γ_{max} ≥ 1 putting a lower limit on the filter bandwidth:

$$\frac{\xi^2}{\gamma_{max}} \left( \frac{\gamma_{max} + x_N^HP_Nx_N}{\gamma_{max}} \right) \leq \xi_{N+1} \leq \frac{\xi^2}{\gamma_{min}} \left( \frac{\gamma_{min} + x_N^HP_Nx_N}{\gamma_{min}} \right)$$  
or  

$$\frac{\xi^2}{\gamma_{max}} \left( 1 + \frac{x_N^HP_Nx_N}{\gamma_{max}} \right) \leq \xi_{N+1} \leq \frac{\xi^2}{\gamma_{min}} \left( 1 + \frac{x_N^HP_Nx_N}{\gamma_{min}} \right)$$  

or
with $0 < \gamma_{\text{min}} < 1$ and $\gamma_{\text{max}} \geq 1$. If these inequalities are verified, the bandwidth can be determined by using equation (5.164). Practically, if the solution $\gamma$ is greater or equal to one, set $\lambda^{[1:N]} = 1$ and choose $\beta^{[1:N+1]}_N \leq 1$. Likewise, if the solution $\gamma$ is strictly less than one, set $\beta^{[1:N+1]}_N = 1$ and pick the appropriate $\lambda^{[1:N]}$ so that $\lambda^{[1:N]}$ and $\beta^{[1:N+1]}_N$ are always between zero and one. This allows the system to think in terms of forgetting either the past or the present. Observe that the last inequalities in equation (5.168) refer to the ratio $\varepsilon^{[N+1]}_N \varepsilon_{N+1} / \xi^2_N$. This quantity can be viewed as a signal to noise power ratio since the innovation represents the amount to new information and the a-posteriori error represents the noise or uncertainty about the motion model. When this ratio does not satisfy the inequalities in equation (5.168), the a-posteriori error cannot be kept constant. The additional errors must be processed by the translation estimator. A methodology for keeping the signal to noise ratio constant is discussed in Appendix G. As a final remark, compare equation (5.164) and (5.156), and notice the similarity between the exponential and the stochastic approximation case. However, as opposed to the exponential case where the same $\alpha$ is used on both past and present data, the stochastic approximation weights $\lambda^{[1:N]}$ and $\beta^{[1:N+1]}_N$ are used in the $(N + 1)th$ iteration only. Therefore, the result shown in equation (5.164) is more powerful as it allows the system to determine itself the instantaneous bandwidth required to maintain the normalized a-posteriori error constant. These results might provide the key to new possibilities such as adaptively monitoring the sampling rate used by the motion estimator. To
conclude, call $\gamma_{\text{opt}}$ the critical value

$$
\gamma_{\text{opt}} = \frac{\xi_N^2 \varepsilon_N^H P_N \varepsilon_N}{\varepsilon_{N+1}^H \varepsilon_{N+1} - \xi_N^2}
$$

(5.169)

and notice that equation (5.160) can be re-written as

$$
\xi_{N+1}^2 - \xi_N^2 = \frac{(\beta_N^{[1:N+1]})^2}{(\beta_N^{[1:N+1]} + \lambda^{[1:N]} S^{[1:N]}) (\lambda^{[1:N]} + \beta_N^{[1:N+1]})} \left\{ \left( \varepsilon_N^H \varepsilon_{N+1} - \xi_N^2 \right) - \frac{\lambda^{[1:N]}}{\beta_N^{[1:N+1]}} - \xi_N^2 \xi_N^H P_N \varepsilon_N \right\}
$$

(5.170)

With the help of equation (5.169), this last equation becomes

$$
\xi_{N+1}^2 - \xi_N^2 = \frac{(\beta_N^{[1:N+1]})^2}{(\beta_N^{[1:N+1]} + \lambda^{[1:N]} S^{[1:N]}) (\lambda^{[1:N]} + \beta_N^{[1:N+1]})} \left\{ \left( \varepsilon_N^H \varepsilon_{N+1} - \xi_N^2 \right) \left( \frac{\lambda^{[1:N]}}{\beta_N^{[1:N+1]}} - \gamma_{\text{opt}} \right) \right\}
$$

(5.171)

In the case where the innovation term agrees with the bandwidth limitations set in equation (5.168), the ratio $\lambda^{[1:N]} / \beta_N^{[1:N+1]}$ is always positive. It follows that the first fraction in equation (5.171) is always positive. As a consequence, it can be concluded from equation (5.171) that if $\varepsilon_N^H \varepsilon_{N+1} > \xi_N^2$, $\xi_{N+1}^2 > \xi_N^2$ whenever $\lambda^{[1:N]} / \beta_N^{[1:N+1]} > \gamma_{\text{opt}}$. Likewise, $\xi_{N+1}^2 < \xi_N^2$ whenever $\lambda^{[1:N]} / \beta_N^{[1:N+1]} < \gamma_{\text{opt}}$. This observation suggests that in some situations it is possible to reduce normalized a-posteriori error power as opposed to keeping it constant. Such a window of opportunity occurs whenever $\gamma_{\text{opt}} > 1$ (inequality in equation (5.165) verified).

In this case, the normalized a-posteriori error can be simply reduced by choosing $\lambda^{[1:N]} = \beta_N^{[1:N+1]} = 1$. 
Chapter 6

APPLICATIONS

The purpose of this chapter is to bring together the various concepts proposed in Chapters 2 through 5 in particular applications in computer vision, robotics and image compression. These applications demonstrate that the motion model provides a practical framework for understanding and organizing object motions beyond the functionality of early vision systems.

6.1 MOTION ESTIMATION

The first application is concerned with building the general setup for a motion estimation system. The various components of such a system are independently reviewed and then brought together in a general block diagram.

6.1.1 VISUAL MODELS

In addition to the perspective projection occurring in human eyes and in cameras, other linear or non-linear projection might be used to take into account particular subjective properties of the human visual system. In this case, the projection operator $h_k()$ is the result of cascading several projection operators.
One well-known limitation of the human visual system is the lack of peripheral vision. In certain situations, it might be desirable to take this property into account to spatially distribute motion modeling errors. A projection operator incorporating this property is now introduced. The projection operator is inspired from the stereographic projection often used in topology [78] and could be called isotropic (radial symmetry) wide-angle projection. Figure 13 shows such a projection. Under this projection, the viewplane coordinates of a point in a three dimensional space is obtained by first projecting it according to a per-
Since the point is then projected orthogonally onto the plane $X_3 = 0$, the overall projection can be analytically described by

$$y_{ij} = \frac{F_0}{\sqrt{x_1^2 + x_2^2 + (x_3 + F_0)^2}} \begin{bmatrix} x_1 \\ x_2 \\ x_3 + F_0 \end{bmatrix}$$

The Jacobian associated with this projection is

$$J = \frac{F_0}{\left(x_1^2 + x_2^2 + (x_3 + F_0)^2\right)^{3/2}} \begin{bmatrix} j_{1,1} & j_{1,2} & j_{1,3} \\ j_{2,1} & j_{2,2} & j_{2,3} \end{bmatrix}$$

where

$$j_{1,1} = x_2^2 + (x_3 + F_0)^2$$
$$j_{1,2} = -x_1x_2$$
Furthermore, the re-shuffled multidimensional Hessian is

\[
H = \frac{F_0}{(x_1^2 + x_2^2 + (x_3 + F_0)^2)^{5/2}}
\]

where

\[
\begin{align*}
h_{1,1,1} &= -3x_1(x_2^2 + (x_3 + F_0)^2) \\
h_{1,1,2} &= x_2(2x_1^2 - x_2^2 - (x_3 + F_0)^2) \\
h_{1,1,3} &= (x_3 + F_0)(2x_1^2 - x_2^2 - (x_3 + F_0)^2) \\
h_{2,1,1} &= x_2(2x_1^2 - x_2^2 - (x_3 + F_0)^2) \\
h_{2,1,2} &= x_1(-x_1^2 + 2x_2^2 - (x_3 + F_0)^2) \\
h_{2,1,3} &= 3x_1x_2(x_3 + F_0) \\
h_{1,2,1} &= x_1(-x_1^2 + 2x_2^2 - (x_3 + F_0)^2) \\
h_{1,2,2} &= 3x_1x_2(x_3 + F_0) \\
h_{2,2,2} &= -3x_2(x_1^2 + (x_3 + F_0)^2) \\
h_{2,2,3} &= (x_3 + F_0)(-x_1^2 + 2x_2^2 - (x_3 + F_0)^2) \\
h_{1,3,3} &= x_1(-x_1^2 - x_2^2 + 2(x_3 + F_0)^2) \\
h_{2,3,3} &= x_2(-x_1^2 - x_2^2 + 2(x_3 + F_0)^2)
\end{align*}
\]
and $h_{i,j,k} = h_{i,k,j}$ for $1 \leq i \leq 2$, $1 \leq j, k \leq 3$. Camera lenses based on this wide-angle projection have been designed for stereoscopic computer vision systems [49]. Isotropic wide-angle projection is an approximation of the conformal mapping occurring in the human visual system whereby retinal images are transformed into cortical projections [104]. This mapping can be described by the following transformation. Given a point with coordinates $[x_1, x_2, x_3]^H$, calculate the polar coordinates $(r, \theta)$ of its perspective projection on the plane $X_3 = 0$.

$$
\begin{align*}
  r &= \sqrt{x_1^2 + x_2^2} \frac{F_0}{x_3 + F_0} \\
  \theta &= \arctan \left( \frac{x_2}{x_1} \right)
\end{align*}
$$

Equation (6.178) expresses the retinal (view plane) polar coordinates of an image feature. The cortical projection coordinates $(\rho, \psi)$ are then obtained by applying a logarithmic transformation to the radial coordinate:

$$
\begin{align*}
  \rho &= \ln (r) = \frac{1}{2} \ln \left( x_1^2 + x_2^2 \right) + \ln (F_0) - \ln (x_3 + F_0) \\
  \psi &= \theta = \arctan \left( \frac{x_2}{x_1} \right)
\end{align*}
$$

Equation (6.179) shows that as in the case of isotropic wide-angle projection, the spacing between samples is the largest at the center of the image and then gets progressively smaller away from the center. Clearly the two non-linear projections discussed in this section provide the ability to approximate the functionality of the human visual system. These projection operators can therefore be incorporated in a motion estimation system to bring subjective weighting in agreement with the limitations of the human visual system. The effect of these new projection operators will be to emphasize on the motion quality at the center of the image while developing a looser understanding of the motion about the borders of the field of view. To conclude, it is worth mentioning that linear
approximations of perspective projection have been proposed [6]. These approximations can be useful to develop motion understanding in an intermediary plane.

6.1.2 FEATURE MATCH VALIDATION

Another important feature of the multidimensional motion model is its ability to perform on-line quality verification of the feature correspondences. This ability comes from the fact that matching and motion model estimation can be simultaneously performed as follows. Given a set of N features in each of two consecutive image frames, run one motion model with \( p = N \) for each of the \( N! = N(N - 1) \ldots 3.2.1 \) possible matches. The motion bandwidth analysis performed in Chapter 5 suggests that a-posteriori errors, innovation terms or estimation bandwidth are all quantities that can be used to monitor the quality of the feature matches. False matches can be eliminated by simply comparing the a-posteriori errors they produce against a large threshold value. By using smaller and smaller threshold values in the next image frames, the number of motion models running in parallel can be progressively reduced from \( N! \) to a smaller number. This number is one if there is no motion ambiguity. In general, the final threshold value must be chosen to reflect a particular level of confidence in the feature correspondences.

6.1.3 OCCLUSIONS

The number of features \( p \) that can be concurrently tracked by the motion model can be instantaneously increased or decreased. Imagine the case where the motion model uses \( p \) distinct feature matches of the same rigid object. Furthermore, imagine that this moving object is being progressively occluded by another mov-
ing object. Clearly, the number of features tracked by the motion model gets smaller and smaller as the object gets more and more occluded. Two possibilities are offered to handle this situation. The first one is to go back to the motion parameters estimation algorithms presented in Chapter 3 and 4 and realize that the feature count \( p \) can vary from one sampling instant to another. The dimensionality of the motion model can therefore be gradually decreased until all the corresponding features are occluded. Another possibility is to replace the missing matches with remaining valid matches still tracked by the motion model. Such an approach results in making the motion estimation algorithm depend more heavily on those duplicated matches. The former possibility is to be preferred as it takes full advantage of the ability of the motion estimation algorithm to change its dimensionality on line. This is an important property as it allows a gradual degradation or recovery of the motion model performance around occlusions. When there is no matching feature left, the motion model can be kept running free so it can attempt to predict the location in the view field where the object may reappear.

### 6.1.4 GENERAL SCHEME

Figure 14 shows the block diagram of a motion estimation system based on the multidimensional framework proposed in chap 2.

In Figure 14, early vision functions such as optical flows and feature correspondences provide information about the feature positions \( Y_{k+1} \) in the viewplane. The matrix \( Y_{k+1} \) along with camera motion parameters, camera focal distance and observability conditions are then used to estimate the transition matrix \( \Phi_{k+1} \). Constraints map the resulting matrix into a given desired class of motion (for example, rotation as seen in Chapter 3). These motion parameters are then
Figure 14. MOTION ESTIMATION SYSTEM
sent to the translation estimator which also reconstructs the filtered positions $\hat{X}_{k+1}$. Both the transition matrix and translation vector estimators use the same projection operator which is either the standard perspective projection or the isotropic wide-angle projection discussed in Section 6.1.1. The information produced by the two motion parameter estimators is then used by the Recursive Least-Square algorithm presented in Chapter 5. An estimation bandwidth diagnosis is performed and the required bandwidth for next iteration is relayed to the motion parameter estimators. Meanwhile, the projected filtered positions are compared with the positions provided by the early vision system. Feature associations yielding large differences are eliminated by the motion estimation system and as a result, the dimensionality $p$ of the model is decreased. As mentioned in Section 6.1.2, this situation can occur in presence of occlusions or noisy/ambiguous conditions.

6.2 MOTION WHITENING

Another application is to use the motion model to make the camera track the slow-varying modes of the motion while leaving out unpredictable motion modes. As shown in Figure 15, this is accomplished by using the transition matrix estimator to determine the rotation matrix $O_{k+1}$ which best describes the motion of the object. This can be done by projecting the matrix $\Phi_{k+1}$ onto the class of orthogonal matrices as discussed in Section 3.5. Once this is done, an estimate of the next rotation, $\hat{O}_{k+2}$, is computed. This rotation matrix is used to rotate the camera before reading the feature positions $Y_{k+2}$. This assumes of course that the camera plane is free from rotating with respect to any rotation axis going through the origin. Likewise, the transition matrix $\Phi_{k+1}$ is used to determine an estimate of the focal distance to be used by the camera to acquire $Y_{k+2}$. 
Figure 15. MOTION WHITENING SYSTEM

This can be done for example by analyzing the determinant of the matrix $\Phi_{k+1}$. The task performed by the system in Figure 15 is to extract the unpredictable modes of an object motion. The output of this system is a residual motion which can be used at later stages for various purposes such as extraction of vibrations. The motion whitening system might also be useful to make a motion analysis system operate in a stochastic framework where the motion residuals can be assumed to be statistically uncorrelated or independent.
6.3 FRACTIONAL INTERPOLATION

An important issue in video processing is the conversion of television standards. The number of frames can vary from one country to another so temporal interpolation or decimation is required to go from one standard to another. To perform these conversions, the positions of the object must be determined at fractional instants of the sampling period. It is therefore necessary to derive a methodology based on the motion model to determine the objects position in the interpolated image frames. This problem can be formulated as follows. Given \( p \) matching positions \( \mathbf{X}_{k+1} \) and \( \mathbf{X}_k \) and given the motion model

\[
vec(\mathbf{X}_{k+1}) = (I_{p \times p} \otimes \Phi_{k+1}) vec(\mathbf{X}_k) + (1 \otimes I_{m \times m}) T_{k+1} \tag{6.180}
\]

interpolate the object positions at \( q + 1 \) instants between sampling time \( k \) and \( k+1 \). To solve this problem, denote the interpolated positions by \( \mathbf{W}_j \), \( 0 \leq j \leq q \). The instants at which the \( \mathbf{W}_j \)'s are determined are assumed to be uniformly spaced. The interpolated positions are such that \( \mathbf{W}_0 = \mathbf{X}_k \) and \( \mathbf{W}_q = \mathbf{X}_{k+1} \). Derive a time-invariant motion model for the \( \mathbf{W}_j \)'s by taking the \( q \)th root of the matrix \( \Phi_{k+1} \) so that

\[
vec(\mathbf{W}_0) = vec(\mathbf{X}_k)
\]

\[
vec(\mathbf{W}_1) = (I_{p \times p} \otimes \Phi_{k+1}^{1/q}) vec(\mathbf{X}_k) + (1 \otimes I_{m \times m}) D_{k+1}
\]

\[
vec(\mathbf{W}_2) = (I_{p \times p} \otimes \Phi_{k+1}^{1/q}) vec(\mathbf{W}_1) + (1 \otimes I_{m \times m}) D_{k+1}
\]

\[
= (I_{p \times p} \otimes \Phi_{k+1}^{2/q}) vec(\mathbf{W}_0)
\]

\[
+ (1 \otimes \Phi_{k+1}^{1/q}) D_{k+1} + (1 \otimes I_{m \times m}) D_{k+1}
\]

\[\vdots \]

\[
vec(\mathbf{W}_q) = (I_{p \times p} \otimes \Phi_{k+1}) vec(\mathbf{W}_0) + \sum_{i=0}^{q-1} \left(1 \otimes \Phi_{k+1}^{i/q}\right) D_{k+1}
\]

\( \tag{6.181} \)
Take the last equality in equation (6.181) and use the fact that \( W_0 = X_k \) and \( W_q = X_{k+1} \), to obtain

\[
 vec(X_{k+1}) = (I_p \otimes \Phi_{k+1}) vec(X_k) + (I \otimes I_{m \times m}) + \sum_{i=0}^{q-1} \Phi_{k+1}^{i/q} D_{k+1}
\]

(6.182)

Upon comparing equation (6.182) with equation (6.180), it is clear that the vector \( D_{k+1} \) should be chosen as:

\[
 D_{k+1} = \left( \sum_{j=0}^{q-1} \Phi_j^{j/q} \right)^{-1} T_{k+1}
\]

(6.183)

if the inverse exists. Assume from now on that the matrix \( \Phi_{k+1} \) is diagonalizable (the algebraic multiplicity of an eigenvalue is equal to its geometric multiplicity, the dimension of the subspace spanned by the eigenvectors associated with this eigenvalue [8]). Furthermore, assume that the real eigenvalues are strictly positive. The \( q \)th root of the matrix \( \Phi_{k+1} \) can be obtained from its eigenvalues and eigenvectors by considering the following similarity transform

\[
 \Phi_{k+1} = U_{k+1} \Lambda_{k+1} U_{k+1}^{-1}
\]

(6.184)

where \( U_{k+1} \) is the \( m \times m \) change of basis matrix made of the eigenvectors of \( \Phi_{k+1} \) and \( \Lambda_{k+1} \) is the \( m \times m \) diagonal matrix made of the eigenvalues of \( \Phi_{k+1} \). Since \( \Phi_{k+1} \) is real, its eigenvalues are all real or occur in conjugate pairs of equal magnitude. The \( i/q \)th power of the matrix \( \Phi_{k+1} \) is

\[
 \Phi_{k+1}^{i/q} = U_{k+1} \Lambda_{k+1}^{i/q} U_{k+1}^{-1}
\]

(6.185)

for \( 1 \leq i \leq q \). In equation (6.185), the \( i/q \)th root of each individual eigenvalue is obtained from its polar coordinates by computing the \( i/q \)th power of the
magnitude and the $i/q$th fraction of the minimum phase. A substitution of equation (6.185) into equation (6.183) yields:

\[ D_{k+1} = U_{k+1} \left( \sum_{j=0}^{q-1} A_{k+1}^{j/q} \right)^{-1} U_{k+1}^{-1} T_{k+1} \]  

Equations (6.185) and (6.186) along with the equation

\[ vec(W_j) = (I_{p \times p} \otimes \Phi_{k+1}^{j/q}) vec(W_0) + \sum_{i=0}^{j-1} (I \otimes \Phi_{k+1}^{i/q}) D_{k+1} \]  

where $2 \leq j \leq q$, are the key to temporal decimation and interpolation of object motion. However, due to the time-varying nature of the motion model, interpolation of object motion must be carefully handled and analyzed on a case by case basis. To see this, take the problem of decimating a sequence of six frames into five frames covering the same time interval. Denote the six successive sets of positions in the first sequence by $X_i$, $1 \leq i \leq 6$. The motion parameters describing the motion of these features are $\Phi_{i+1}$ and $T_{i+1}$, $1 \leq i \leq 5$. Call $W_i$, $0 \leq i \leq 4$, the five interpolated positions. $W_0$ and $W_4$, the positions first and last interpolated frames, coincide with $X_1$ and $X_6$, respectively. $W_1$, the second set of interpolated positions, lies in an image frame a quarter of the time between the frame containing $X_2$ and the frame containing $X_3$. $W_2$, the third set of interpolated positions, occurs at half way between $X_3$ and $X_4$. Finally, $W_3$, the fourth set of interpolated positions, can be found in an image frame at three quarters of the time between $X_4$ and $X_5$. Figure 16 provides an illustration, where the object being tracked is a black dot.

From Figure 16, it can be seen that

\[ vec(W_0) = vec(X_1) \]

\[ vec(W_1) = (I_{p \times p} \otimes \Phi_{3}^{1/4}) vec(X_2) + D_3 \]
Figure 16. FRACTIONAL MOTION INTERPOLATION
where

\[ D_{i+1} = \left( \sum_{j=0}^{3} \Phi_{i+1}^{j/q} \right)^{-1} T_{i+1} \]  

(6.189)

for \( 1 \leq i \leq 4 \). Notice that the positions \( X_5 \) are not used at all. This follows from the fact that the motion model has been built looking toward the future only. When the transition matrix is not diagonalizable or at least one of its real eigenvalues is negative or null, the eigenvalue decomposition shown in equation (6.184) can be replaced by a singular value decomposition

\[ \Phi_{k+1} = U_{k+1} \begin{bmatrix} \Sigma_{k+1} & 0_{r \times (m-r)} \\ 0_{(m-r) \times r} & 0_{(m-r) \times (m-r)} \end{bmatrix} V_{k+1}^H \]  

(6.190)

where \( U_{k+1} \) and \( V_{k+1} \) are both unitary matrices and \( \Sigma_{k+1} \) is a square diagonal matrix made of the nonzero singular values of the matrix \( \Phi_{k+1} \). In this case, the \( i/q \)th power of the matrix \( \Phi_{k+1} \) can be identified with the matrix

\[ \Phi_{k+1}^{i/q} = U_{k+1} \begin{bmatrix} \Sigma_{k+1}^{i/q} & 0_{r \times (m-r)} \\ 0_{(m-r) \times r} & 0_{(m-r) \times (m-r)} \end{bmatrix} V_{k+1}^H \]  

(6.191)

The translation vector \( D_{k+1} \) can then be calculated with the help of equation (6.186) modified as follows:

\[ D_{k+1} = V_{k+1} \begin{bmatrix} \left( \sum_{j=0}^{q-1} \Sigma_{k+1}^{j/q} \right)^{-1} & 0_{r \times (m-r)} \\ 0_{(m-r) \times r} & 0_{(m-r) \times (m-r)} \end{bmatrix} U_{k+1}^H T_{k+1} \]  

(6.192)

The object positions in the interpolated image frames can then be determined by using equation (6.181).
6.4 PLANAR MOTION

Another application of the motion model is to identify the camera orientation providing the best observability conditions. For the case of orthogonal projection ($\ell = 2, \ m = 3$), the camera orientation must be such that the motion of the object is as planar as possible. The equation of the best viewplane is now derived in the case $m = 3, \ell = 2$. Let $z_i^{(j)}$ denote the position of the $j$th, $1 \leq j \leq p$, feature in the image frame $i, 1 \leq i \leq N$. The vectors $z_i^{(j)}$ are $m$ dimensional vectors. Let $n$ denote a unit length vector perpendicular to the image plane. Therefore

$$n^H z_i^{(j)} = 0 \quad \text{for} \quad 1 \leq i \leq N \quad \text{and} \quad 1 \leq j \leq p \quad (6.193)$$

The feature positions $x_i^{(j)}$ reconstructed by the adaptive motion model are such that the vectors $x_i^{(j)} - z_i^{(j)}$ are perpendicular to the viewplane. The best plane orientation is therefore given by the vector $n_0$ minimizing the distance between the $x_i^{(j)}$'s and the viewplane. In other words,

$$n_0 = \min_n \left\{ \sum_{i=1}^{N} \sum_{j=1}^{p} \left( n^H (x_i^{(j)} - z_i^{(j)}) \right)^2 \right\} \quad (6.194)$$

subject that $n_0^H n_0 = 1$ (unit vector length). In light of equation (6.193), equation (6.194) can be re-written as

$$n_0 = \min_n \left\{ \sum_{i=1}^{N} \sum_{j=1}^{p} (x_i^{(j)})^H n_0 H x_i^{(j)} \right\}$$

$$= \min_n \left\{ n^H \left( \sum_{i=1}^{N} \sum_{j=1}^{p} (z_i^{(j)} x_i^{(j)})^H \right) n \right\} \quad (6.195)$$
subject to the constraint $u^H u = 1$. Adding this constraint to equation (6.195), the problem becomes to find $u_0$ minimizing

$$u_0 = \min_{\tilde{u}} \left\{ \frac{u^H \left( \sum_{i=1}^{N} \sum_{j=1}^{p} x_i^{(j)} (x_i^{(j)})^H \right) u}{u^H u} \right\}$$

(6.196)

This can be recognized as a minimization of the Rayleigh quotient associated with the $m \times m$ matrix $M = \sum_{i=1}^{N} \sum_{j=1}^{p} x_i^{(j)} (x_i^{(j)})^H$. An extremum of the Rayleigh quotient are reached whenever $u$ is an eigenvector of the matrix $M$ [48]. Consequently, the vector $u_0$ must be chosen to be the eigenvector associated with the smallest eigenvalue of $M$. In the case $N = 2$ (two consecutive image frames), and using the motion model $x_{i+1}^{(j)} = \Phi_{i+1} x_i^{(j)} + T_{i+1}^{(j)}$, this result becomes

$$u_0 = \text{Eigenvector associated with minimum eigenvalue of}$$

$$\left\{ \sum_{j=1}^{p} (x_i^{(j)} (x_i^{(j)})^H + \Phi_{i+1} x_i^{(j)} (x_i^{(j)})^H \Phi_{i+1}^H + 2\Phi_{i+1} x_i^{(j)} T_{i+1}^H) \right\}$$

(6.197)

Equation (6.197) determines the best camera orientation for the $x_1^{(j)}$'s and $x_2^{(j)}$'s, $1 \leq j \leq p$. This formula can be modified to predict the best camera orientation for capturing the next image frame by simply replacing $\Phi_{i+1}$ and $T_{i+1}$ with predicted motion parameters $\hat{\Phi}_{i+1}$ and $\hat{\Phi}_{i+1}$. 
Chapter 7

SIMULATIONS

7.1 SYNTHETIC MOTION

The performance of the motion model estimation algorithms is now tested with object motion taken from Newtonian dynamics of particles. We start with a helicoidal motion generating a spiral in the projection plane. Note that this motion can also be viewed as the perspective projection of a screw displacement with a rotation axis perpendicular to the projection plane. The motion trajectory (dashed line) shown in Figure 17 is generated by

\[
\mathbf{x}_{k+1} = \begin{bmatrix}
  a \cos(\theta) & a \sin(\theta) & 0 \\
  -a \sin(\theta) & a \cos(\theta) & 0 \\
  0 & 0 & 1
\end{bmatrix} \mathbf{x}_k
\]

(7.198)

with \( a = 0.99 \) and \( \theta = 0.05 \). The adaptive model-based motion estimation algorithm developed in Chapter 3 and 4 was used on this data for the case of a time invariant perspective projection. Polar coordinates are used to describe the consecutive positions of the point. Given position \([x_1, x_2, x_3]^H\), let \( R = \sqrt{x_1^2 + x_2^2} \) and \( \Psi = \arctan(x_2/x_1) \). The polar coordinates of the projected point, denoted
by the vector \([r, \Theta, 0]^H\) are

\[
\begin{bmatrix}
  r \\
  \Theta \\
  0
\end{bmatrix} = h \begin{bmatrix}
  R \\
  \Psi \\
  x_3
\end{bmatrix} = \begin{bmatrix}
  \frac{F_0 R}{x_3 + F_0} \\
  \Psi \\
  0
\end{bmatrix}
\]

(7.199)

The \(2 \times 3\) Jacobian matrix and the \(6 \times 3\) re-shuffled Hessian matrix used in the motion model estimators are

\[
J = \begin{bmatrix}
  \frac{F_0}{x_3 + F_0} & 0 & -\frac{F_0 R}{(x_3 + F_0)^2} \\
  0 & 1 & 0 \\
  0 & 0 & 1
\end{bmatrix} = \begin{bmatrix}
  \frac{F_0}{x_3 + F_0} & 0 & -\frac{F_0}{x_3 + F_0} \\
  0 & 1 & 0 \\
  0 & 0 & 1
\end{bmatrix}
\]

(7.200)

and

\[
H = \begin{bmatrix}
  0 & 0 & \frac{-F_0}{(x_3 + F_0)^2} \\
  0 & 0 & 0 \\
  0 & 0 & 0 \\
  0 & 0 & 0
\end{bmatrix}
\]

\[
\begin{bmatrix}
  0 & 0 & \frac{2RF_0}{(x_3 + F_0)^3} \\
  \frac{-F_0}{(x_3 + F_0)^2} & 0 & 0 \\
  0 & 0 & 0 \\
  0 & 0 & 0
\end{bmatrix}
\]

(7.201)

The solid line in Figure 17 shows the trajectory generated by the estimated transition matrices alone. The focal distance is \(F_0 = 1\) and the initial depth of the point is \(x_3 = 50.0\). The depth was then progressively increased from sample to sample to make the motion estimator track a three dimensional screw displacement. This displacement correspond to the point rotating and moving away from the projection plane. Figure 18 shows the trajectory produced by both the transition matrices and the translation vectors (solid line) against the original point trajectory (dashed line). The forgetting profile was \(\lambda^{[1:4]} = 0.9\) and \(\beta_k^{[1:k+1]} = 1\). In the transition matrix estimator, the matrix \(P\) is periodically reset to keep the algorithm sensitive to a new trend [43]. Both the transition
matrix and the translation vector estimators include a mechanism to unwrap the phase $\Psi$ along the trajectory.

A more complex motion is chosen to put to test the time-varying motion model estimator. The complexity of the motion is increased by the addition of discontinuities in the trajectory. Consider a swinging mass $m$ attached to a slowly oscillating spring as shown in Figure 19. The motion of the mass is restricted by two walls placed at angle $\pm \Theta_0$ from the horizontal plane. The force applied to the mass is the result of a gravitational force $\mathcal{F}_g$ directed downward and a restoring force $\mathcal{F}_r$ proportional to the distance $r$ of the mass to the origin and directed along the spring. In short, $\mathcal{F}_g = mg$ where $g$ is the gravitational acceleration constant and $\mathcal{F}_r = -kr$ where $k$ is the constant of the spring.

Let $l_0$ be the length of the spring at equilibrium. In this case $\mathcal{F}_g = -\mathcal{F}_r$ and it follows that

$$mg = kl_0$$

(7.202)
Figure 18. HELICOIDAL MOTION MODELED WITH $\Phi$ and $T$

Figure 19. OSCILLATING MASS
Away from the equilibrium point, the force $\mathbf{F}$ acting on the mass is

$$\mathbf{F} = \mathbf{F}_g + \mathbf{F}_r$$

$$\begin{bmatrix} \mathbf{F}_x \\ \mathbf{F}_y \end{bmatrix} = \begin{bmatrix} m \ddot{x} \\ m \ddot{y} \end{bmatrix} = \begin{bmatrix} -kx \\ -k(y - l_0) \end{bmatrix}$$

(7.203)

where $\mathbf{F}_x$ and $\mathbf{F}_y$ denote the components of $\mathbf{F}$ along the horizontal and vertical axis, respectively. The solution to the second order differential system of equations in (7.203) is

$$x(t) = A \cos(w_0 t + \varphi)$$

$$y(t) = B \cos(w_0 t + \psi) + l_0$$

(7.204)

where the $w_0 = \sqrt{m/k}$. Equation (7.204) describes a harmonic oscillation in each of the two directions. The resulting trajectory is therefore an ellipse. The parameters $A$, $B$, $\varphi$ and $\theta$ are determined by the initial position $[x_0, y_0]^H$ and velocity $[\dot{x}_0, \dot{y}_0]^H$ of the mass:

$$\varphi = \arctan \left( -\frac{\dot{x}_0}{x_0 w_0} \right)$$

$$\psi = \arctan \left( -\frac{\dot{y}_0}{(y_0 - l_0) w_0} \right)$$

$$A = \frac{x_0}{\cos(\varphi)}$$

$$B = \frac{y_0 - l_0}{\cos(\psi)}$$

(7.205)

The dashed line in Figure 20 shows the trajectory of the oscillating mass with $x_0 = \cos(\theta_0)$, $y_0 = -\sin(\theta_0)$, $\dot{x}_0 = 0.01$, $\dot{y}_0 = 0.02$, $l_0 = 0$ and $w_0 = 0.01$. The collision of the mass against the two walls is assumed elastic so the kinetic energy is preserved. At collision time, new initial conditions are determined from the vector tangent to the trajectory.
Figure 20. OSCILLATIONS MODELED WITH $\Phi$

Figure 21. OSCILLATIONS MODELED WITH $\Phi$ and $T$
Figure 20 shows the motion modeled by the transition matrices alone (solid line). The three dimensional motion was induced by making the mass move toward the projection plane. The coordinates of the mass where recorded in polar form, so the Jacobian and Hessian shown in equation (7.200) and (7.201) are used again here. Notice that transition matrices have difficulty tracking the fast motion modes. Figure 21 shows the motion modeled by both the transition matrices and the translation vector. Notice that with the addition of the translation vector, the motion model is able to model the motion of the mass accurately.

7.2 DIGITAL VIDEO SEQUENCE

The second example illustrates the motion estimation of a ping-pong ball. The data is taken from a digital PAL sequence sampled at 13.5MHz. Some of the odd or even luminance fields of this 250 fields long sequence are shown in Figure 23 through 28. Figure 22 shows the trajectory of the center of the ping-pong ball on the camera projection plane. For clarity sake, time was added as the third dimension in the plot.

Figure 24 corresponds to the instant where the ball bounces off the paddle while Figure 25 shows the positions of the ball when it reaches its vertex. It follows that the motion of the ball varies from deceleration in Figure 23 and 25 to acceleration in Figure 24. Notice that there is a slight backward zoom from Figure 23 to Figure 25. Figure 26 shows the player making a serve. Figure 27 and 28 show snapshots of the ball subject to fast motion. Figure 27 shows the ball bouncing off the table and Figure 28 shows the ball passing over the net. Notice the blur on the ball due to the integration time in the camera.
The coordinates of the ball were taken at five points: the upper, lower leftmost and righmost edge plus the center of the ball. The focal distance was taken to be $F_0 = 1$ and the depth of the ball was estimated by setting its initial value to $x_3 = 10.0$ and then changing it according to measured diameter variations. The projection used was a linear projection on the plane $x_3 = 0$. The recursive estimation of the $3 \times 3$ transition matrix $\Phi$ was therefore based on equations (3.47) and (3.48) with $p = 5$ (5 points). The algorithm included an additional feature whereby the matrix $P_{N+1}$ was reset whenever the magnitude of the innovation vector $\varepsilon$ exceeded a preset threshold. The translation vectors were estimated according to equations (4.88), (4.91) and (4.92). Figure 29 shows the results of building the time-varying motion model and then using it to filter the positions. The filtered trajectory (solid lines) in Figure 29 follow closely the original trajectory (dashed lines). To magnify the small differences between the two trajectories, only the results for the first 32 fields were plotted. The

**Figure 22. PING-PONG BALL TRAJECTORY**
discontinuity shown in Figure 29 corresponds to frame shown in Figure 24 where the ball bounces off the paddle. This was one of the few instances where the innovation was large enough to trigger resetting of the matrix $P_{N+1}$. Figure 30 shows the results of applying the motion model in a first order prediction mode. This means that given $\Phi_i$ and $T_i$, the predicted value was $\hat{x}_{i+q} = \Phi_i^q \hat{x}_i + \sum_{k=0}^{q-1} \Phi_i^k T_i$, with $q = 1$. As expected, the prediction (solid line) fails at the discontinuity as the motion model predicts the ball to fall further down when it actually bounces off the paddle. Figure 31 shows the corresponding picture. The first order predicted position of the ball center is shown as a white cross. Higher order predictions work when the motion trajectory is smooth enough. Figure 32 shows various order predicted positions ($q = 2, 4, 6, 7, 8$) when the ping-pong ball is close to its vertex. The second order predicted position is shown as a black cross and the other predicted values can be seen to stack up as the order of the prediction increases. This comes from the fact that the motion
model has captured the ball dynamics, where the magnitude of the ball velocity decreases with time as the ball reaches the vertex. Figure 33 shows predicted positions of the ball at a time in the sequence where the velocity of the ball is very large. The black cross shows the second order predicted position \( (q = 2) \) which corresponds closely to the actual position of the ball as it passes over the net. The other white crosses correspond from right to left to predictions of order \( q = 4,5 \) and \( q = 6 \), respectively. The positions of these crosses can be explained by the fact that past motion parameters include dynamics of the ball rising over the net after it bounces off the table (see Figure 27). The prediction becomes unacceptable past the fifth order.
Figure 31. PREDICTION AT FRAME 7 - FIELD 2

Figure 32. PREDICTION AT FRAME 13 - FIELD 1

Figure 33. PREDICTION AT FRAME 69 - FIELD 2
8.1 CONTRIBUTION

The motion model introduced in this work offers the possibility to build a new generation of motion estimation algorithms for implementing the medium-level vision functionality in the human visual system. In Chapter 1, the three existing categories of motion estimation algorithms were discussed. It was discussed that none of them provides a satisfactory framework for estimating long range motion of objects over several frames. To fill this gap, a multi-dimensional time-varying, motion model was introduced in Chapter 2. This model allows a variable set of object features/points to share a common model thereby providing a tool for early motion understanding and reasoning, a functionality not provided by early vision mechanisms such as the estimation of optical flow or displacement fields. One of the most valuable characteristic of the model lies in its ability to cover a wide range of motions. This is made possible by the presence of two complementary sets of time-varying motion parameters, the transition matrix and the translation vector, designed for capturing slow and fast motion modes, respectively. The recursive estimation of the transition matrix was discussed
in Chapter 3. Both linear and non-linear projection of the image points onto the viewplane were discussed. The resulting algorithm is a multidimensional extended Kalman filter in which the forgetting profile is driven by a stochastic approximation. The algorithm for estimating a transition matrix is the result of the previous transition matrix plus a motion innovation term. The minimum number of feature associations required to identify the transition matrix was shown to be equal to five in the case of three dimensional motion projected on a two dimensional viewplane. Likewise, the minimum number of image frames required to observe the initial position of an object feature was also computed. Due to the lack of general results on observability in non-linear systems, various methods such as depth from zooming or observability at infinity are proposed. Constraints on the transition matrix were also studied. In particular, a new method was proposed to determine the parameters of the closest rotation matrix. This was achieved by considering the skew-symmetric decomposition of a rotation matrix. The estimation of the translation was discussed in Chapter 4. The estimation technique is a multidimensional tracking algorithm that allows the translation vector to act as a slack variable for capturing the remaining motion modes. Conditions for avoiding velocity aliasing were derived. It was shown that a velocity vector does not alias when its magnitude lies in a closed domain determined by the spatial frequency of the feature being tracked and by the spatial and temporal sampling grid. It was shown in Chapter 5 that the problem of estimating translation vectors is dual to the problem of estimating transition matrices, just as controllability is dual to observability in linear system theory. An adaptive policy was proposed to dynamically allocate the estimation bandwidth among these two sets of motion parameters. This technique is based on forcing the motion model to operate at a constant level of error power or at
the same signal to noise ratio. Several domains of applications for the motion model were reviewed in Chapter 6. In the general case of motion estimation, it was suggested that a wide-angle, non-linear projection be substituted for the perspective projection occurring in cameras and the human vision system. The purpose of this fictitious projection is to lay a subjectively pleasing weighting on top of the arbitrary $l^2$ minimizations carried out in the model estimation. It was also suggested that the ability of the motion model to track a variable count of features at any given time be taken advantage of in situations such as occlusions and appearing/disappearing ambiguities. This is an important property which gives a chance to the motion model to break gracefully or to improve smoothly. Besides motion estimation, other applications such as motion whitening and motion interpolation were discussed. The results obtained for the latter suggest that the dynamics captured by the transition matrices and the translation vectors can still be used in motion interpolation/decimation. This provides a noticeable departure from current motion interpolation schemes which merely scale up or down vectors in a displacement field [63]. Explicit examples were provided in Chapter 7. The model parameter estimators were tested on both computer generated and sampled real motions. It was shown that the model performs well for filtering and predicting object trajectory in both slow and fast motion modes. Because of its flexibility, many extensions can be brought to the model. The next section lists the most promising ones.

8.2 FUTURE WORK

In this section, future extensions for the motion model are presented. Applications of the model beyond its original deterministic setup are then proposed.
8.2.1 HIGHER MODEL ORDERS

The motion model considered so far establishes a relationship between the feature positions in two consecutive image frames. Note that this does not preclude the fact that the image frames might have been acquired at non-uniformly separated instants. However, in certain situations, the motion model might need to be extended to include more positions from the past. In this case, the motion model must take the following general form

\[
\begin{align*}
vec(X_{k+1}) &= \sum_{i=0}^{M-1} \left( (I_{p\times p} \otimes R_{k+1-i} \Phi_{k+1-i}) vec(X_{k-i}) \right) + (1 \otimes R_{k+1}) T_{k+1} \\
vec(Y_k) &= vec\left( \begin{bmatrix} h_k(x_k(1)), \ldots, h_k(x_k(p)) \end{bmatrix} \right)
\end{align*}
\]

(8.206)

The effect of this extension is the ability to model faster motion modes with the transition matrices and to help the translation vector be less subject to aliasing. However, this model is not without potential problems. One of them deals with the selection of the model order \( M \). Sudden cuts are frequent in video sequences and the motion model in equation (8.206) must contain a provision for making \( M \) adaptive.

8.2.2 DESCRIPTOR SYSTEMS

Another possibility for extension is to give the motion model the form of a descriptor system:

\[
\begin{align*}
Y_{k+1}X_{k+1} &= \Phi_{k+1}X_k + T_{k+1}^\perp \\
Y_k &= \begin{bmatrix} h_k(x_k^{(1)}), \ldots, h_k(x_k^{(p)}) \end{bmatrix}
\end{align*}
\]

(8.207)
Descriptor systems offer a more general setup allowing constraints and system dynamics to be integrated in a more natural system description. They also are a convenient way to express relationships among dynamical quantities beyond causality [91, 9] For example, when the matrix \( \mathbf{Y}_{k+1} \) is singular, it cannot be passed over to the other side of the equality without the calculation of a pseudo-inverse. Such a descriptor form arises naturally in the skew symmetric decomposition of a rotation matrix discussed in Section 3.5.2. In fact, in light of equation (3.64), the discrete time system

\[
\mathbf{X}_{k+1} = \mathbf{R}_{k+1} \mathbf{X}_k + \mathbf{T}_{k+1} \mathbf{1}^H
\]  

(8.208)

where \( \mathbf{R}_{k+1} \) is a rotation transition matrix, can be re-written as

\[
(\mathbf{I}_{m \times m} - \mathbf{S}) \mathbf{X}_{k+1} = (\mathbf{I}_{m \times m} + \mathbf{S}) \mathbf{X}_k + \mathbf{T}'_{k+1}
\]

(8.209)

The \( m \times m \) matrix \( \mathbf{T}'_{k+1} \) in equation (8.209) is

\[
\mathbf{T}'_{k+1} = (\mathbf{I}_{m \times m} - \mathbf{S}) \mathbf{T}_{k+1} \mathbf{1}^H
\]

(8.210)

The result stated in equation (8.209) is valid as long as none of the eigenvalues of matrix \( \Phi_{k+1} \) is equal to minus one. The skew symmetric matrix \( \mathbf{S} \), the translation matrix \( \mathbf{T}'_{k+1} \) and the rotation matrix \( \mathbf{R}_{k+1} \) in equation (8.208) are all \( m \times m \) matrices. Clearly, equation (8.209) is of the form shown in equation (8.207). In Appendix H, it is shown that for \( T = 0 \) and \( p = 1 \), the skew symmetric matrix satisfying (8.209) is given by

\[
\text{vec}(\mathbf{S}) = \left[ (\mathbf{I}_{m \times m} \otimes \mathbf{N}) + (\mathbf{N} \otimes \mathbf{I}_{m \times m}) \right]^{-1} \text{vec}(\mathbf{M} - \mathbf{M}^H)
\]

(8.211)

where \( \mathbf{M} \) and \( \mathbf{N} \) are \( m \times m \) data matrices defined by

\[
\mathbf{M} = \sum_{i=1}^{N-1} (\mathbf{x}_{i+1} - \mathbf{x}_i)(\mathbf{x}_{i+1} + \mathbf{x}_i)^H
\]

\[
\mathbf{N} = \sum_{i=1}^{N-1} (\mathbf{x}_{i+1} + \mathbf{x}_i)(\mathbf{x}_{i+1} + \mathbf{x}_i)^H
\]

(8.212)
Equation (8.211) demonstrates that a least-square solution can also be obtained to estimate the parameters of the descriptor-based motion model, although a total least-square approach will generally be required to estimate $\mathbf{Y}_{k+1}$ and $\Phi_{k+1}$ simultaneously [42]. Such method allows to find a solution lying in the range of the matrix $[\mathbf{Y}_{k+1} \Phi_{k+1}]$ [134].

8.2.3 STATISTICAL FRAMEWORK

The motion model is developed in a fully deterministic setup. However, it is imaginable to use the various motion models evaluated across one or several image view-planes to build a probability space where concepts such as motion density function can be developed. This idea opens the door to the possibility of using many statistical techniques which until now are prohibited. Among these techniques are the calculations of conditional probabilities. It is mentioned in Chapter 1 that Bayesian approaches have been proposed to bring oriented smoothness constraints in the optical flow fields. This stochastic motion concept can be carried to a higher dimensionality setup in order to solve statistical motion reasoning problems. Spatial and temporal motion predictors can then be built. For example, along the time dimension, a $n$th order predictor would take the form

$$\hat{\Phi}_{k+1} = \sum_{i=1}^{n} a_i \Phi_{k+1-i} + V_{k+1}$$

$$\hat{T}_{k+1} = \sum_{i=1}^{n} b_i T_{k+1-i} + W_{k+1} \quad (8.213)$$

where $V_{k+1}$ and $W_{k+1}$ would be random, uncorrelated motion matrices and vectors, respectively. The knowledge of the covariance among the motion parameters could then be used to derive a Wiener predictor. Furthermore, Vector Quantization can be used to quantize the various transition matrices and trans-
lation vectors into a representative finite set of motion parameters. To avoid the calculation of probability density functions, non-parametric statistical techniques can be used. In this case, ranking of the transition matrices could be designed around the ranking in $\mathbb{R}^m$ of the $m$ eigenvalues of the transition matrices.

### 8.2.4 INTEGRATION OF EARLY VISION

In Chapter 6, it was proposed to validate the feature matching by running $N! = N(N-1)\ldots 3.2.1$ motion models in parallel. For implementation purposes, it might be desirable to use only one model for all the possible matches and reduce the dimension of the model as the level of motion understanding increases in time. Note that this does not preclude the motion model to sometimes increase its dimension when new motion ambiguities appear (splitting one correspondence into two or more correspondences in the next image frame). This approach gives the motion model a chance to organize the motion of the various features tracked. One way to bring learning in the model is to introduce weighting $p = N!$ dimensional vectors $\mu_k$ in the model as follows:

\[
\left(\begin{array}{c}
\mu_{k+1}H \otimes I_{pxp} \\
\mu_{k+1}H \otimes I_{pxp} \otimes \Phi_{k+1}
\end{array}\right)\text{vec}(X)_{k+1} = \\
\left(\begin{array}{c}
\mu_{k+1}H \otimes I_{pxp} \\
\mu_{k+1}H \otimes I_{pxp} \otimes \Phi_{k+1}
\end{array}\right)\text{vec}(X)_{k} + \left(\begin{array}{c}
\mu_{k+1} \otimes I_{mxm}
\end{array}\right)T_{k+1}
\]

(8.214)

The operator $\otimes$ represents the component to component multiplication of two matrices of equal dimensions. The vector $1$ in equation (8.214) is a $p$ dimensional vector whose entries are all equal to one. The question remains as to derive a learning strategy for the motion model in equation (8.214). The learning law for the vector $\mu$ can be drawn from the Least Mean Square algorithm [123]. In this case, either a search for the "smoothest" correspondence mapping or for the
smallest feature displacements are criteria to be considered. Intra or inter-image ranking of the recorded feature intensity values has the potential to be a key ingredient in early elimination of the most unlikely matches. Rank correlation algorithm seem to be particularly good candidates for this task [66]. However, because frequency domain estimation of displacement field is a non-parametric method (see 1.4.2) and because of the results on velocity aliasing obtained in Chapter 3, frequency domain analysis of the motion appears to provide the best scheme for accepting/rejecting candidate associations. Current displacement estimation methods do not account for the nonstationary nature of images in the spatial domain. In this respect, the Fourier transform needs functions of infinite extent in the spatial domain (periodic complex exponential functions) to localize sinusoids in the frequency domain perfectly. It is believed that the use of spatial windows with compact support or with rapid decay could help better localize the features and their displacements. The most basic technique to achieve that is to use the short-time Fourier transform. The short-time Fourier transform is simply the Fourier transform of the image multiplied by a window function being translated horizontally and vertically in the image. Its effect is to introduce a joint space-frequency (or time-frequency) analysis of the image. The window is usually a Gaussian window as it yields a minimal simultaneous localization of the image in the spatial and the translation domain [4]. Considerations about displacement aliasing and the two dimensional space-frequency uncertainty principle provide the basic tools for establishing possible image point associations [29]. It is believed that image to image displacement fields estimated in this fashion provide a sound foundation for establishing good feature associations.
BIBLIOGRAPHY


[69] Ljung, L. and Söderström, T., "Theory and Practice of Recursive Identification", The MIT Press Series in Signal Processing, Optimization and Control, Section 2.4, 2.5 and 1.6, pages 43-61, 1983.


[86] Nesi, P., Del Bimbo, A. and Sanz J.L.C., "A Unified Approach to Opti-
cal Flow Constraint Analysis", Department of Systems and Informatics

[87] Netravali, A.N. and Robbins, J.D., Motion-Compensated Television Cod-
ing: Part I, Bell System Technical Journal, Vol. 58, No. 3, March 1979,
pages 631-670.

[88] Netravali, A.N. and Robbins, J.D., Motion-Compensated Coding: Some
New Results;, Bell System Technical Journal, Vol. 59, No. 9, November

[89] Netravali, A.N. and Salz, J., "Algorithms for Estimation of Three-
Dimensional Motion", AT&T Technical Journal, Vol. 64, No. 2, February
1985, pages 335-344.

[90] Neudecker, H., "A Note on Kronecker Matrix Products and Matrix Equa-
tion Systems", SIAM Journal of Applied Mathematics, Vol. 17, No. 3,
May 1969, pages 603-606.

[91] Nikoukhah, R., Willsky, A.S. and Levy, B.C., "Kalman Filtering and Ric-
cati Equations for Descriptor Systems", IEEE Transactions on Automatic

Recursive Region-Splitting Method", Computer Graphics and Image Pro-

264-278.


APPENDICES
Appendix A

NON-LINEAR PROJECTION APPROXIMATION

The purpose of this appendix is to derive a second order Taylor series approximation of a multidimensional non-linear mapping \( h() \) defined from \( \mathbb{R}^m \) to \( \mathbb{R} \):

\[
h() : \quad \mathbb{R}^m \quad \longrightarrow \quad \mathbb{R}
\]

\[
x = \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix} \quad \quad h(x)
\]  \hspace{1cm} (A.215)

which is at least twice differentiable in an open neighborhood of a point \( A \) and given \( \varphi() \) constructed as follows

\[
\varphi() : \quad [0,1] \quad \longrightarrow \quad \mathbb{R}^m \quad \longrightarrow \quad \mathbb{R}
\]  \hspace{1cm} (A.216)

\[
t \quad A + ta \quad h(A + ta)
\]

the first derivative \( \varphi'(t) \) exists and is defined as

\[
\varphi'(t) = \sum_{j=1}^{j=m} \frac{\partial h}{\partial x_j}(A + ta)a_j
\]
\[ \begin{align*}
&= \begin{bmatrix}
\frac{\partial h}{\partial x_1}(A + ta), & \ldots, & \frac{\partial h}{\partial x_m}(A + ta)
\end{bmatrix}
\begin{bmatrix}
a_1 \\
\vdots \\
a_m
\end{bmatrix} \\
&= (\nabla h(A + ta))^H a
\end{align*} \] (A.217)

where \( \nabla h(A) \) denotes the gradient of \( h() \) about the point \( A \). The second derivative is

\[ \varphi''(t) = \sum_{j=1}^{m} \left( \sum_{k=1}^{m} \frac{\partial^2 h}{\partial x_k x_j}(A + ta) a_k \right) a_j \]

\[ = a^H \begin{bmatrix}
\frac{\partial^2 h}{\partial x_1 x_1}(A + ta) & \ldots & \frac{\partial^2 h}{\partial x_1 x_m}(A + ta) \\
\vdots & \ddots & \vdots \\
\frac{\partial^2 h}{\partial x_m x_1}(A + ta) & \ldots & \frac{\partial^2 h}{\partial x_m x_m}(A + ta)
\end{bmatrix} a \]

\[ = a^H \text{Hess}(A + ta) a \] (A.218)

where \( \text{Hess}(A) \) denotes the Hessian of \( h() \) about the point \( A \). When the function \( h() \) is a \( \ell \) dimensional function from \( \mathbb{R}^m \) to \( \mathbb{R}^\ell \), the first derivative becomes the following vector

\[ \begin{bmatrix}
\varphi'(t_1) \\
\vdots \\
\varphi'(t_\ell)
\end{bmatrix} = \begin{bmatrix}
(\nabla h(A + t_1 a))^H \\
\vdots \\
(\nabla h(A + t_\ell a))^H
\end{bmatrix} \]

\[ = \begin{bmatrix}
\frac{\partial h}{\partial x_1}(A + t_1 a) & \ldots & \frac{\partial h}{\partial x_m}(A + t_1 a) \\
\vdots & \ddots & \vdots \\
\frac{\partial h}{\partial x_1}(A + t_\ell a) & \ldots & \frac{\partial h}{\partial x_m}(A + t_\ell a)
\end{bmatrix} a \]
\[ D_{\mathbf{x}^n} \{ h(x) \} = J_a \] (A.219)

The definition of \( D_{\mathbf{x}^n} \{ h(x) \} \) is consistent with the definition of a matrix derivative operator whose action in this case would be to take the derivative of each column entry of the \( \ell \) dimensional function \( h() \) with respect to the row vector \( x^H \). The result is the \( \ell \times m \) Jacobian matrix \( J \) evaluated at \( (A + t_i a), 1 \leq i \leq \ell \). The second derivative is again a vector defined as

\[
\begin{bmatrix}
\varphi''(t_1) \\
\vdots \\
\varphi''(t_\ell)
\end{bmatrix} = \begin{bmatrix}
a^H \text{Hess}(A + t_1 a) a \\
\vdots \\
a^H \text{Hess}(A + t_\ell a) a
\end{bmatrix}
\] (A.220)

which can be expanded as follows:

\[
\begin{bmatrix}
\frac{\partial^2 h_1}{\partial x_1^2} (A + t_1 a) & \ldots & \frac{\partial^2 h_1}{\partial x_m^2} (A + t_1 a) \\
\vdots & \ddots & \vdots \\
\frac{\partial^2 h_\ell}{\partial x_1^2} (A + t_\ell a) & \ldots & \frac{\partial^2 h_\ell}{\partial x_m^2} (A + t_\ell a)
\end{bmatrix}
\begin{bmatrix}
\varphi''(t_1) \\
\vdots \\
\varphi''(t_\ell)
\end{bmatrix} = (a^H \otimes I_{\ell \times \ell})
\begin{bmatrix}
\frac{\partial^2 h_1}{\partial x_1 x_1} (A + t_1 a) & \ldots & \frac{\partial^2 h_1}{\partial x_1 x_m} (A + t_1 a) \\
\vdots & \ddots & \vdots \\
\frac{\partial^2 h_\ell}{\partial x_1 x_1} (A + t_\ell a) & \ldots & \frac{\partial^2 h_\ell}{\partial x_1 x_m} (A + t_\ell a)
\end{bmatrix}
\] (A.221)
The definition of $D_{xx^H} \{h(x)\}$ is consistent with the definition of a matrix derivative operator whose action in this case would be to take the derivative of each column entry of the $\ell$ dimensional function $h()$ with respect to the matrix $x^H$. This explains why we used this notation in equation (A.222). Furthermore, the notation $H$ has been chosen to make the expression even simpler while reminding the reader that this $m \ell \times m$ matrix is merely a reshuffled multidimensional Hessian matrix. If the norm of the vector $a$ is small, the function $h()$ can be approximated by the first three terms of its Taylor Series Expansion about the point $A$. This is given by

$$
\begin{bmatrix}
\varphi(t_1) \\
\vdots \\
\varphi(t_\ell)
\end{bmatrix} =
\begin{bmatrix}
\varphi(0) \\
\vdots \\
\varphi(0)
\end{bmatrix} +
\begin{bmatrix}
\varphi'(0) t_1 \\
\vdots \\
\varphi'(0) t_\ell
\end{bmatrix} +
\begin{bmatrix}
\varphi''(0) t_1^2 \\
\vdots \\
\varphi''(0) t_\ell^2
\end{bmatrix} +
\begin{bmatrix}
o(t_1^3) \\
\vdots \\
o(t_\ell^3)
\end{bmatrix}
$$

(A.223)

Setting $t_1 = \ldots = t_\ell = 1$, we obtain

$$
h(A + a) = h(A) + J a + \frac{1}{2} (a^H \otimes I_{\ell \times \ell}) H a + o(\|a\|^3)
$$

(A.224)

where $J$ and $H$ are both evaluated at $A$. 

Appendix B

NORM OF THE MODELING ERROR

This appendix starts from equation (3.38) to get the result claimed in equation (3.40). Let $\varepsilon_{N+1}^{(j)} = y_{N+1}^{(j)} - h_{N+1} \left( R_{N+1} \Phi N x_{N}^{(j)} \right)$. Equation (3.38) is

$$y_{N+1}^{(j)} - h_{N+1} \left( R_{N+1} \Phi N x_{N}^{(j)} \right) = \varepsilon_{N+1}^{(j)} - J_{N+1}^{(j)} R_{N+1} \Delta \Phi N x_{N}^{(j)}$$

$$- \frac{1}{2} \left( R_{N+1} \Delta \Phi N x_{N}^{(j)} \otimes I_{t \times t} \right) H_{N+1}^{(j)} R_{N+1} \Delta \Phi N x_{N}^{(j)}$$  \hspace{2cm} (B.232)

If follows that

$$\| y_{N+1}^{(j)} - h_{N+1} \left( \Phi N x_{N}^{(j)} \right) \|^2$$

$$= (\varepsilon_{N+1}^{(j)})^H \varepsilon_{N+1}^{(j)}$$

$$- 2 (\varepsilon_{N+1}^{(j)})^H \left( (x_{N}^{(j)})^H \otimes J_{N+1}^{(j)} R_{N+1} \right) vec(\Delta \Phi N)$$

$$- (\varepsilon_{N+1}^{(j)})^H \left[ (R_{N+1} \Delta \Phi N x_{N}^{(j)})^H \otimes I_{t \times t} \right] H_{N+1}^{(j)} \left( x_{N}^{(j)} \otimes R_{N+1} \right) vec(\Delta \Phi N)$$

$$+ vec(\Delta \Phi N)^H \left( (x_{N}^{(j)} \otimes J_{N+1}^{(j)} R_{N+1})^H \right) \left( (x_{N}^{(j)})^H \otimes J_{N+1}^{(j)} R_{N+1} \right) vec(\Delta \Phi N)$$

\hspace{2cm} (B.226)
Recall the property 
\((M \otimes S)(W \otimes Z) = MW \otimes SZ\) where \(M, S, W\) and \(Z\) are \(p \times q, v \times t, q \times w\) and \(t \times u\) matrices, respectively. It follows that

\[
\|y_{N+1}^{(j)} - h_{N+1}(\Phi x_{N}^{(j)})\|^2 = (\varepsilon_{N+1}^{(j)})^H \varepsilon_{N+1}^{(j)} - (2\varepsilon_{N+1}^{(j)})^H (x_{N}^{(j)})^H \otimes J_{N+1} R_{N+1} \) vec(\(\Delta \Phi_{N}\))
\]

\[
- \left[\left(\left(R_{N+1} \Delta \Phi_{N} x_{N}^{(j)}\right)^H \otimes I_{t \times t}\right) \varepsilon_{N+1}^{(j)}\right]^H H_{N+1}^{(j)} \left(\left(x_{N}^{(j)}\right)^H \otimes R_{N+1}\right) vec(\(\Delta \Phi_{N}\))
\]

\[
+ vec(\(\Delta \Phi_{N}\))^H \left(x_{N}^{(j)} \otimes R_{N+1}^{H} J_{N+1}^{(j)} J_{N+1}^{(j)} R_{N+1}\right) vec(\(\Delta \Phi_{N}\))
\]

(B.227)

vec(MPN) = \(\left(N^H \otimes M\right) vec(P)\), \(M, N, P\) being \(p \times q, s \times t\) and \(q \times s\) matrices, respectively. Applying this equality in one direction and the other, it can found that

\[
\left[\left(R_{N+1} \Delta \Phi_{N} x_{N}^{(j)}\right)^H \otimes I_{t \times t}\right] \varepsilon_{N+1}^{(j)}\right] = vec \left(\varepsilon_{N+1}^{(j)} \left(R_{N+1} \Delta \Phi_{N} x_{N}^{(j)}\right)^H\right)
\]

(B.228)

Moreover, since

\[
vec \left(\varepsilon_{N+1}^{(j)} \left(R_{N+1} \Delta \Phi_{N} x_{N}^{(j)}\right)^H\right)^H = \left[I_{m \times m} \otimes \varepsilon_{N+1}^{(j)}\right] vec \left(\left(R_{N+1} \Delta \Phi_{N} x_{N}^{(j)}\right)^H\right)
\]

(B.229)

and since vec(\(u\)) = vec(\(u^H\)) for any vector \(u\), it follows that

\[
\left[\left(R_{N+1} \Delta \Phi_{N} x_{N}^{(j)}\right)^H \otimes I_{t \times t}\right] \varepsilon_{N+1}^{(j)}\right]^H = \left[I_{m \times m} \otimes \varepsilon_{N+1}^{(j)}\right] vec \left(\left(R_{N+1} \Delta \Phi_{N} x_{N}^{(j)}\right)^H\right)
\]

(B.229)
where

\[
\begin{aligned}
&= \left( (z_N^{(j)})^H \otimes R_{N+1} \right) \text{vec}(\Delta \Phi_N) \left( I_{mxm} \otimes (\varepsilon_{N+1}^{(j)})^H \right) \\
&= \text{vec}(\Delta \Phi_N)^H \left( z_N^{(j)} \otimes R_{N+1} \right) \left( I_{mxm} \otimes (\varepsilon_{N+1}^{(j)})^H \right) \\
&= \text{vec}(\Delta \Phi_N)^H \left( z_N^{(j)} \otimes R_{N+1} \right) \left( I_{mxm} \otimes (\varepsilon_{N+1}^{(j)})^H \right)
\end{aligned}
\]

(B.230)

Now, using this last result in equation (B.225), the following result is obtained:

\[
\begin{aligned}
\| y_{N+1}^{(j)} - h_{N+1} (\Phi z_{N}^{(j)}) \|^2 \\
&= \left( \varepsilon_{N+1}^{(j)} \right)^H \left( \varepsilon_{N+1}^{(j)} \right) \\
&= 2\left( \varepsilon_{N+1}^{(j)} \right)^H \left( z_N^{(j)} \otimes J_{N+1}^{(j)} R_{N+1} \right) \text{vec}(\Delta \Phi_N) \\
&+ \text{vec}(\Delta \Phi_N)^H \left( z_N^{(j)} \otimes R_{N+1}^H \right) \left( J_{N+1}^{(j)} \right)^H \left( J_{N+1}^{(j)} R_{N+1}^{(j)} \right) \text{vec}(\Delta \Phi_N) \\
&- \text{vec}(\Delta \Phi_N)^H \left( z_N^{(j)} \otimes R_{N+1}^H \right) \mathcal{H}_{N+1}^{(j)} \left( (z_N^{(j)})^H \otimes R_{N+1}^{(j)} \right) \text{vec}(\Delta \Phi_N)
\end{aligned}
\]

(B.231)

where

\[
\mathcal{H}_{N+1}^{(j)} = \left( I_{mxm} \otimes (\varepsilon_{N+1}^{(j)})^H \right) H_{N+1}^{(j)}
\]

As a result, it can be written

\[
\begin{aligned}
\| y_{N+1}^{(j)} - h_{N+1} (\Phi z_{N}^{(j)}) \|^2 \\
&= \left( \varepsilon_{N+1}^{(j)} \right)^H \left( \varepsilon_{N+1}^{(j)} \right) \\
&= 2\left( \varepsilon_{N+1}^{(j)} \right)^H \left( z_N^{(j)} \otimes J_{N+1}^{(j)} R_{N+1} \right) \text{vec}(\Delta \Phi_N) \\
&+ \text{vec}(\Delta \Phi_N)^H \left( z_N^{(j)} \otimes R_{N+1}^H \right) \left( J_{N+1}^{(j)} \right)^H \left( J_{N+1}^{(j)} R_{N+1}^{(j)} \right) \text{vec}(\Delta \Phi_N) \\
&- \text{vec}(\Delta \Phi_N)^H \left( z_N^{(j)} \otimes R_{N+1}^H \right) \mathcal{H}_{N+1}^{(j)} R_{N+1}^{(j)} \text{vec}(\Delta \Phi_N)
\end{aligned}
\]

(B.233)
or, finally,

\[
\| y^{(j)}_{N+1} - h_{N+1} (\Phi x^{(j)}_N) \|^2 = \\
(\varepsilon^{(j)}_{N+1})^H \varepsilon^{(j)}_{N+1} - 2(\varepsilon^{(j)}_{N+1})^H \left( (x^{(j)}_N)^H \otimes J^{(j)}_{N+1} R_{N+1} \right) vec(\Delta \Phi_N) \\
+ vec(\Delta \Phi_N)^H \left( \\
x^{(j)}_N (x^{(j)}_N)^H \otimes R^H_{N+1} \left( (J^{(j)}_{N+1})^H J^{(j)}_{N+1} - h^{(j)}_{N+1} \right) R_{N+1} \right) vec(\Delta \Phi_N)
\]

(B.234)

which is the result claimed in equation (3.40).
Appendix C

EXTENDED RECURSIVE LEAST-SQUARE

The recursion formulae for the linearized recursive least-square algorithm are derived. The results are obtained upon comparing equations (3.33):

\[
\mathcal{F}_{N+1} (\Phi) = \mathcal{F}_{N+1} (\Phi_{N+1}) + \frac{1}{2} \text{vec}(\Phi_N - \Phi_{N+1} + \Delta \Phi_N)^H \Gamma_{N+1} \text{vec}(\Phi_N - \Phi_{N+1} + \Delta \Phi_N)
\]

(C.235)

with equation (3.36) in which the results of equation (3.40) were used:

\[
\mathcal{F}_{N+1} (\Phi) = \lambda^{\lceil \frac{N}{N+1} \rceil} \mathcal{F}_N (\Phi_N) + \lambda^{\lceil \frac{N}{N+1} \rceil} \frac{1}{2} \text{vec}(\Delta \Phi_N)^H \Gamma_N \text{vec}(\Delta \Phi_N)
\]

\[
+ \beta^{\lceil \frac{N}{N+1} \rceil} \sum_{j=1}^P \left\{ \left( \varepsilon_{N+1}^{(j)} \right)^H \varepsilon_{N+1}^{(j)} - 2(\varepsilon_{N+1}^{(j)})^H \left( \varepsilon_{N}^{(j)} \otimes J_{N+1}^{(j)} R_{N+1} \right) \right. \text{vec}(\Delta \Phi_N)
\]

\[
+ \text{vec}(\Delta \Phi_N)^H \left( \varepsilon_{N}^{(j)} \varepsilon_{N}^{(j)} \otimes R_{N+1}^H \left( J_{N+1}^{(j)} - H_{N+1}^{(j)} \right) R_{N+1} \right) \text{vec}(\Delta \Phi_N) \right\}
\]

(C.236)
The equality of the expressions in (C.235) and (C.236) has to hold for any $\text{vec}(\Delta \Phi N)$. By equating the first order terms, we obtain

$$\text{vec}(\Phi N - \Phi_{N+1})^H \Gamma_{N+1} = -2\beta_{N}^{[1:N+1]} \sum_{j=1}^{p} (\varepsilon_{N+1}^{(j)})^H \left( (\varepsilon_{N}^{(j)})^H \otimes J_{N+1}^{(j)} R_{N+1} \right)$$

(C.237)

The matrix $\Gamma_{N+1}$ is assumed symmetric. Taking the transpose of both sides and assuming that the inverse $P_{N+1} = \Gamma_{N+1}^{-1}$ exists, we obtain

$$\text{vec}(\Phi_{N+1}) = \text{vec}(\Phi_{N}) + 2\beta_{N}^{[1:N+1]} P_{N+1} \sum_{j=1}^{p} \left( \varepsilon_{N}^{(j)} \otimes R_{N+1}^H (J_{N+1}^{(j)})^H \right) \varepsilon_{N+1}^{(j)}$$

$$= \text{vec}(\Phi_{N})$$

$$+ 2\beta_{N}^{[1:N+1]} P_{N+1} \sum_{j=1}^{p} \left( \varepsilon_{N}^{(j)} \otimes R_{N+1}^H (J_{N+1}^{(j)})^H \right) \text{vec}(\varepsilon_{N+1}^{(j)})$$

$$= \text{vec}(\Phi_{N}) + 2\beta_{N}^{[1:N+1]} P_{N+1} \sum_{j=1}^{p} \text{vec} \left( R_{N+1}^H (J_{N+1}^{(j)})^H \varepsilon_{N+1}^{(j)} (\varepsilon_{N}^{(j)})^H \right)$$

(C.238)

which is the result claimed in equation (3.42). By equating the the second order terms in equations (C.235) and (C.236), we obtain

$$\Gamma_{N+1} = \lambda^{[1:N]} \Gamma_{N}$$

$$+ 2\beta_{N}^{[1:N+1]} \sum_{j=1}^{p} \left( x_{N}^{(j)} (x_{N}^{(j)})^H \otimes R_{N+1}^H \left( (J_{N+1}^{(j)})^H J_{N+1}^{(j)} - \mathcal{H}_{N+1}^{(j)} \right) R_{N+1} \right)$$

$$= \lambda^{[1:N]} \Gamma_{N} \left\{ I_{m^2 \times m^2} + 2\beta_{N}^{[1:N+1]} \frac{\lambda^{[1:N]}}{\lambda^{[1:N]}} P_{N} \right\} \sum_{j=1}^{p} \left( x_{N}^{(j)} (x_{N}^{(j)})^H \otimes R_{N+1}^H \left( (J_{N+1}^{(j)})^H J_{N+1}^{(j)} - \mathcal{H}_{N+1}^{(j)} \right) R_{N+1} \right)$$

(C.239)
Taking the inverse,

\[
P_{N+1} = \frac{1}{\lambda_{[1:N]}} \left\{ \mathbf{I}_{m^2 \times m^2} + 2 \frac{\beta_{[1:N+1]}^{[1:N+1]}}{\lambda_{[1:N]}} \mathbf{P}_N \right. \]
\[
\sum_{j=1}^{P} \left( \varepsilon_{N}^{(j)} \varepsilon_{N}^{(j)^H} \otimes \mathbf{R}_{N+1} \right) \left( \mathbf{J}_{N+1}^{(j)^H} \mathbf{J}_{N+1}^{(j)} - \mathbf{H}_{N+1}^{(j)} \right) \mathbf{R}_{N+1} \right\}^{-1} \mathbf{P}_N
\]

(C.240)

This is the result claimed in equation (3.43).
Appendix D

DIFFERENTIATION WITH RESPECT TO A MATRIX

The purpose of this section is to provide a few derivation rules of a matrix trace with respect to a given $m \times m$ matrix $E$. The matrix $M$ is a $r \times m$ matrix and the matrix $N$ is a $m \times r$ matrix. The fundamental equality used in this appendix is

$$\text{trace} \left\{ A^H D W \right\} = \text{vec}(A)^H (I_{m \times m} \otimes D) \text{vec}(W)$$  \hspace{1cm} (D.241)

where the matrices $A$, $D$ and $W$ are all $m \times m$ matrices.

Derivation rule 1:

$$\frac{\partial \text{trace} \{ MEEN \}}{\partial E} = E^H (NM)^H + (NM)^H E^H$$  \hspace{1cm} (D.242)

Proof:

$$\text{vec} \left( \frac{\partial \text{trace} \{ MEEN \}}{\partial E} \right) = \text{vec} \left( \frac{\partial \text{trace} \{ ENME \}}{\partial E} \right) = \frac{\partial \left\{ \text{vec} \left( E^H \right)^H (I_{m \times m} \otimes NM) \text{vec}(E) \right\}}{\partial \text{vec}(E)}$$
Derivation rule 2:

\[
\frac{\partial}{\partial \text{vec}(E)} \{ \text{trace} \{ \text{ME}^H E^H N \} \} = E^H (NM) + (NM) E^H \quad (D.243)
\]

Proof:

\[
\text{vec} \left( \frac{\partial}{\partial E} \left\{ \text{ME}^H E^H N \right\} \right) = \text{vec} \left( \frac{\partial}{\partial E} \left\{ E^H N E^H \right\} \right)
\]

\[
= \frac{\partial}{\partial \text{vec}(E)} \{ \text{trace} \{ \text{E}^H (I_{mxm} \otimes NM) \text{vec}(E^H) \} \}
\]

\[
= \text{vec} \left( E^H (NM) + (NM) E^H \right) \quad (D.245)
\]

Derivation rule 3:

\[
\frac{\partial}{\partial \text{vec}(E)} \{ \text{ME}^H E^H N \} = E (NM) + E (NM)^H \quad (D.246)
\]

Proof:

\[
\text{vec} \left( \frac{\partial}{\partial E} \left\{ \text{ME}^H E^H N \right\} \right) = \text{vec} \left( \frac{\partial}{\partial E} \left\{ E^H N E^H \right\} \right)
\]

\[
= \frac{\partial}{\partial \text{vec}(E)} \{ \text{trace} \{ \text{E}^H (I_{mxm} \otimes NM) \text{vec}(E^H) \} \}
\]
Derivation rule 4:

\[
\frac{\partial \text{trace} \{ MEE^HN \}}{\partial E} = (NM)^H E + (NM) E
\]  
(D.248)

Proof:

\[
\text{vec} \left( \frac{\partial \text{trace} \{ MEE^HN \}}{\partial E} \right) = \text{vec} \left( \frac{\partial \text{trace} \{ E^HNME \}}{\partial E} \right)
\]

\[
= \frac{\partial \left\{ \text{vec} (E)^H (I_{m \times m} \otimes NM) \text{vec} (E) \right\}}{\partial \text{vec} (E)}
\]

\[
= \text{vec} \left( (NM)^H E + (NM) E \right)
\]  
(D.249)

Derivation rule 5:

\[
\frac{\partial \text{trace} \{ MEN \}}{\partial E} = (NM)^H
\]  
(D.250)

Proof:

\[
\text{vec} \left( \frac{\partial \text{trace} \{ MEN \}}{\partial E} \right) = \text{vec} \left( \frac{\partial \text{trace} \{ ENM \}}{\partial E} \right)
\]

\[
= \frac{\partial \left\{ \text{vec} (E^H)^H \text{vec} (NM) \right\}}{\partial \text{vec} (E)}
\]

\[
= \text{vec} \left( (NM)^H \right)
\]  
(D.251)
Derivation rule 6:

\[
\frac{\partial \text{tr}\{ME^HN\}}{\partial E} = NM \quad \text{(D.252)}
\]

Proof:

\[
\text{vec}\left(\frac{\partial \text{trace}\{ME^HN\}}{\partial E}\right) = \text{vec}\left(\frac{\partial \text{trace}\{E^HNM\}}{\partial E}\right)
\]

\[
= \frac{\partial \{\text{vec}(E)^H\text{vec}(NM)\}}{\partial \text{vec}(E)}
\]

\[
= \text{vec}(NM) \quad \text{(D.253)}
\]
Appendix E

STOCHASTIC APPROXIMATION

The RLS algorithm derived in section 5.2.2 becomes a conventional RLS algorithm with an exponential forgetting factor when \( \lambda^{[1:N]} = \lambda \forall i \) and \( \beta_i^{[1:N+1]} = 1 \forall i \). However, this algorithm can also be viewed as a stochastic approximation algorithm. The purpose of this section is to establish this relationship. Since that \( P_{N+1} = \Gamma_{N+1}^{-1} \) and \( P_N = \Gamma_N^{-1} \), equation (5.130) can be re-written as follows

\[
P_{N+1}^{-1} = \lambda^{[1:N]} P_N^{-1} + \beta_i^{[1:N+1]} X_N X_N^H
\]

(E.254)

Now, let \( \beta_i^{[1:N+1]} = 1 \) and

\[
\lambda^{[1:N]} = \frac{\gamma^{[1:N]}}{\gamma^{[1:N+1]}} (1 - \gamma^{[1:N+1]})
\]

(E.255)

After multiplying both sides by \( \gamma^{[1:N+1]} \), equation (E.254) becomes

\[
\gamma^{[1:N+1]} P_{mm}[1 : N + 1] - 1 = \gamma^{[1:N]} (1 - \gamma^{[1:N+1]}) P_N^{-1} + \gamma^{[1:N+1]} X_N X_N^H
\]

\[
= \gamma^{[1:N]} P_N^{-1} + \gamma^{[1:N+1]} (X_N X_N^H - \gamma^{[1:N]} P_N^{-1})
\]

(E.256)

At this point, define the \( m \times m \) matrix \( \tilde{P}_N \) as

\[
\tilde{P}_N = (\gamma^{[1:N]})^{-1} P_N
\]

\[
= \gamma_N^{-1} \left[ \sum_{i=1}^{i=N-1} \beta_i^{[1:N]} X_i X_i^H \right]^{-1}
\]

(E.257)
so equation (E.256) becomes

$$
\tilde{P}_{N+1}^{-1} = \hat{P}_N^{-1} + \gamma^{[1:N+1]} (X_N X_N^H - \hat{P}_N^{-1})
$$

(E.258)

This is similar to the covariance update equation found in conventional stochastic approximation algorithms. To continue drawing the similarity between the RLS algorithm from the previous section with stochastic approximation algorithms, re-write equation (5.138) by recalling the result obtained in equation (5.137):

$$
\Phi_{N+1} = \Phi_N + \beta^{[1:N+1]} (X_{N+1} - \Phi_N X_N) K_N
$$

$$
= \Phi_N + \beta^{[1:N+1]} (X_{N+1} - \Phi_N X_N) X_N^H P_{N+1}^{-1}
$$

(E.259)

Taking again $\beta^{[1:N+1]} = 1$, this last equation can be written:

$$
\Phi_{N+1} = \Phi_N + (X_{N+1} - \Phi_N X_N) X_N^H P_{N+1}^{-1}
$$

$$
= \Phi_N + \gamma^{[1:N+1]} (X_{N+1} - \Phi_N X_N) X_N^H \gamma_{N+1}^{-1} P_{N+1}
$$

$$
= \Phi_N + \gamma^{[1:N+1]} (X_{N+1} - \Phi_N X_N) X_N^H \tilde{P}_{N+1}
$$

(E.260)

which again can be identified as the conventional results obtained in stochastic approximation. In fact this result is no more than an iterative Newton algorithm where the update is made of the innovation, the current positions $X_N$, the Hessian $P_{N+1}^{-1}$ and a gain $\gamma_{N+1}$ [69].
Appendix F

A-POSTERIORI ERROR CALCULATION

This appendix provides a detailed derivation of the a-posteriori error in the recursive multidimensional recursive least-square algorithm. The starting point of the derivations is the expansion of equation (5.143) by using equation (5.145) and equation (5.142). First notice that

\[
D_{N+1} - \Phi_{N+1}C_{N+1} = \left[ \sqrt{\lambda^{[1:N]}}D_N, \sqrt{\beta_N^{[1:N+1]}}X_{N+1} \right] \\
- \left( \Phi_N + \sqrt{\beta_N^{[1:N+1]}}\Xi_{N+1}K_N \right) \left[ \sqrt{\lambda^{[1:N]}}C_N, \sqrt{\beta_N^{[1:N+1]}}X_N \right] \\
= \left[ \sqrt{\lambda^{[1:N]}} \left( D_N - \Phi_N C_N - \sqrt{\beta_N^{[1:N+1]}}\Xi_{N+1}K_N C_N \right), \right. \\
\sqrt{\beta_N^{[1:N+1]}}\Xi_{N+1} - \beta_N^{[1:N+1]}\Xi_{N+1}K_N X_N \left. \right] \\
= \left[ \sqrt{\lambda^{[1:N]}} \left( D_N - \Phi_N C_N, \sqrt{\beta_N^{[1:N+1]}}\Xi_{N+1} \right) \right. \\
\left. - \sqrt{\beta_N^{[1:N+1]}}\Xi_{N+1} \left[ \sqrt{\lambda^{[1:N]}}K_N C_N, \sqrt{\beta_N^{[1:N+1]}}K_N X_N \right] \right] \\
\text{(F.261)}
\]
Use equation (F.261) into equation (5.143)

\[
\xi_{N+1}^2 = \text{trace} \left\{ \left( \sqrt{\lambda_{[1:N]}} \left( D_N - \Phi_N C_N \right), \sqrt{\beta_{N^{[1:N+1]}}} \Xi_{N+1} \right) \right. \\
- \sqrt{\beta_{N^{[1:N+1]}}} \Xi_{N+1} \left[ \sqrt{\lambda_{[1:N]}} K_N C_N, \sqrt{\beta_{N^{[1:N+1]}}} K_N X_N \right] \left. \right) \right. \\
\left. \left. \left( \left[ \sqrt{\lambda_{[1:N]}} (D_N - \Phi_N C_N)^H \sqrt{\beta_{N^{[1:N+1]}}} \Xi_{N+1}^H \right) \right. \\
- \sqrt{\beta_{N^{[1:N+1]}}} \left[ \sqrt{\lambda_{[1:N]}} C_N^H K_N^H \right. \left. X_N^H K_N^H \Xi_{N+1}^H \right. \right) \right. \\
\right\} \quad (F.262) \\
\]

Since \text{trace}() is a linear operator,

\[
\xi_{[1:N+1]}^2 = \lambda_{[1:N]} \text{trace} \left\{ \left( D_N - \Phi_N C_N \right) (D_N - \Phi_N C_N)^H \right\} \\
+ \beta_{N^{[1:N+1]}} \text{trace} \left\{ \Xi_{N+1} \Xi_{N+1}^H \right\} \\
- \lambda_{[1:N]} \sqrt{\beta_{N^{[1:N+1]}}} \text{trace} \left\{ \left( D_N - \Phi_N C_N \right) C_N^H K_N^H \Xi_{N+1}^H \right\} \left. \right|_{0_{m \times m}} \\
- \lambda_{[1:N]} \sqrt{\beta_{N^{[1:N+1]}}} \text{trace} \left\{ \Xi_{N+1} K_N C_N (D_N - \Phi_N C_N)^H \right\} \left. \right|_{0_{m \times m}} \\
- \beta_{N^{[1:N+1]}} \sqrt{\beta_{N^{[1:N+1]}}} \text{trace} \left\{ \Xi_{N+1} K_N X_N K_N^H \Xi_{N+1}^H \right\} \\
- \beta_{N^{[1:N+1]}} \sqrt{\beta_{N^{[1:N+1]}}} \text{trace} \left\{ \Xi_{N+1} K_N X_N \Xi_{N+1}^H \right\} \\
+ \lambda_{[1:N]} \beta_{N^{[1:N+1]}} \text{trace} \left\{ \Xi_{N+1} K_N C_N C_N^H K_N^H \Xi_{N+1}^H \right\} \\
+ \beta_{N^{[1:N+1]}} \text{trace} \left\{ \Xi_{N+1} K_N X_N X_N^H K_N^H \Xi_{N+1}^H \right\} \quad (F.263) \\
\]
The cancellations occurring in equation (F.263) are due to the equalities in equation (5.141). Re-grouping the terms,

\[ \xi_{[1:N+1]}^2 = \lambda^{[1:N]} \xi_{[1:N]}^2 + \beta_N^{[1:N+1]} \left( \text{trace} \left\{ \mathbf{E}_{N+1} (\mathbf{I}_{p \times p} - \sqrt{\beta_N^{[1:N+1]} \mathbf{K}_N \mathbf{X}_N})^H (\mathbf{I}_{p \times p} - \sqrt{\beta_N^{[1:N+1]} \mathbf{K}_N \mathbf{X}_N}) \mathbf{E}_{N+1}^H \right\} \right) + \lambda^{[1:N]} \text{trace} \left\{ \mathbf{E}_{N+1} \mathbf{K}_N \mathbf{C}_N \mathbf{C}_N^H \mathbf{K}_N^H \mathbf{E}_{N+1}^H \right\} \]

(F.264)

At this point, note that equation (5.134) can be used to get

\[ \mathbf{I}_{p \times p} = \mathbf{I}_{p \times p} - \sqrt{\beta_N^{[1:N+1]} \mathbf{K}_N \mathbf{X}_N} \]

\[ = (\lambda^{[1:N]} \mathbf{I}_{p \times p} + \beta_N^{[1:N+1]} \mathbf{X}_N^H \mathbf{P}_N \mathbf{X}_N)^{-1} \lambda^{[1:N]} \mathbf{I}_{p \times p} + \beta_N^{[1:N+1]} \mathbf{X}_N^H \mathbf{P}_N \mathbf{X}_N \]

(F.265)

The result above is obtained by using

\[ \mathbf{I}_{p \times p} = (\lambda^{[1:N]} \mathbf{I}_{p \times p} + \beta_N^{[1:N+1]} \mathbf{X}_N^H \mathbf{P}_N \mathbf{X}_N)^{-1} (\lambda^{[1:N]} \mathbf{I}_{p \times p} + \beta_N^{[1:N+1]} \mathbf{X}_N^H \mathbf{P}_N \mathbf{X}_N) \]

(F.266)

Also, from the definition of the Kalman gain,

\[ \mathbf{K}_N \mathbf{C}_N \mathbf{C}_N^H \mathbf{K}_N^H = (\lambda^{[1:N]} \mathbf{I}_{p \times p} + \beta_N^{[1:N+1]} \mathbf{X}_N^H \mathbf{P}_N \mathbf{X}_N)^{-1} \left( \beta_N^{[1:N+1]} \mathbf{X}_N^H \mathbf{P}_N \mathbf{C}_N \mathbf{C}_N^H \mathbf{P}_N^H \right) \left( \lambda^{[1:N]} \mathbf{I}_{p \times p} + \beta_N^{[1:N+1]} \mathbf{X}_N^H \mathbf{P}_N \mathbf{X}_N \right)^{-1} \]

(F.267)

Since

\[ \mathbf{P}_N = \mathbf{P}_N^H = (\mathbf{C}_N \mathbf{C}_N^H)^{-1} \]

(F.268)
Equation (F.267) becomes

$$\mathbf{K}_N \mathbf{C}_N \mathbf{C}_N^H \mathbf{K}_N^H = \left( \lambda^{[1:N]} \mathbf{I}_{p \times p} + \beta_N^{[1:N+1]} \mathbf{X}_N^H \mathbf{P}_N \mathbf{X}_N \right)^{-1}$$

$$\beta_N^{[1:N+1]} \mathbf{X}_N^H \mathbf{P}_N \mathbf{X}_N$$

$$\left( \lambda^{[1:N]} \mathbf{I}_{p \times p} + \beta_N^{[1:N+1]} \mathbf{X}_N^H \mathbf{P}_N \mathbf{X}_N \right)^{-1}$$

Going back to equation (F.264), use the results in equations (F.265) and (F.269) to obtain

$$\xi_{N+1}^2 = \lambda^{[1:N]} \xi_N^2 + \beta_N^{[1:N+1]} \lambda^{[1:N]} \text{trace} \left\{ \mathbf{E}_{N+1} \left( \lambda^{[1:N]} \mathbf{I}_{p \times p} + \beta_N^{[1:N+1]} \mathbf{X}_N^H \mathbf{P}_N \mathbf{X}_N \right)^{-1} \right\}$$

$$\lambda^{[1:N]} \xi_N^2 + \beta_N^{[1:N+1]} \lambda^{[1:N]} \text{trace} \left\{ \mathbf{E}_{N+1} \left( \lambda^{[1:N]} \mathbf{I}_{p \times p} + \beta_N^{[1:N+1]} \mathbf{X}_N^H \mathbf{P}_N \mathbf{X}_N \right)^{-1} \mathbf{Y}_{N+1}^H \right\}$$

$$\lambda^{[1:N]} \xi_N^2 + \beta_N^{[1:N+1]}$$

$$\text{trace} \left\{ \mathbf{E}_{N+1} \left( \lambda^{[1:N]} \mathbf{I}_{p \times p} + \beta_N^{[1:N+1]} \mathbf{X}_N^H \mathbf{P}_N \mathbf{X}_N \right)^{-1} \lambda^{[1:N]} \mathbf{I}_{p \times p} \mathbf{E}_{N+1}^H \right\}$$

Now, note

$$\left( \lambda^{[1:N]} \mathbf{I}_{p \times p} + \beta_N^{[1:N+1]} \mathbf{X}_N^H \mathbf{P}_N \mathbf{X}_N \right)^{-1} \lambda^{[1:N]} \mathbf{I}_{p \times p}$$

$$\left( \lambda^{[1:N]} \mathbf{I}_{p \times p} + \beta_N^{[1:N+1]} \mathbf{X}_N^H \mathbf{P}_N \mathbf{X}_N - \beta_N^{[1:N+1]} \mathbf{X}_N^H \mathbf{P}_N \mathbf{X}_N \right)$$

$$\left( \lambda^{[1:N]} \mathbf{I}_{p \times p} + \beta_N^{[1:N+1]} \mathbf{X}_N^H \mathbf{P}_N \mathbf{X}_N - \beta_N^{[1:N+1]} \mathbf{X}_N^H \mathbf{P}_N \mathbf{X}_N \right)$$
\[ I_{p \times p} - \left( \lambda^{[1:N]} I_{p \times p} + \beta_N^{[1:N+1]} X_N^H P_N X_N \right)^{-1} \beta_N^{[1:N+1]} X_N^H P_N X_N \]

\[ = I_{p \times p} - \sqrt{\beta_N^{[1:N+1]}} K_N X_N \]  

(F.271)

The result obtained in equation (F.270) becomes

\[ \xi_{N+1}^2 = \lambda^{[1:N]} \xi_N^2 + \beta_N^{[1:N+1]} \lambda^{[1:N]} \text{trace} \left\{ \Xi_{N+1} \left( \lambda^{[1:N]} I_{p \times p} + \beta_N^{[1:N+1]} X_N^H P_N X_N \right)^{-1} \Xi_{N+1}^H \right\} \]

\[ = \lambda^{[1:N]} \xi_N^2 + \beta_N^{[1:N+1]} \text{trace} \left\{ \Xi_{N+1} \left( I_{p \times p} - \sqrt{\beta_N^{[1:N+1]}} K_N X_N \right) \Xi_{N+1}^H \right\} \]  

(F.272)

which is the desired result.
Appendix G

ESTIMATION BANDWIDTH FOR CONSTANT SIGNAL TO NOISE RATIO

The purpose of this appendix is to determine the forgetting profile to keep the motion estimation algorithm operate at a constant signal to noise ratio. The signal to noise ratio, call it $SNR$, is defined as the ratio of the innovation power and the normalized a-posteriori power. It measures the amount of information brought by the innovation terms with respect to the noise or uncertainty in the motion model. Let $\gamma = \lambda_{[1:N]} / \beta_{N}^{[1:N+1]}$. The goal is to chose $\gamma$ such that given an $SNR$ value, the following equality is verified

$$SNR = \frac{\xi_{N+2}^H \xi_{N+2}}{\xi_{N+1}^2}$$

(G.273)

where

$$\xi_{N+2} = \xi_{N+2} - \Phi_{N+1} \xi_{N+1}$$

$$= \xi_{N+2} - \left( \Phi_N + \xi_{N+1} \frac{x_N^H P_N}{\gamma + x_N^H P_N x_N} \right) \xi_{N+1}$$

$$= (\xi_{N+2} - \Phi_N \xi_{N+1}) + \frac{x_N^H P_N \xi_{N+1} \xi_{N+1}}{\gamma + x_N^H P_N x_N}$$
\[
= \varepsilon + \frac{\underline{x}_N^H \underline{P}_N \underline{x}_{N+1} \varepsilon_{N+1}}{\gamma + \underline{x}_N^H \underline{P}_N \underline{x}_N} \tag{G.274}
\]

and where, from equation (5.159),
\[
(\gamma^{-1} + \mathcal{S}^{[1:N]}) \xi_{N+1}^2 = \mathcal{S}^{[1:N]} \xi_N^2 + \frac{\underline{x}_N^H \varepsilon_{N+1}}{\gamma + \underline{x}_N^H \underline{P}_N \underline{x}_N} \tag{G.275}
\]

Equations (G.273), (G.274) and (G.275) yield

\[
\xi_{N+1}^2 \frac{SNR}{\gamma_sN} = \xi_{N+2}^2 \frac{SNR}{\gamma_sN} - \xi_{N+1}^2 \\
(\gamma^{-1} + \mathcal{S}^{[1:N]}) \xi_{N+1}^2 \frac{SNR}{\gamma_sN} = (\gamma^{-1} + \mathcal{S}^{[1:N]}) \xi_{N+2}^2 \frac{SNR}{\gamma_sN} \\
(\gamma^{-1} + \mathcal{S}^{[1:N]}) \xi_{N+1}^2 \frac{SNR}{\gamma_sN} = (1 + \gamma \mathcal{S}^{[1:N]}) \xi_{N+2}^2 \frac{SNR}{\gamma_sN} \tag{G.276}
\]

Consider the two terms of the equality separately. The term on the left can be expanded as follows:

\[
(\gamma^{-1} + \mathcal{S}^{[1:N]}) \xi_{N+1}^2 \frac{SNR}{\gamma_sN} \\
= \left( \mathcal{S}^{[1:N]} \xi_N^2 + \frac{\underline{x}_N^H \varepsilon_{N+1}}{\gamma + \underline{x}_N^H \underline{P}_N \underline{x}_N} \right) \gamma \frac{SNR}{\gamma_sN} \\
= \frac{\left( \mathcal{S}^{[1:N]} \xi_N^2 + \left( \gamma + \underline{x}_N^H \underline{P}_N \underline{x}_N \right) + \underline{x}_N^H \varepsilon_{N+1} \right) \gamma \frac{SNR}{\gamma_sN}}{\gamma + \underline{x}_N^H \underline{P}_N \underline{x}_N} \tag{G.277}
\]

The term on the right in equation (G.276) can be expanded as follows:

\[
(1 + \gamma \mathcal{S}^{[1:N]}) \xi_{N+2}^2 \frac{SNR}{\gamma_sN} = (1 + \gamma \mathcal{S}^{[1:N]}) \left\{ \varepsilon^H \varepsilon + \frac{2 \underline{x}_N^H \underline{P}_N \underline{x}_{N+1} \varepsilon_{N+2}}{\gamma + \underline{x}_N^H \underline{P}_N \underline{x}_N} \right\} \\
+ \frac{\underline{x}_{N+1}^H \varepsilon_{N+1} \left( \underline{x}_N^H \underline{P}_N \underline{x}_{N+1} \right)^2}{\left( \gamma + \underline{x}_N^H \underline{P}_N \underline{x}_N \right)^2} \tag{G.278}
\]

Establish the equality between equation (G.277) and (G.278) and multiply both sides by \((\gamma + \underline{x}_N^H \underline{P}_N \underline{x}_N)^2\) so the equality in equation (G.276) can be re-written
This is a third order polynomial in unknown $\gamma$. Therefore, there are up to three solutions for $\gamma$. Negative solutions are not valid and if it exists, the positive solution which is the closest to 1 must be selected.
Appendix H

DESCRIPTOR SYSTEM BASED ON
ROTATION MATRIX

The purpose of this appendix is to derive the least-square estimate of the skew-
symmetric matrix appearing in the descriptor motion system in Section 8.2.2. Note that in the case \( p = 1 \) and \( T_{k+1} = 0 \), equation (8.209) becomes

\[
(I_{m \times m} - S) x_{k+1} = (I_{m \times m} + S) x_k
\]  

which can be re-written as

\[
(x_{k+1} - x_k) = S (x_{k+1} + x_k)
\]

Since \( S \) must be a skew-symmetric matrix, include this constraint in equation (H.281) by letting \( S = F - F^H \) where \( F \) is a \( m \times m \) matrix. Clearly, \( S^H = F^H - F = -S \). Given \( N \) consecutive positions of the same object feature, determine \( F \) by performing the following minimization

\[
\min_F \left\{ \sum_{i=1}^{N-1} \left( (x_{i+1} - x_i) - (F - F^H) (x_{i+1} + x_i) \right)^H \right\} 
\]

\[
\left( (x_{i+1} - x_i) - (F - F^H) (x_{i+1} + x_i) \right) \right\} 
\]

(H.282)
With the help of the derivation rules established in Appendix D, it can be verified that at the minimum, the following equality is verified:

\[ \mathbf{M}^H - \mathbf{M} + \left( \mathbf{F} - \mathbf{F}^H \right) \mathbf{N} + \mathbf{N} \left( \mathbf{F} - \mathbf{F}^H \right) = \mathbf{0}_{m \times m} \]  

where \( \mathbf{0}_{m \times m} \) is the \( m \times m \) matrix whose elements are all equal to zero and where

\[
\mathbf{M} = \sum_{i=1}^{N-1} (x_{i+1} - x_i) (x_{i+1} + x_i)^H
\]

\[
\mathbf{N} = \sum_{i=1}^{N-1} (x_{i+1} + x_i) (x_{i+1} + x_i)^H
\]

Since \( \mathbf{S} = \mathbf{F} - \mathbf{F}^H \), equation (H.284) becomes

\[
\mathbf{M} - \mathbf{M}^H = \mathbf{SN} + \mathbf{NS}
\]

Apply the \( \text{vec}() \) operator to the left and right side of the equality in equation (H.285) to obtain

\[
\text{vec} \left( \mathbf{M} - \mathbf{M}^H \right) = \left[ \left( \mathbf{I}_{m \times m} \otimes \mathbf{N} \right) + \left( \mathbf{N} \otimes \mathbf{I}_{m \times m} \right) \right] \text{vec} (\mathbf{S})
\]

The result claimed in equation (8.211) can be easily obtained by passing the inverse of the Kronecker sum \( \mathbf{N} \oplus \mathbf{N} \) to the left side of equation (H.286).