Simultaneously Modelling and Tracking using Mutual Information

NDH Dowson

Submitted for the Degree of
Doctor of Philosophy
from the
University of Surrey

Centre for Vision, Speech and Signal Processing
School of Electronics and Physical Sciences
University of Surrey
Guildford, Surrey GU2 7XH, U.K.

October 2006

Abstract

The aim of this work is to track non-rigid objects using appearance in real-time without a pre-learned model. The tracker should be robust to noise, occlusions, changes in lighting conditions, and pose variations. Recovery from tracking failure should be possible.

The approach taken is to localise small image patches within each frame of a video sequence using function optimisation. Mutual Information (MI) is chosen as the similarity metric, due to its robustness to outlier pixel values and lack of assumptions about linear intensity relationships. The choice of similarity metric and descriptor strongly influence performance, whatever the application, hence the contributions made here have wide applicability.

Despite the wide use of MI for registration, many open problems still exist. MI suffers from artefacts in its function surface, which can prevent convergence. Numerous different MI methods have been proposed to overcome artefacts, but few methods have published analytic derivatives, precluding their use in fast Newton-type optimisation. This thesis shows that MI methods only differ in how the joint-histogram is generated from image intensity samples, and places them into a single mathematical framework consisting of four families. Analytic derivatives are derived in every case.

The availability of analytic derivatives allows MI to be placed into a Lucas-Kanade tracking framework, termed MILK. An inverse-compositional formulation for optimisation is presented. This yields a faster convergence, because the Hessian is pre-computed.

In addition to slowing convergence, artefacts shift the maximum away from its "true" position, *i.e.* they induce bias. Multiple proposals to reduce the effect of artefacts are tested in this light. Of these, only super-sampling reduces artefacts in a controllable way. This concurs with existing theory that histogram (and hence MI) accuracy is limited by the number of samples and the histogram resolution.

Super-sampling is taken to its logical conclusion using an extension of the Non-Parametric (NP) windowing method. NP-windowing is equivalent to taking samples at infinite resolution, by utilising the implicit structure of image data to integrate interpolation functions over entire pixels. Novel simplifications to existing theory using Green's theorem are also introduced. NP-windowing yields biases of half (or less) than that of other MI methods. It also improves convergence, and is independent of template size.

MI is applied to tracking where the problems of drift and mismatch, must be traded off. The primary cause of drift and mismatch is simplistic appearance models. This is overcome using a proposed method to cluster appearance exemplars on-the-fly using a Bayesian approach to descriminate between cluster inliers and outliers. The method is termed Simultaneous Modelling and Tracking (SMAT). Greedy selection of particular clusters for tracking allows real-time performance. SMAT is extended to model object shape and recovery from tracking failures is made possible by selective correction of the feature position. The zig-zag-zen family of semi-automated tests is proposed, making emperical tests using multiple (11) sequences of many frames (568 on average) practical. In testing, the advantages of: MI over SSD, SMAT over other template based methods, and shape modelling are clearly demonstrated.

Key words: Mutual Information, Artefacts, Sampling, Resolution, Registration, Appearance Modelling, Shape Modelling, Tracking, Template Update Problem, Lucas-Kanade framework, Inverse Compositional formulation

Acknowledgements

This work would not have been possible without the financial support of the Centre for Vision Speech and Signal Processing (CVSSP) and the Department of Education and Skills, UK, in the form of a grant and on Overseas Research Student Scholarship.

I must especially thank my supervisor, Richard Bowden, for all his excellent advice and suggestions; and his tireless support. Working with you has been a pleasure and an inspiration. I hope to be as good and kind a mentor as you, when the time comes.

I also had the pleasure of working with Timor Kadir for some of this work, who was an excellent source of ideas. Many other people in the CVSSP have also unselfishly taken time to provide advice and assistance, especially Bill Christmas, Jiri Matas, Lee Gregory, Simon Alcott, Rob King, John Stark, Marc Servais, Susan Servais, Helen Cooper, Eng-Ong Jon, Tony Micilotta and John Chiverton.

Finally, the encouragement of family and my wife, Michelle, before and during this PhD must be gratefully acknowledged.

Thank you.

Contents

List of Tables

List of Figures

Nomenclature and Notation

Symbol	Explanation
Interpolation Methods	
BCI	Bi-Cubic Interpolation
BLI	Bi-Linear Interpolation
H-BLI	Half Bi-Linear Interpolation
NNI	Nearest Neighbour Interpolation
Similarity Metrics	
MI	Mutual Information
NC	Normalised Correlation
SSD	Sum of Squared Differences
Histogram Sampling Methods	
IPZ/ipz	In-ParZen windowing (lower case used in mathematical functions)
PPZ/ppz	Post-ParZen windowing
PVI	Partial Volume Interpolation
PVE/pve	Partial Volume Estimation
STD/std	STanDard sampling
Miscellaneous Terms	
DoF	Degrees of Freedom
LK tracking	Lucas-Kanade Tracking
SLAM	Simultaneous Localisation and Mapping
SMAT	Simultaneous Modelling and Tracking
PDF	Probability Distribution Function
ZIP	Zero Interpolation Principle

Table 1: List of acronyms

Symbol	Explanation
$\mathbf{x}$	2D spatial position in an image given as a vector.
$\mathbf{x}'$	
$\mathbf{v}$	Vector defining a set of warp parameters or homography. No. components varies with warp type e.g. 2D for translation, 3D for Euclidean, 4D for similarity, 6D for affine.
$\mathbf{w}(\mathbf{x}, \mathbf{v})$	Function taking a 2D spatial position, $\mathbf{x}$ and a warp parameter, $\mathbf{v}$, outputting a warped spatial position.
$\mathbf{x_w}$	Shorthand for $\mathbf{w}(\mathbf{x}, \mathbf{v})$.
f_R	The reference (fixed) image, when registering two images.
f_T	The template (floating) image, when registering two images.
r	Index of image intensity in the reference image.
t	Index of image intensity in the template image.
d_{mi}	Mutual Information distance function.
d_{ssd}	Sum of squared differences distance function.
d_{nc}	Normalised Correlation distance function.
$h_r(r)$	Histogram function. No. pixel values taking value r in f_r.
$h_t(t)$	Histogram function. No. pixel values taking value t in f_t.
$h_rt(r, t)$	Joint Histogram function. No. pixel values respectively taking value r and t in corresponding positions in f_r and f_t.
p_r	Probability distribution function of intensity in f_r.
p_t	Probability distribution function of intensity in f_t.
p_{rt}	Joint probability distribution function of intensity in f_r and f_t.
$N_\mathbf{x}$	The number of pixels in an image patch.
N_i	The number of intensity indices in an image.
$\mathbb{Z}$	The set of all integers.
$\mathbb{R}$	The set of all real numbers.
C_n	The smoothness of a surface. The surface is continuous to the nth derivative.
o	The order of a particular interpolation method.
p_x	Probability Distribution Function of some variable x

Table 2: Standard Symbols Used

Chapter 1

Introduction

The motivation behind this work was to track non-rigid objects using appearance in real-time without a pre-learned model. The tracker should be robust to noise, changes in lighting conditions, occlusions, background clutter and changes in the appearance of the object, due to its non-rigidity and pose variation. Moreover, recovery from tracking failure should be possible, as should failure detection.

Tracking algorithms come in a number of guises and are widely used in many different applications of which a few examples are given. Small patches are tracked to solve for the 3D positions of tracked features, to build 3D models of objects [85], to insert virtual objects into augmented reality environments [95], to assist in robot navigation and mapping [23, 58], and to solve for structure and motion within a scene [83]. In some cases, multiple regions in a scene are correlated over a sequence of frames to obtain a description of scene motion [74]. Tracking objects through sequences is also useful to generate motion models of humans [86] and hands [80, 41] and for use in downstream processes like identification *e.g.* when head tracking [64, 19]. In addition, the detection of humans may be improved by the use of so-called temporal templates [8].

Tracking in this work, is treated as a series of sequential registrations, where a similarity measure between a template and a reference image is maximised. Although tracking algorithms vary widely, there are three requirements that most methods have in common:

1. A method to represent image data.

2. A warp or transformation between the coordinates of the template image (appearance model) and the current frame (reference image) in the sequence, in order to find the region of correspondence between them.

3. A similarity metric to compare the corresponding sets of data from the current frame and the template.

Numerous methods to represent data exist *e.g.*: local polar patches of height or intensity [35], collections of local gradient histograms called SIFT features [44], and edge maps for chamfer-matching [80]. Localisation of the outline or silhouette of an object has many different applications [75, 31, 79, 101], and the use of contours can be particularly useful when solving for sets of occlusions in a scene [100]. In this work, a somewhat traditional representation, consisting of the intensities of a rectangular image patch, is used. A variety of similarity metrics are available to compare image patches, and a rich description of articulated object motion is possible [2, 6], since the rectangle centre gives an explicit object location for use in downstream algorithms.

Standard rigid warps are utilised in this work, including: translation, Euclidean, similarity and affine warps. Rigid warps are used to avoid over-parameterisation and resulting instability when using the small image patches associated with tracking applications. The size of the image patches depend on the features

being tracked and the resolution of the video sequence, but typical dimensions ranged from 11×11 pixels to 50×30 pixels.

Although Sum of Square Differences and Normalised Correlation are popular similarity metrics for tracking, Mutual Information is favoured in this work. Mutual Information is used, due to its well known robustness to noise, occlusion and variations in lighting conditions.

In Chapter 2, the reasons for the above choices of representation method, warp parameterisation and similarity metric are discussed in more detail, within the context of previous work. A background to the remainder of the thesis is also provided. Within this scope, several contributions are made.

Multiple variants of the MI function exist. The most common variants form four broad families, which differ primarily in how the signal (image) is sampled. Chapter 3 places these four families into a single mathematical framework. The Jacobian and Hessian of each type of MI is derived, which is novel for three families. The methods are also compared in terms of computational cost and registration performance.

It was found that MI has artefacts in its surface, which has two effects: the introduction of numerous spurious local maxima, and the biasing of the global maximum away from its "true" position. Extensive literature exists on this subject proposing a variety of solutions [47, 66, 70, 34], but many solutions only solve the symptoms of the problem (artefacts in the function surface) rather than the cause (insufficient samples and ignoring available information). In Chapter 4 a detailed analysis is made of the root causes of artefacts. Several measures to reduce the effects of artefacts are proposed along with a discussion of existing proposals. All of these methods are methodically tested, but one method in particular: super-sampling the template; allows the effect of artefacts on bias

and convergence accuracy to be finely controlled and removes the assumption of independence and identical distribution.

In Chapter 5, super-sampling is extended to its logical conclusion, where a method is proposed that analytically integrates the effect of an interpolation method when generating a histogram from a pair of sampled images. The method extends a technique called Non-Parametric Windowing, introduced by Kadir and Brady [37], which is effectively the same as taking infinite samples. This eliminates much of the bias in the position of the global maximum and improves convergence.

The Jacobian and Hessian derived in Chapter 3 allows MI to be placed into a Lucas-Kanade (LK) tracking framework. In a LK-framework, localisation is performed by using the Jacobian and Hessian to iteratively update the warp parameters. Calculating the Hessian is usually the most expensive operation in each iteration. In Chapter 6, the MI function is reformulated into a so-called inverse-compositional formulation to give a constant Hessian, using an approach similar to that proposed by Baker and Matthews for SSD [4]. The proposed technique is termed MILK: MI in a Lucas-Kanade framework.

Having placed MI into a LK-framework and surmounting the problems associated with registering small image patches, MI is used for tracking features through a motion sequence. The problems of drift and misrepresentation, due to simplistic appearance models, are overcome by a proposed method to incrementally cluster feature exemplars. The cluster used for localisation is selectively chosen on-the-fly, allowing real-time performance. A Bayesian approach is used to threshold the inliers and outliers of each cluster. This method, referred to as Simultaneous Modelling and Tracking (SMAT) is presented in Chapter 7.

Additionally, the same clustering methods are used to build a shape model using groups of features. The use of shape improves robustness since it allows recovery

from tracking failures. The appearance and shape models are stacked together to form an N-tier hierarchical model, to give a richer description of the object and further improve robustness. This method is referred to as N-tier SMAT, and is also presented in Chapter 7.

Chapter 7 concludes by introducing the zig-zag-zen family of semi-automated tests to measure the performance of the tracker. The partially automated approach to measuring performance characteristics allowed the testing of eleven sequences with a mean length of 568 frames, giving more objective results than many evaluations in the literature. All the test frameworks used in this work have been made available on Internet.

The thesis is concluded in Chapter 8.

Chapter 2

Background

2.1 Previous Work

2.1.1 Tracking

Extensive research has been dedicated to tracking non-rigid objects. However, even within controlled environments, with pre-learned models and without real-time constraints, this is a difficult problem. Tracking algorithms may perhaps be grouped into three major types: object detectors, motion predictors and alignment algorithms.

Detection-type algorithms rely on training a model of a particular type of feature. A brute-force search using the model is then applied to every frame of a sequence to detect the features of interest. Detectors have become popular since Viola and Jones [91] demonstrated a face detector that was sufficiently optimised to detect faces at above real-time rates. This has spawned further human detection algorithms such as the Hockey player detector of Okuma *et al.* [61] and the human tracker of Micilotta *et al.* [55].

Since a brute-force search is made every single frame, detection-type methods tend to be robust, as there is no reliance on successful localisation in previous frames. However, the approach also has some disadvantages. Training is normally time-consuming (of the order of days), precluding the models from being updated on-the-fly. The object being tracked must be known beforehand and sufficient examples of the object must be gathered to train a model. The requirement of *a priori* data, restricts the tracker to a particular application, *e.g.* a tracker trained on faces will not track cars. The localisation of objects is coarse, as it is limited to the resolution of the brute-force search, resulting in a noisy track. Finally, identification of specific objects, *e.g. Nick*'s face versus *a* face, is difficult.

Motion predictors, such as the CONDENSATION method of Isard and Blake [31], rely on sampling an image randomly to locate a given feature. By fitting a Probability Density Function (PDF) to the locations based on the match scores, the sampling method then focuses on regions where the tracked feature is likely to be found. A motion model is fitted to inter-frame variations in the object position, allowing the position of the object to predicted in the next frame. Due to the random sampling that these methods use, they allow an explicit trade-off between speed and accuracy to be made. Motion predictors also offer good localisation and support multiple probable locations for the same object. However, the initial motion model is often inaccurate and multiple passes through the same sequence are required for the method to work [31].

Alignment algorithms are based upon maximising or minimising a similarity function between the model and the image for a set of warp parameters. Function minimisation may resemble the poor cousin of the previous two methods, because without augmentation, multiple hypotheses are not supported, and accurate registration is required in every frame during a sequence. However, alignment-methods are generally rapid to compute, and give the most precise localisation of the three

tracking approaches. Moreover, an *a priori* model is not necessary, although one is often used. Alignment or registration, also has wider applicability as a "low level vision" process, *e.g.* aligning medical images [92, 77, 42, 17], 3D object modelling [25], image-mosaicking [12, 15] and object identification [13, 26, 24, 9, 51].

Many recent approaches rely on a combination of these methods. In this work, the main focus will be on an alignment-type approach, because of the desire for real-time tracking without an *a priori* model.

2.1.2 Modelling Appearance Variation

Despite the wide applicability of alignment-based tracking algorithms and registration algorithms, there are still many open problems. In particular, many tracking algorithms unavoidably tend to drift off target or suffer from mismatch error and catastrophic failure [52, 40]. These failures occur because tracking algorithms generally rely on appearance remaining the same across multiple frames. Ambiguity caused by distractors in the scene also induce tracking error [18] as does occlusion [98, 57, 100, 60].

The use of an appearance model can partially overcome these issues, but in many cases, *a priori* data is used to build the model or train the tracker *e.g.* [61, 86]. Other methods incrementally build probabilistic appearance models to improve robustness, *e.g.* the WSL tracker of Jepson *et al.* [33]. The WSL tracker maintains a 3 component model for each pixel in the template and "shifts" between the components using Expectation Maximisation (EM). However, the use of EM combined with a large pixel-level model means that it is too expensive for real-time use. The discriminative tracker of Collins *et al.* [18] also builds a model on the fly. For robustness they use different combinations of RGB channels to form multiple features. However, the method assumes the foreground to be a rectan-

gle centred in a rectangular background region, which makes tracking irregularly shaped objects that vary in pose difficult. Also, the use of many features (125) makes real-time tracking of large objects difficult. Nguyen and Smeulders use a simpler model that does not require multiple iterations per frame [57]. Instead they apply a Kalman filter to each pixel, which yields more robust results. Even this simpler model is too slow for real-time performance.

One of the simplest and earliest approaches to localisation, suggested by Lucas and Kanade [45], has often proven to be as effective as more sophisticated approaches. Lucas and Kanade used the Sum of Squared Difference (SSD) operation to measure similarity between a representative image patch (template) and the region overlapped in the reference image, utilising a Newton-Raphson method to traverse the search space and thereby limit processing. This method is relatively accurate, particularly so in modern manifestations such as the strategic update method of Matthews *et al.* [53]. Moreover, modern formulations of registration algorithms use quasi-Newton methods and many of the more expensive tasks are pre-computed, thereby allowing above real-time performance. Notable examples are the Inverse Compositional approach of Baker and Matthews [3] and the parametric models of Hager and Belhumeur [28].

2.1.3 Organisation

The remainder of the chapter is organised as follows. In Section 2.2 the problem of registering two images is formalised. In Section 2.3 various similarity measures are introduced. Feature detection methods were also used in this work although no contributions were made in this area. These are briefly discussed in Section 2.4. A more detailed background into the formulation of the Mutual Information operation and appearance modelling methods is more appropriately given in their

respective Chapters 3 and 7.

2.2 Registration

To begin, the problem of tracking is formalised as an optimisation problem. Given a grey-scale motion sequence of N_n frames, let $f_R^{(n)}(\mathbf{x})$ represent the image intensity at position $\mathbf{x}$ in frame n. The current frame is also referred to as the *reference* image. The frame index n will often be omitted for clarity.

The tracking of a feature through a sequence is treated as a series of registrations between the (fixed) reference image, f_R, and a floating *template* image, f_T. Both images are functions of position $\mathbf{x} \in \mathbb{R}^2$, *i.e.* $\mathbf{x} = (x_1, x_2)$. Adding components to $\mathbf{x}$ allows the representation of volumetric or higher dimensional data.

Since f_R and f_T are represented as lattices of intensity at integral positions for $\mathbf{x}$, interpolation is used to obtain values at non-integral positions. The choice of interpolation method is beyond the scope of of this thesis, although it has some bearing on advanced sampling schemes, such as NP-windowing, which is discussed in in Chapter 5. Even so, the interpolation method is an important design choice and the interested reader is referred to the survey of Amidror [1].

For convenience and computational efficiency f_R is treated as infinite in extent and sampling is only performed within the bounds of the lattice of f_T. For regions outside of the defined lattice for f_R, the image is defined as 0. Hence f_T may be considered constant with respect to a warp function, and the use of complex boundary checking routines is avoided.

Each registration operation aims to align f_R and f_T, by minimising a distance function d between the two images for some warp function $\mathbf{w}$ with parameters $\mathbf{v}$:

$$\mathbf{v}_{reg} = \arg_{\mathbf{v}} \min d[f_R(\mathbf{w}(\mathbf{x}, \mathbf{v})), f_T(\mathbf{x})] \tag{2.1}$$

To maintain notational clarity $\mathbf{w}(\mathbf{x}, \mathbf{v})$ is referred to hereafter as $\mathbf{x_w}$. Distance functions that increase with greater similarity may be trivially converted to minimisation convention by multiplying them by -1.

Numerous minimisation algorithms exist, and the interested reader is referred to the Coconut Project [7] and Numerical Recipes [68] for an overview of these. A detailed analysis of what algorithms should be used is beyond the scope of this work, so only "simple" gradient descent algorithms were implemented, namely: the simplex algorithm, Powell's method, the Variable Metric Method, the Conjugate Gradients Method, and Levenberg-Marquardt's Method [68, 50]. These may be distinguished from methods using simulated annealing methods and hierarchical searches, but this choice does not imply the superiority of a particular set of methods.

Within the gradient descent methods a further distinction is made between Newton-type methods and Brent-type methods. Newton-type methods "jump" to a new position based on the local curvature and gradient *e.g.* the Levenberg-Marquardt method [50]. Brent-type methods choose a *direction* to minimise along and then perform a Brent line minimisation using a combination of the golden search and a parabolic interpolation [10], *e.g.* Powell's Direction Set method. Newton-type methods are less robust than Brent-type methods but are often faster when the Hessian (local curvature) matrix is cheap to obtain. Using the methods of Chapter 3, Hessians are obtainable, so the Levenberg-Marquardt method was favoured.

2.2.1 Warp Parameterisations

In this thesis, "warp" is a term used for any transformation between two positions, $\mathbf{x}$ and $\mathbf{x_w}$, of the same dimensionality as specified by a warp parameter $\mathbf{v}$. The dimensionality of $\mathbf{v}$ increases with the complexity of the warp. In all, four different

warps were considered and are summarised in Table 2.1. Components of $\mathbf{v}$ are referred to as v_1, v_2, *etc.*. The Jacobians of warps are typically multiplied by the gradient image, because the chain rule is required for differentiation. Hence the gradient images multiplied by the warp gradients are supplied for reference:

$$\begin{aligned}
\nabla f \frac{\partial \mathbf{w}_{tx}}{\partial \mathbf{v}} &= \begin{pmatrix} f_{x1} & f_{x2} \end{pmatrix} \\
\nabla f \frac{\partial \mathbf{w}_{eu}}{\partial \mathbf{v}} &= \begin{pmatrix} f_{x1} & f_{x2} & (f_{x1}\, f_{x2}) R'(x_1\, x_2)^T \end{pmatrix} \\
\nabla f \frac{\partial \mathbf{w}_{si}}{\partial \mathbf{v}} &= \begin{pmatrix} f_{x1} & f_{x2} & (f_{x1}\, f_{x2}) v_4 R'(x_1\, x_2)^T & (f_{x1}\, f_{x2}) R(x_1\, x_2)^T \end{pmatrix} \\
\nabla f \frac{\partial \mathbf{w}_{af}}{\partial \mathbf{v}} &= \begin{pmatrix} f_{x1}x_1 & f_{x2}x_1 & f_{x1}x_2 & f_{x2}x_2 & f_{x1} & f_{x2} \end{pmatrix}
\end{aligned}$$

where R is the standard rotation matrix:

$$R(\theta) = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}$$

The Hessians for these warps are trivial to derive and are shown in Appendix A.2. Warps are sometimes referred to by their number of Degrees of Freedom (DoF): *e.g.* 3DoF warp instead of Euclidean warp.

More complex grid based warps could have been used, such as camera homographies [30], large parameter viscous fluid model based warps used by D'Agostino [22], mesh deformation warps used by Bartoli & Zisserman [5] and Pilet [65], and surface fitting used by Torresani & Hertzmann [84] However, this thesis focuses on registering small patches, which have too little information for high DoF warps. For this reason, only the warps of Table 2.1 were implemented. Arguably, the hierarchical structural models, proposed in Chapter 7, provide equivalence to more complex warps.

Name	Dimensions	Equation for $\mathbf{x_w}$
Translation	2	$\mathbf{w}_{tx}(\mathbf{x},\mathbf{v}) = \begin{pmatrix} x_1 + v_1 \\ x_2 + v_2 \end{pmatrix}$
Euclidean	3	$\mathbf{w}_{eu}(\mathbf{x},\mathbf{v}) = \begin{pmatrix} +x_1 \cos v_3 + x_2 \sin v_3 + v_1 \\ -x_1 \sin v_3 + x_2 \cos v_3 + v_2 \end{pmatrix}$
Similarity	4	$\mathbf{w}_{si}(\mathbf{x},\mathbf{v}) = \begin{pmatrix} +x_1 v_4 \cos(v_3) + x_2 v_4 \sin(v_3) + v_1 \\ -x_1 v_4 \sin(v_3) + x_2 v_4 \cos(v_3) + v_2 \end{pmatrix}$
Affine	6	$\mathbf{w}_{af}(\mathbf{x},\mathbf{v}) = \begin{pmatrix} x_1 v_1 + x_2 v_3 + v_5 \\ x_1 v_2 + x_2 v_4 + v_6 \end{pmatrix}$

Table 2.1: The various rigid warps used in this work

2.3 Similarity Measures

When registering a pair of images, any similarity function d may be used, but several constraints affect this choice. The evaluation of d should be fast, robust to outlier pixels, and have a basin of convergence with sufficient gradient to allow rapid convergence without giving ambiguous results. The ability to obtain a Hessian rapidly is also a consideration.

Many similarity metrics are available for the comparison of an appearance model (template) and the region of overlap in the reference image (current frame), and subsequent optimisation. For the localisation of image patches, SSD is probably the most widely used measure [45, 28, 40, 3, 99]. Perhaps this is so, because SSD was shown to be rapid to compute and easily differentiated, when it was introduced for image registration by Lucas and Kanade [45]. SSD is also simple to implement and has a wide basin of convergence, aiding optimisation. As the name implies, the SSD function is simply a normalised sum of the squared

differences between corresponding pixels in the pair of images:

$$d_{ssd}(\mathbf{v}) = \frac{1}{N_{\mathbf{x}}} \sum_{\mathbf{x}} [f_r(\mathbf{w}(\mathbf{x}, \mathbf{v})) - f_t(\mathbf{x})]^2 \tag{2.2}$$

where $N_{\mathbf{x}}$ is the number pixels defined by the lattice f_t.

However, SSD is not tolerant of the outlying pixel values caused by noise and occlusions. SSD assumes a linear relationship between the intensities of the two images, making it intolerant of changes in lighting conditions. An example of an SSD function surface is shown in Figure 2.1a.

Normalised Correlation (NC) has been proposed to surmount the problems caused by outlying pixel intensities with some success, and is also popular for registration applications, *e.g.* Ukrainitz and Irani aligning sequences in space and time [89]. Levin and Weiss [43] use NC to register object fragments as one step in segmenting an object from the background. Normalised Correlation relies on computing the products of pixel values. As a result it is more robust to lighting changes since outliers have less influence on the product operation.

$$d_{nc}(\mathbf{v}) = \frac{\sum_{\mathbf{x}} f_R(\mathbf{x}) f_T(\mathbf{x_w})}{(\sum_{\mathbf{x}} f_R(\mathbf{x})^2)^{\frac{1}{2}} (\sum_{\mathbf{x}} f_T(\mathbf{x_w})^2)^{\frac{1}{2}}} \tag{2.3}$$

NC also has a wide basin of convergence, as shown in Figure 2.1b. However, it still assumes a linear relationship between the intensities of the template and reference images, and the wide basin of convergence can make convergence slow.

Mutual Information (MI) has proven to be a superior metric in many respects to both SSD or NC. Since its concurrent introduction and popularisation by Viola & Wells [92], Studholme *et al.* [77] and Collignon *et al.* [17], it has been widely adopted, particularly in the medical imaging field. MI has seldom been applied to tracking problems in the literature, because the limited number of intensity samples available in small image patches leads to over-sparse histograms and

inaccurate estimates of entropy. Reducing the histogram resolution solves this issue (*e.g.* 8×8 bins for an 11×11 template worked well), but this also assumes that the tracked feature contains sufficient intensity variation (*i.e.* gradients at this discretisation are finite). If the assumption of sufficient gradient is not met, SSD and NC will perform better, as they only have a single degree of freedom and hence a higher sensitivity to intensity variations.

2.3.1 Mutual Information

Mutual Information (MI) is only slightly more expensive than SSD or NC to compute, and was favoured in this work, due to its known robustness to noise, occlusions and environmental lighting conditions [93]. MI also has a sharper peak leading to more specific localisation. In a comparative study of registration methods, an MI based algorithm outperformed 15 other algorithms [97]. The choice of similarity metric and descriptor patch often has a large effect on performance, whatever the final application. Hence, the contributions made in this work to understanding MI should be of wide interest.

The origins of MI are in information theory[20]. It was first proposed by Shannon [71] as method to measure the entropy of the shared information between two signals, with quantised *amplitudes* over a period of *time*. It is a simple extension to consider 2D images rather than 1D signals, which consist of quantised *intensities* over a 2D *space*.

MI relies on computing a joint-histogram of intensities, and as a result, is tolerant of non-linear relationships between the intensities in images. Chapter 3 discusses MI in detail, but the MI equation is given here for reference:

$$d_{mi} = \sum_{r,t} p_{rt}(r,t) \log \left(\frac{p_{rt}(r,t)}{p_r(r) p_t(t)} \right) \tag{2.4}$$

where p_r, p_t, and p_{rt}, are the respective Probability Density Function (PDF) estimates of intensity for the reference, template and joint of the two images, while r and t are the reference and template intensities. Note how (2.4) does not follow the somewhat traditional form of distance function in (2.1), where a one-to-one functional relationship between intensities in f_R and intensities in f_T is assumed. Rather the statistical relationship between the images, summarised using a histogram, is used.

Multiple forms of MI have been developed [67], of which four examples are shown in Figure 2.1b-f. The main difference between MI methods is how the joint-histogram is constructed from image intensity samples. One approach, used by Wells *et al.* [96] is to convolve the histogram with a Parzen window [63], to account for uncertainty in the intensity values. Thevenaz and Unser use a more sophisticated approach to Parzen windowing [81] using B-splines, which is applied during the construction of the histogram, giving more accurate results.

In addition, Partial Volume Interpolation (PVI) was introduced by Maes *et al.* [47], which increments several histogram bins for each sample based on the distance of the sample point from the surrounding pixels. Chen and Varshney extended this concept to Generalised Partial Volume Estimation, which uses extended spatial support [14].

Mutual Information has been applied as a similarity measure using a number of different optimisation methods with varying degrees of success, including the simplex algorithm [54], Powell's method [17, 47], Gradient Descent [96], hierarchical brute-force searches [78] and hierarchical approaches [81, 32]. A number of these methods were systematically compared by Maes in [48].

Few of the available MI methods use an analytic derivative, precluding the use of Newton-type methods. The analytic derivative is difficult to obtain because

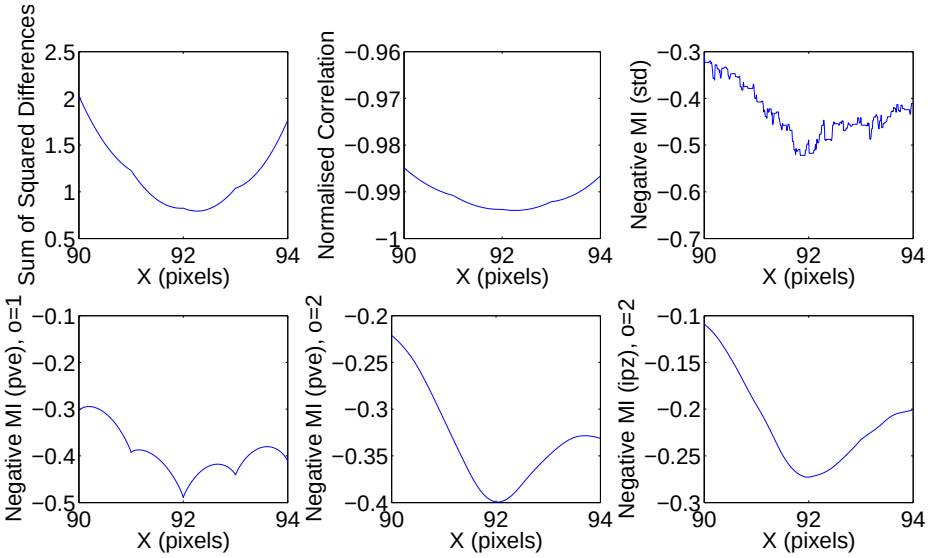


Figure 2.1: Some examples of function surfaces for SSD, NC and several MI measures, as a template is translated in the X direction across the reference image. The same template is used in each case, with the ground truth at 92 pixels.

of the non-linear flooring functions implicit to the histogramming process used in calculating MI. Notable exceptions are a Jacobian for Partial Volume Interpolation by Maes *et al.* [48], a Jacobian for standard sampled MI by Crum *et al* [21], and a Jacobian and Hessian for MI using B-spline Parzen windowing by Thevenaz & Unser [81]. A general derivation for Jacobians and Hessians for the four common types of MI has been developed in this work and is discussed in Chapter 3. Figure 2.1 shows some examples of function surfaces for SSD, NC and several MI measures.

All forms of MI are prone to the effect of artefacts to some degree, and these have been under examination for some time [46], with a study by Pluim *et al.* of artefacts [66] probably being the most comprehensive review to date. Attempts to better explain these phenomena have been made by several authors. Tsao

published the results of a large number of empirical tests in [88]. Some theoretical explanations have also been given, like that of Moddemeijer [56], who showed that the variance of an MI estimate is dependent on the number of samples and the number of intensity bins. Ji *et al.* extended this work and related it back to the issue of interpolation artefacts [34].

Artefacts can cause optimisation algorithms to converge slowly, necessitate manual pre-alignment close to the correct position, and result in the algorithm becoming trapped in local maxima [48, 66]. More seriously, even if the optimisation algorithm converges to the global maximum, this maximum may have been shifted slightly due to bias. When tracking small image patches, the limited number of samples makes the effect of artefacts particularly evident.

2.4 Feature Detection

Before leaving this Chapter, one important issue for any tracking algorithm should be addressed: the detection of features. Although for some applications it is adequate for a human operator to select features by hand, in many cases this will lead to poor results. Firstly, automated algorithms are more practical when large numbers of features must be selected. Secondly, a human operator may not necessarily select features that a good to track, since the nature of trackable features will depend on the similarity method used.

Tomasi and Kanade deem good features to be those that have a Hessian (obtained by multiplying the gradient vector by its transpose) with sufficiently large eigenvalues [82]. This implies that the image gradient is above the noise level of the image and is well conditioned, *i.e.* is not ambiguous in the x or y direction. Harris and Stephens [29] independently (and earlier) came to a similar conclu-

sion and applied the same approach to very local (3×3) image patches to detect corners and edges. The result is a score for "edginess" using a linear combination of the (3×3) patch intensities with precomputed weights. The Harris corner detector is widely used when detecting features for SSD trackers.

One of the problems with the Harris corner detector is that it only considers small local regions. In fact scale is also an important consideration when selecting salient features to track. Kadir and Brady suggested using entropy as a measure of saliency for regions that considered scale [36]. This has more recently been extended to have affine invariance [39].

Lowe also considers scale [44], and passes multiple difference of Gaussian filters over a pyramid of images to detect local intensity extrema. This serves to extract regions that are uniform compared to their surroundings. A histogram based representation of each region is used to make the selected features invariant to transformations (SIFT). Obdrzalek and Matas [59, 51] use a similar approach in selecting Maximally Stable Extremal regions. Transformation invariance is achieved by using the position of maximum curvature to define a frame of reference.

Feature detection was not within the scope of this thesis, and methods were used as published. The saliency detector of Kadir is favoured in this thesis, because like the Mutual Information metric, it is entropy based and hence a better match for the measure. If SSD or NC are used as similarity measures, then the Harris corner detector, or the more advanced SUSAN detector of Smith and Brady [76], would make a better choice.

Chapter 3

Mutual Information

In this chapter, the four most common variants of Mutual Information are placed into a single mathematical framework. In Section 3.1, it is demonstrated that these four variants, namely: standard sampling, Partial Volume Estimation, In-Parzen Windowing and Post-Parzen Windowing; vary only in how the joint histogram is constructed from image samples. Jacobians and Hessians are derived for all these methods, using an approach similar to that of Thevenaz and Unser [81], but who considered In-Parzen Windowing only. In Section 3.2, this is extended to standard sampling, post-Parzen window estimation and the general form of partial volume estimation. Using the established framework, the methods are compared in terms of computational cost, in Section 3.3. Finally, in Section 3.5, the MI methods are tested and compared in terms of convergence and bias using 144,000 tests, before concluding in Section 3.6. The symbolic conventions used in this chapter follow those defined in the Section 2.2 on registration.

3.1 The families of Mutual Information

3.1.1 Histogram Estimation

A measure of the information mutual to f_T and the corresponding region in f_R is obtained from the joint intensity histogram $h(r, t, \mathbf{v})$ of the two images. The symbols $r \in [0; r_{mx}] \cap \mathbb{Z}$ and $t \in [0; t_{mx}] \cap \mathbb{Z}$ index the intensities that f_R and f_T respectively consist of ($\mathbb{Z}$ is the set of integers).

The histogram may be normalised to give an approximation of the PDF of intensities, *i.e.* $p_{rt}(r, t, \mathbf{v}) = \frac{1}{N_\mathbf{x}} h(r, t, \mathbf{v})$, where $N_\mathbf{x}$ is the number of samples used to construct the histogram. MI is defined in terms of p rather than h for clarity, and the dependence on $\mathbf{v}$ is explicitly indicated:

$$d_{mi}(\mathbf{v}) = -\sum_{r,t} p_{rt}(r, t, \mathbf{v}) \log \left(\frac{p_{rt}(r, t, \mathbf{v})}{p_r(r, \mathbf{v}) p_t(t)} \right) \tag{3.1}$$

A frequently used form of (3.1) has three entropy terms: $d_{mi} = H_r + H_t - H_{rt}$. This form gives the same result, but the condensed form above is used for conciseness. Mutual Information increases with increasing similarity between f_t and the corresponding region in f_r, so to maintain the convention of referring to function *minimisation* the negative MI value is used, *i.e.* the negative sign is removed from (3.1).

The PDFs p_r and p_t are easily obtained from the joint PDF, since $p_r = \sum_t p_{rt}$, and $p_t = \sum_r p_{rt}$. Note the treatment of r and t as discrete variables (or indices), indicating the finite bin-size of the histogram h from which p is derived. MI is *not* invariant to the bin-size Δi, which limits its bounds, as does the number of sample points: $d_{mi} \leq \log(\min(\frac{r_{mx}}{\Delta i}, \frac{t_{mx}}{\Delta i}, k_{mx} N_\mathbf{x}))$, where k_{mx} indicates the number of histogram bins populated per sample. The joint histogram is defined in terms

of two window functions $\psi()$, which act as membership functions:

$$h(r,t,\mathbf{v}) = \sum_{\mathbf{x}} \psi\left(r - \frac{f_R(\mathbf{x_w})}{\Delta i}\right)\psi\left(t - \frac{f_T(\mathbf{x})}{\Delta i}\right) \tag{3.2}$$

Each sample taken from f_R and f_T is added to one histogram bin:

$$\psi(\epsilon) = \beta_0^-(\epsilon) = \begin{cases} 1 & 0 < \epsilon < 1 \\ 0 & \text{otherwise} \end{cases} \tag{3.3}$$

This kind of sampling is referred to as *standard sampling*. The $\beta()$ function in the above equation comes from the B-spline family of functions, and a brief digression describing these is now made.

3.1.2 B-splines

B-spline functions are a family of functions with several useful properties. Firstly, the sum of a B-spline function for all integral distances from a real value is one, *i.e.* it has a partition of unity. This means that no additional renormalisation is required when histogramming. Secondly, the integral of a B-spline is one. Thirdly, order n B-splines are the convolution of any set of B-splines whose order sums to n. Lastly, the derivative of an order n B-spline is the difference between two, offset, order $n-1$ B-splines. These properties are summarised below.

$$\begin{aligned} \sum_{a\in\mathbf{Z}} \beta(\epsilon + a) &= 1 \qquad \epsilon \in \mathbb{R} \\ \int_{\epsilon\in\mathbf{R}} \beta(\epsilon) &= 1 \\ \beta_n(\epsilon) &= \beta_{n-1}(\epsilon) * \beta_0(\epsilon) \\ \frac{\partial \beta_n}{\partial \epsilon} &= \beta_{n-1}(\epsilon + \frac{1}{2}) - \beta_{n-1}(\epsilon - \frac{1}{2}) \end{aligned}$$

The $0th$ order B-spline β_0 is simply a top hat function, centred about 0, *i.e.* $\beta_0(\epsilon) = 1$ when $|\epsilon| \leq \frac{1}{2}$ and 0 otherwise. The offset top-hat functions $\beta_0^- =$

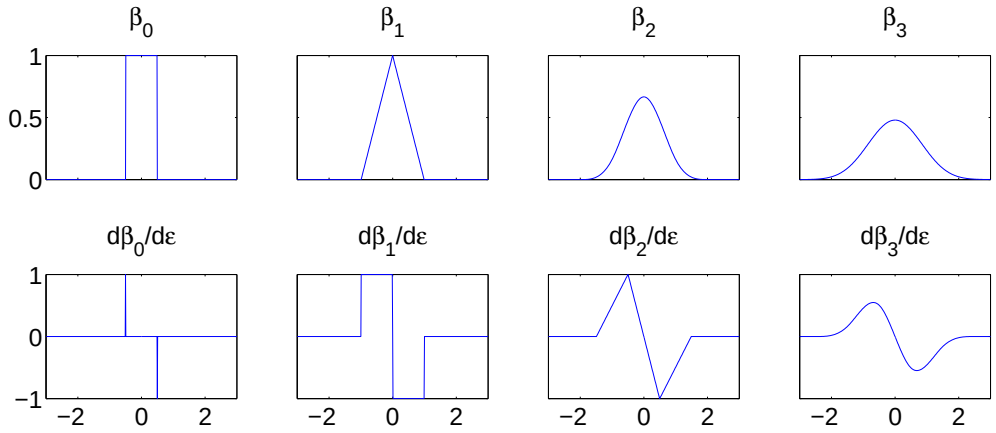


Figure 3.1: The first four B-spline functions are shown in the top row along with their respective derivatives in the bottom row. B-spline order increases from left to right. Note how each B-spline is the convolution of the previous B-spline with the zeroth order B-spline. Note also, that each derivative is the difference between two order $n-1$ B-splines offset from each other by one. Higher order B-splines can be constructed recursively in exactly the same manner.

$\beta_0(\epsilon-\frac{1}{2})$ and $\beta_0^+ = \beta_0(\epsilon+\frac{1}{2})$ are also defined. Some examples of B-spline functions and their derivatives are given in Figure 3.1. A more detailed description of B-spline functions and their numerical computation is given by Unser *et al.*in [90].

3.1.3 Choice of Δi

When using MI as a similarity metric, the size of the bins in the joint-histogram, Δi, must be chosen. The choice affects the computational cost for evaluating MI (see ancillary costs in Table 3.3), but more importantly it affects how meaningful the MI measure is. Too small a Δi, results in too many bins relative to the number of samples, $N_{\mathbf{x}}$, and an underpopulated histogram that does not describe how intensity values are clustered. The practical effect of an underpopulated histogram

is uniformly low MI values, with the position of match being indistinguishable from the surrounding surface, making optimisation difficult or impossible.

In this work, the relationship established by Hadjidemetriou *et al.* [27] was used, where they recommended that the number of bins, $N_i = \frac{256}{\Delta i}$, be proportional to the cube root of the number of samples:

$$N_i \propto N_{\mathbf{x}}^{\frac{1}{3}} \Rightarrow \Delta i \propto N_{\mathbf{x}}^{-\frac{1}{3}} \tag{3.4}$$

The experiments of Chapters 4 and 5 established that for an 11×11 template, a Δi of 32 worked well, allowing a constant of proportionality to be established.

3.1.4 Different Sampling Methods

For Standard Sampling (STD), (3.2) and (3.3) show that for each of the $N_{\mathbf{x}}$ lattice points in f_T, each histogram in h_{rt} is incremented once. For reference the windowing function for STD is restated in (3.5a), where f is an image, i is an intensity index, Δi is the bin size of the histogram, and $\mathbf{x}$ is a sample point. An explanation of each sampling function below follows, and each method is illustrated in Figure 3.2.

$$\psi_{std}(i - f(\mathbf{x})) = \beta_0^-(i - \frac{f(\mathbf{x})}{\Delta i}) \tag{3.5a}$$

$$\psi_{pve}^{(o)}(i) = \sum_{\mathbf{x}' \in \mathbb{Z}^2} \beta_n(\mathbf{x} - \mathbf{x}')\beta_0^-(i - \frac{f(\mathbf{x}')}{\Delta i}) \tag{3.5b}$$

$$\psi_{ppz}^{(o)}(i) = \beta_o^-(i - \frac{\text{floor}(f(\mathbf{x}))}{\Delta i}) \tag{3.5c}$$

$$\psi_{ipz}^{(o)}(i) = \beta_o^-(i - \frac{f(\mathbf{x})}{\Delta i}) \tag{3.5d}$$

Partial Volume Estimation (PVE), introduced by Maes as Partial Volume Interpolation (PVI) in [47], aims to make shifts between histogram bins smooth as the

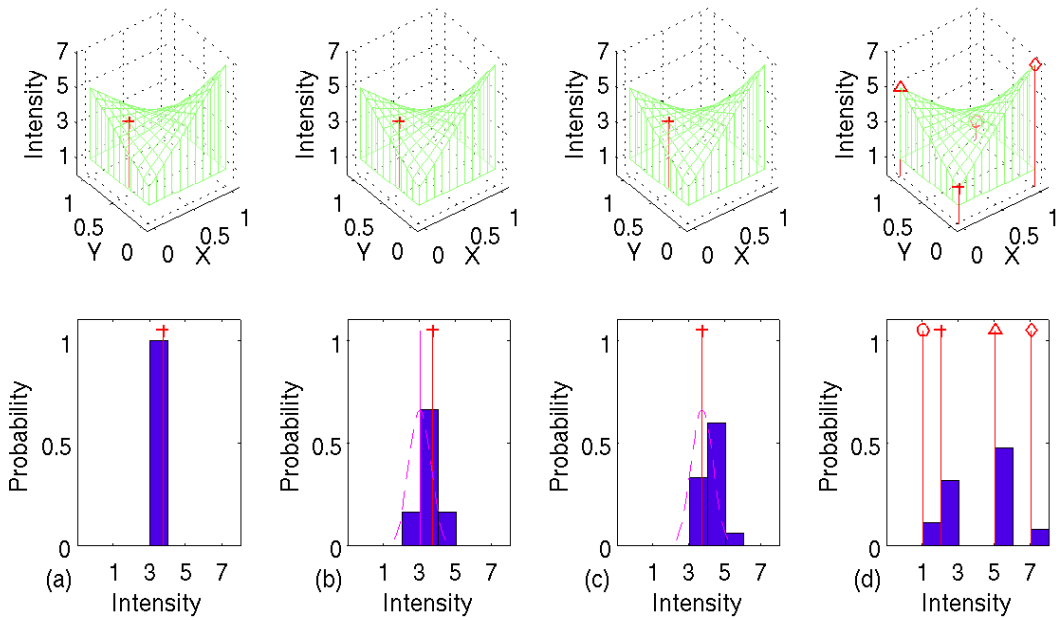


Figure 3.2: Illustration of various Sampling Methods. **(a)** standard sampling extracts a single interpolated value and populates a single histogram bin. **(b)** shows Post Parzen windowing, which is similar to (a) except the histogram is blurred after the bins have been populated. **(c)** shows In-Parzen windowing, which differs from (b) in that the blurring occurs during histogram construction, *i.e.* before the intensity value is binned. Finally, **(d)** shows Partial Volume Estimation, where the values of the surrounding lattice intensities are used to populate the histogram.

parameters of $\mathbf{v}$ vary. PVE has the added advantage of not adding any (possibly false) information other than the given data. The method involves populating the intensity histogram bins of the four lattice points surrounding each sample by a weighted amount. The weighting is proportional to the area of overlap between the square regions around the sample and lattice points. This is equivalent to integrating the region of influence of each neighbouring pixel when using nearest neighbour interpolation: *i.e.* $h(i) = \int_{\mathbf{x}} \beta_0^-(i - f_{nni}(\mathbf{x}))$.

Chen and Varshney extended partial volume interpolation to generalised Partial Volume Estimation (PVE) by using higher order B-splines to weight a larger region of pixels [14]. Although PVE techniques have been treated as alternative *interpolation* methods, strictly speaking they are alternative *sampling* methods. Hence the term "PVE" being used, rather than "PVI".

The windowing function for PVE is given in (3.5b), where $\mathbf{x}'$ are the coordinates of all lattice points in the image and n indicates the order in the sampling family. Note that (3.5b) collapses to (3.5a) with nearest neighbour interpolation for $o = 0$, since in this case only one valid value for $\mathbf{x}'$ exists, that of the lattice point nearest to $\mathbf{x}$. For notational clarity, the first window function in (3.5b) is shown to take a vector as an input. This is simply a product of two window functions, one for each dimension of the vector, *i.e.* $\beta_o(\mathbf{x}) = \prod_{k=1}^{K} \beta_o(x_k)$, where K is the number of components in the vector $\mathbf{x}$.

The advantages of PVE are that it only adds information explicitly given in the image, it is relatively inexpensive to compute and has a smooth surface. The disadvantage is that for orders below 2, PVE is only C_1 smooth with cusps at points of $\mathbf{v}$ where grid-alignment between f_t and f_r occur. A strong bias towards these cusped positions exists. Also, a nearest neighbour model of the world ignores much of the information implicit to the image. Note, that C_n smooth

means that the nth derivative of the function is continuous at all points.

Two types of Parzen windowing routines exist: *Post-Parzen* Windowing (PPZ) and *In-Parzen* windowing (IPZ). In PPZ, the histogram is constructed first and then convolved with a Parzen window. In IPZ, each sample is convolved during histogram construction. IPZ uses a sample's intensity value before the information loss implicit to discretisation occurs.

The window equation for post-Parzen windowing is given in (3.5c), where floor($f(\mathbf{x})$) indicates reduction to the first integer value below $f(\mathbf{x})$. The floor() function could have been shown as a sum of windowing functions, but this would make the equation unwieldy. (3.5c) shows an oth order B-spline window function, but any window function may be used. B-splines were used since they are inexpensive and their derivatives are easily obtainable [90]. The advantage of post-Parzen windowing is that it improves on the basic sampling method using a computationally cheap operation: $O(i_{\max}^2 w^2)$. However, there is information loss due to blurring of the histogram, and the function is not necessarily smooth.

In-Parzen windowing differs slightly, in that it lacks the implicit discretisation of intensity values as shown in (3.5d). As a result, In-Parzen windowing has a guaranteed C_{o-1} smooth function surface and a more accurate histogram. Again some information loss occurs due to blurring of the histogram, and the method is comparatively expensive. Both (3.5c) and (3.5d) collapse to (3.5a) for $o = 0$.

3.2 Jacobians and Hessians

The Jacobian of MI may now be found by applying the product and chain rules to (3.1) and collecting the terms together:

$$\frac{\partial d_{mi}}{\partial \mathbf{v}} = \sum_{r,t} \frac{\partial p_{rt}}{\partial \mathbf{v}} \left(1 + \log\left(\frac{p_{rt}}{p_r}\right) - \log(p_t)\right) - \frac{p_{rt}}{p_r}\frac{\partial p_r}{\partial \mathbf{v}}$$

$N_{\mathbf{x}}$ is assumed to be constant for a given template. A more general definition of the above equation has been given by Thevenaz [81], where a non-constant $N_{\mathbf{x}}$ was accounted for. However their approach constructs the problem such that $N_{\mathbf{x}}$ is constant anyway. Directly adapting [81], to make this assumption at the start of the derivation (using assumption of a bounded template and a reference image of infinite extent, made in Section 2.2) simplifies the remaining steps considerably.

The summations in the fourth (last) term may be split to give $\sum_r \frac{1}{p_r} p'_r \cdot \sum_t p_{rt}$, since p_r and p'_r are not dependent on t. However $\sum_t p_{rt} = p_r$ since it is a sum of a joint histogram. So the fourth term becomes $\sum_r p'_r$, because p_r^{-1} and p_r cancel. However, because $N_{\mathbf{x}}$ is constant and p is based on a histogram, $\sum p$ always equals one, and therefore $\sum p'$ always equals zero, so this term disappears.

Also if the third term ($\sum_{r,t} p'_{rt} \log(p_t)$) is separated out, the summations may again be split to get $\sum_t \log(p_t) \sum_r p'_{rt}$. But $\sum_r p'_{rt} = p'_t$ is zero as the template is constant, so the third term also disappears.

The remaining two terms are combined and the derivative of MI becomes:

$$\frac{\partial d_{mi}}{\partial \mathbf{v}} = \sum_{r,t} \frac{\partial p_{rt}}{\partial \mathbf{v}} \log\left(\frac{e p_{rt}}{p_r}\right) \tag{3.6}$$

3.2.1 Derivative of histogram function

The derivative of the histogram function may be obtained using the chain rule:

$$\begin{aligned}\frac{\partial p_{rt}}{\partial \mathbf{v}} &= \frac{\partial}{\partial \mathbf{v}}\frac{1}{N_{\mathbf{x}}}\sum_{\mathbf{x}}\psi[t-\frac{f_T(\mathbf{x})}{\Delta i}]\psi[r-\frac{f_R(\mathbf{x_w})}{\Delta i}] = \frac{1}{N_{\mathbf{x}}}\sum_{\mathbf{x}}\psi_T[t-\frac{f_T(\mathbf{x})}{\Delta i}]\frac{\partial}{\partial \mathbf{v}}\psi_R[r-\frac{f_R(\mathbf{x_w})}{\Delta i}] \\ &= \frac{1}{N_{\mathbf{x}}\Delta i}\sum_{\mathbf{x}}\psi_T\frac{\partial\psi_R}{\partial\epsilon}\frac{\partial\epsilon}{\partial f_R}\frac{\partial f_R}{\partial \mathbf{w}}\frac{\partial \mathbf{w}}{\partial \mathbf{v}} = -\frac{1}{N_{\mathbf{x}}\Delta i}\sum_{\mathbf{x}}\psi_T\frac{\partial\psi_R}{\partial\epsilon}\nabla f_R\frac{\partial \mathbf{w}}{\partial \mathbf{v}} \end{aligned} \tag{3.7}$$

The derivatives for the reference window functions differ for each sampling method. The intensities are indicated by r since only a derivative for the reference image is required:

$$\frac{\partial\psi_{std}}{\partial\epsilon}(r) = \delta(r-\frac{f_R(\mathbf{x_w})}{\Delta i}) - \delta(r-1-\frac{f_R(\mathbf{x_w})}{\Delta i}) \tag{3.8a}$$

$$\frac{\partial\psi^{(o)}_{pve}}{\partial\epsilon}(r) = \sum_{\mathbf{x}}{}' \left(\beta^+_{o-1}(\mathbf{x_w}-\mathbf{x}') - \beta^-_{o-1}(\mathbf{x_w}-\mathbf{x}')\right)\beta^-_0(r-\frac{f_R(\mathbf{x}')}{\Delta i}) \tag{3.8b}$$

$$\frac{\partial\psi^{(o)}_{ppz}}{\partial\epsilon}(r) = \left(\beta^+_{o-1}(r-\frac{f_R(\mathbf{x_w})}{\Delta i}) - \beta^-_{o-1}(r-\frac{f_R(\mathbf{x_w})}{\Delta i})\right)\sum_{m\in\mathbb{Z}}\delta(r-m) \tag{3.8c}$$

$$\frac{\partial\psi^{(o)}_{ipz}}{\partial\epsilon}(r) = \left(\beta^+_{o-1}(r-\frac{f_R(\mathbf{x_w})}{\Delta i}) - \beta^-_{o-1}(r-\frac{f_R(\mathbf{x_w})}{\Delta i})\right) \tag{3.8d}$$

It should also be noted that for PVE the ∇f factor in (3.7) should be removed, since ψ_{pve} does not depend on $f_R(\mathbf{x_w})$, but on $\mathbf{x_w}$. Apart from this difference, note how similar the structures of all these equations are. Note also how the δ functions in $\partial_\epsilon\psi_{std}$ and $\partial_\epsilon\psi_{ppz}$ imply that the function is constant at most points, except at certain $\mathbf{v}$ positions on the cost function surface where a step change occurs. This exactly mirrors reality.

In these cases (STD and PPZ) the derivative function surface is a zero plane populated by impulse functions, and the analytic derivative supplies almost no

information to the optimisation function, resulting in failed convergence. Hence it is better to use the approximate derivative $\frac{\partial \psi_R}{\partial \epsilon} \approx \frac{\Delta \psi_R}{\Delta \epsilon}$:

$$\frac{\partial \psi_{std}}{\partial \epsilon}(r) \approx \beta_0^-(r - \frac{f_R(\mathbf{x_w})}{\Delta i}) - \beta_0^-(r - 1 - \frac{f_R(\mathbf{x_w})}{\Delta i}) \tag{3.9a}$$

$$\frac{\partial \psi_{ppz}^{(o)}}{\partial \epsilon}(r) \approx \left(\beta_{o-1}^+(r - \frac{f_R(\mathbf{x_w})}{\Delta i}) - \beta_{o-1}^-(r - \frac{f_R(\mathbf{x_w})}{\Delta i})\right) \sum_{m \in \mathbb{Z}} \beta_0^-(r-m) - \beta_0^-(r-m-1) \tag{3.9b}$$

Crum *et al.* also obtained the result in (3.9a) for the MI gradient for standard sampling [21], but via a different route. Crum *et al.* examined the fractional variation in the neighbouring bins of a given intensity caused by perturbating each warp parameter. The limit as the perturbations shrank to zero was taken. Since registration in [21] was achieved using a viscous fluid model, only the first order gradient was necessary and the second order gradient (Hessian) was not derived. In the following subsection the Hessian is derived for all four MI families discussed thus far.

3.2.2 MI Hessian

The MI Hessian is approximated to:

$$\begin{aligned} \frac{\partial d_{mi}^2}{\partial v_1 \partial v_2} &= \sum_{r,t} \left(\frac{\partial p_{rt}}{\partial v_1} \frac{\partial p_{rt}}{\partial v_2} \frac{1}{p_{rt}} - \frac{\partial p_r}{\partial v_2} \frac{\partial p_{rt}}{\partial v_1} \frac{1}{p_r} + \frac{\partial p_{rt}^2}{\partial v_1 \partial v_2} \log\left(\frac{e p_{rt}}{p_r}\right) \right) \\ &= \sum_{r,t} \left(\frac{\partial p_{rt}}{\partial v_1} \frac{\partial p_{rt}}{\partial v_2} \left(\frac{1}{p_{rt}} - \frac{1}{p_r} \right) \right) \end{aligned} \tag{3.10}$$

because in the second term $\sum_t \frac{\partial p_{rt}}{\partial v_2} \frac{\partial p_r}{\partial v_1} \frac{1}{p_r} = \sum_t \frac{\partial p_{rt}}{\partial v_2} \frac{\partial p_{rt}}{\partial v_1} \frac{1}{p_r}$. The third term is approximately zero near the minimum. Its use only improves the speed of optimisation slightly at considerable computational expense. In Chapter 6 the implications of this approximation are examined in more detail.

3.3 Computational Costs

The computational costs of sampling methods are important when selecting which one to use for a particular application. Also, MI is sometimes regarded as an expensive option compared to say sum of square differences or normalised correlation. This subsection shows that this is not necessarily true.

The SSD operation is $O(N_{\mathbf{x}})$: each operation requiring a warp, a template pixel access and multiple reference pixel accesses for interpolation. For MI, the only additional cost is to access each bin in the histogram after its construction, *i.e.* MI is $O(N_{\mathbf{x}} + t_{mx} r_{mx})$. More sophisticated MI methods require multiple bin updates per sample, which can also increase computational cost. Theoretical estimates of costs for each function are given in Table 3.3.

The costs of calculating the Jacobian would appear to be substantially higher, since one histogram per warp parameter is required. However, there is some redundancy between the gradient and function evaluations, so this increase is not substantial. Likewise for the Hessian. Depending on how the registration problem is formulated, there are many opportunities for pre-processing to reduce costs substantially. In Chapter 6 such techniques are considered in detail.

Some empirical tests were performed to verify the predictions of computational cost, the results of which are given in Figure 3.3. The mean time to evaluate each function 10000 times is displayed for six template sizes in units of μs per pixel. There is some overhead to each function evaluation, but this becomes negligible as the size of the template increases, and the mean cost per pixel reaches a steady state value.

Table 3.1: The computational complexity of various similarity methods, using Nearest Neighbour Interpolation (NNI), Bi-Linear Interpolation (BCI) and Bi-Cubic Interpolation (BCI). Standard sampled MI is only slight more expensive than SSD to compute.

Function	Order (o)	Interp. Method	Reads of f_t+f_r	Writes updates	Ancillary
SSD	n/a	NNI	$N_{\mathbf{x}}$(1+1)	$N_{\mathbf{x}}$	
SSD	n/a	BLI	$N_{\mathbf{x}}$(1+4)	$N_{\mathbf{x}}$	
SSD	n/a	BCI	$N_{\mathbf{x}}$(1+16)	$N_{\mathbf{x}}$	
MI(std)	n/a	NNI	$N_{\mathbf{x}}$(1+1)	$N_{\mathbf{x}}$	$t_{mx}r_{mx}$
MI(std)	n/a	BLI	$N_{\mathbf{x}}$(1+4)	$N_{\mathbf{x}}$	$t_{mx}r_{mx}$
MI(std)	n/a	BCI	$N_{\mathbf{x}}$(1+16)	$N_{\mathbf{x}}$	$t_{mx}r_{mx}$
MI(pve)	1st	n/a	$N_{\mathbf{x}}$(1+4)	$4N_{\mathbf{x}}$	$t_{mx}r_{mx}$
MI(pve)	2nd	n/a	$N_{\mathbf{x}}$(1+9)	$9N_{\mathbf{x}}$	$t_{mx}r_{mx}$
MI(pve)	3rd	n/a	$N_{\mathbf{x}}$(1+16)	$16N_{\mathbf{x}}$	$t_{mx}r_{mx}$
MI(ipz)	1st	BLI,BCI	$N_{\mathbf{x}}$(1+(4,16))	$4N_{\mathbf{x}}$	$t_{mx}r_{mx}$
MI(ipz)	2nd	BLI,BCI	$N_{\mathbf{x}}$(1+(4,16))	$9N_{\mathbf{x}}$	$t_{mx}r_{mx}$
MI(ipz)	3rd	BLI,BCI	$N_{\mathbf{x}}$(1+(4,16))	$16N_{\mathbf{x}}$	$t_{mx}r_{mx}$
MI(ppz)	1st	BLI,BCI	$N_{\mathbf{x}}$(1+(4,16))	$N_{\mathbf{x}}$	$4t_{mx}r_{mx}$
MI(ppz)	2nd	BLI,BCI	$N_{\mathbf{x}}$(1+(4,16))	$N_{\mathbf{x}}$	$9t_{mx}r_{mx}$
MI(ppz)	3rd	BLI,BCI	$N_{\mathbf{x}}$(1+(4,16))	$N_{\mathbf{x}}$	$16t_{mx}r_{mx}$

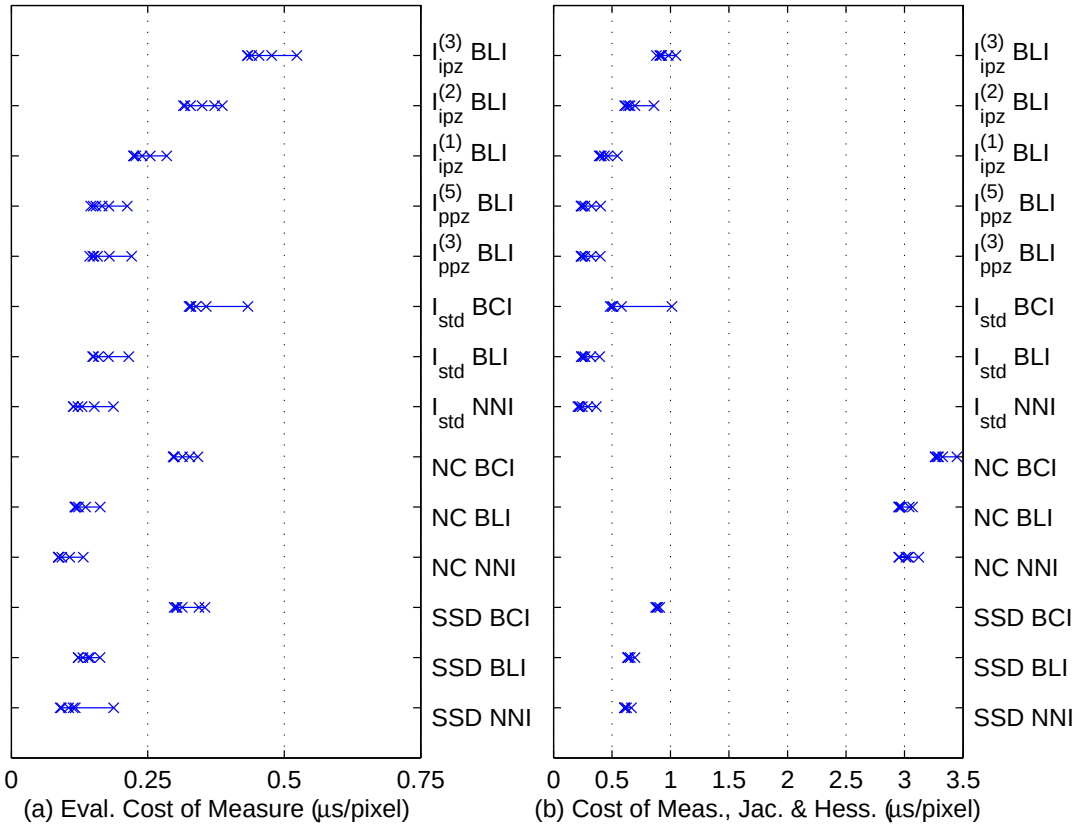


Figure 3.3: Computational cost when evaluating (a) similarity functions and (b) their Jacobians as well. For each function the mean time per function evaluation is given for 10000 evaluations. Individual crosses indicate the results for six template sizes ($15^2, 20^2, 30^2, 40^2, 50^2, 60^2$ pixels). Results are given in units of μs per pixel. Each function has a some overhead, but as the number of pixels in the template increases the realtive overhead becomes negligible. Hence, the cost per pixel asymptotes to the minimum cost shown.

Figure 3.4: The eight data sets used for testing the similarity metrics. The templates are the green shaded images in the upper left corner in each case, and the corresponding region in each image is indicated as a small green shaded region. The resolution is approximately three times higher than what was used in testing.

3.4 Test Setup

A series of experiments was performed to evaluate the ability of each MI family to converge to a ground truth position. For comparison the MI methods were also compared to the Sum of Squared Differences metric and to Normalised Correlation. Analytic Jacobians and Hessians were used in all cases. The derivatives for SSD are given in Section 6.1.1, as they are more relevant to the material of Chapter 6 than this Chapter. The derivatives for NC are large expressions and were not found in the literature, so these are given in Appendix A.1. The PPZ and IPZ variants of MI with third order Parzen windows were tested, while second order PVE was used. Although first order PVE (also known as Partial volume Interpolation) is more widely used at this time, it exhibits significant artefacts which negatively impact registration performance.

A standard test set was used for the testing in this and the following three chap-

ters. The test set consisted of the eight high resolution (2560×1920) images shown in Figure 3.4. In each case the full image was used as a reference and small sub-regions were selected as templates to be matched to the original. The images were blurred with a normalised 12×12 top-hat function before being down-sampled by 12 times.

Pairs of templates were extracted from each image. The first template was chosen to align exactly with the lattice of the down-sampled reference image. The second template was translated relative to the first by $\frac{1}{3}$ of a down-sampled pixel (or 4 normally sampled pixels) in each direction. The two templates in each pair are respectively referred to as a *lattice-aligned* template and a *lattice-offset* template.

Pairs of templates were chosen because it was found that certain similarity measures have a bias towards or away from positions of lattice alignment. This bias is referred to as *positional bias* to distinguish it from a positive or negative bias in the amplitude of the function value, so called amplitude bias. The two types of bias are related, but the former is easier to relate to convergence performance.

The positional-bias may be measured by performing a hierarchical brute-force search in the region of the ground truth, to obtain a *biased ground truth* position. An example of a function surface with a non-zero bias is given in 3.5a. A brute-force search is used because some function surfaces, like that of STD, have many discontinuities causing other optimisation methods to fail. The hierarchical search consisted of a 21×21 lattice of points in $\mathbf{v}$, with 0.1 pixel spacing. Twelve levels were used with the spacing decreasing by a factor of $\frac{2}{3}$ each level. The grid always centred on the previous position with the best (lowest) function value.

In Chapter 4, a comprehensive study is made of bias. For now, it suffices to say that the positional-bias should be measured before testing convergence, to ensure the effects of positional-bias do not influence convergence rate. Lattice-aligned

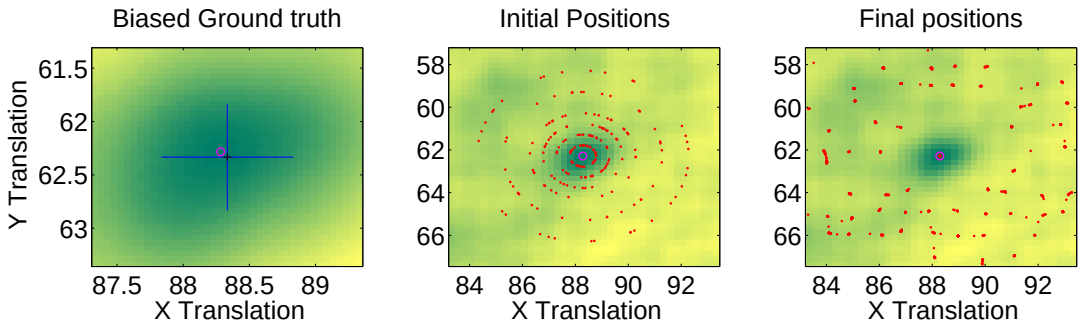
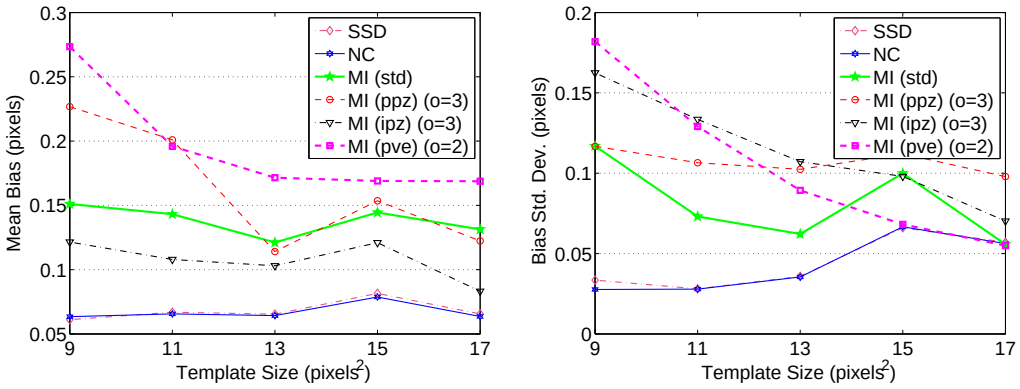


Figure 3.5: Illustration of a convergence test. (a) First the biased ground truth (red o) is located relative to the "true" ground truth (blue cross) (b) The registration is initialised from random starting positions (red dots) at pre-set distances from the biased ground-truth (c) The distances from converged positions (red dots) to the biased ground truth are measured.

solutions are generally rare, so experiments using lattice-aligned templates are relegated to Chapter 4 where they are more relevant.

Convergence tests were initiated from 600 positions at six offsets (0.5, 1.0, 1.5, 2.0, 3.0 and 4.0 pixels) at random angles, much like the example in Figure 3.5b. The same set of offsets was used for every similarity metric, to avoid favouring one metric over the others. After convergence (shown in Figure 3.5c) the Euclidean distances between each converged point and the biased ground truth was measured to obtain an error value. The use of multiple starting points and multiple images prevents the results from being affected by a particular dataset or by the starting position on the function surface.

The experiments were performed for five template sizes, ranging quadratically from 9^2 to 17^2 pixels, to examine the effects of sample size. For the MI metrics, a histogram bin-size of 16 intensities was found to work well. In all cases, the Levenberg-Marquardt algorithm was used for convergence, and translation warps were used. Comparable results were obtained with other methods, so only



(a) (b)

Figure 3.6: Bias for each MI family, SSD and NC, split into (a) mean and (b) standard deviation values.

translation results are shown to preserve space. In all 144,000 tests were made (5 template sizes * 6 measures * 100 points * 6 distances * 8 images).

3.5 Results

Using the test setup of Section 3.4 the bias and convergence were measured. The results for positional-bias are shown in Figure 3.6, with separate series for each similarity measure, plotted for increasing template size. As expected, bias generally decreases as the size of the template increases. This occurs because larger sample sizes supply more information to the similarity measure, diluting the effects of individual outliers that could pull the template off-target. The obvious exceptions are the SSD and NC similarity metrics, which are virtually unaffected by the number of samples. These metrics also exhibited the lowest bias. The reason for the good performance and lack of variation for template size is due to the fact that the templates extracted from the test-set have exactly the same lighting conditions and no additional noise. These are assumptions that

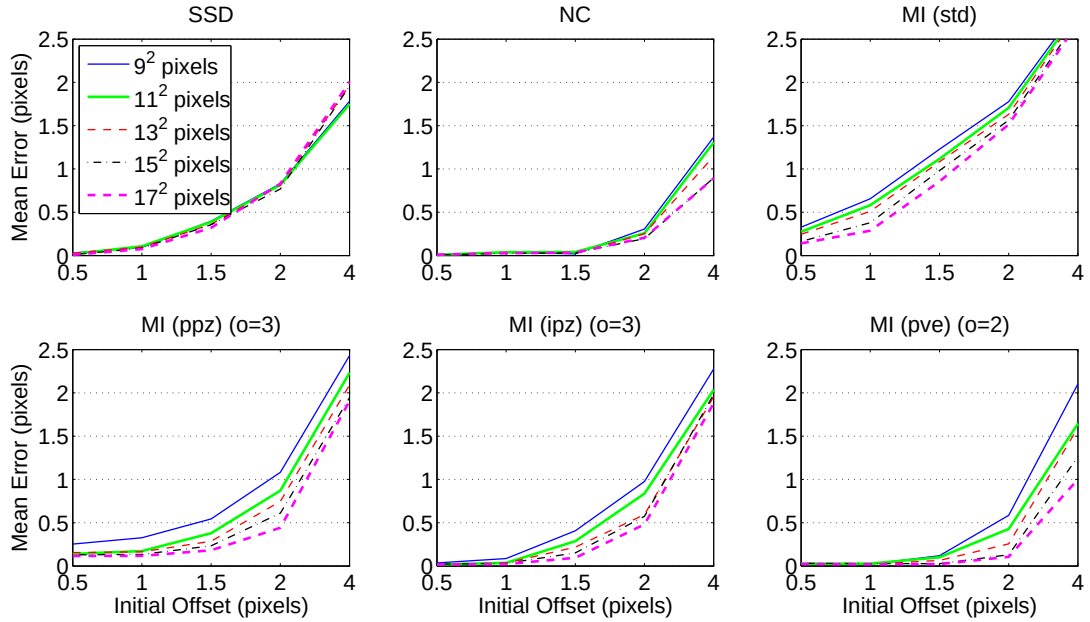


Figure 3.7: Mean of convergence error for each MI family, SSD and NC

SSD and NC explicitly make. The standard deviations of bias in Figure 3.6b, are generally around half the value of the mean, showing that the results are consistent.

Within the MI similarity functions, the ascending order of performance was: PVE, PPZ, STD and IPZ. PVE generally has a bias towards integral positions, which is what leads to its large bias value. This is discussed in detail in Section 4.2.3. For PPZ, the large bias at low template sizes occurs due to slight blurring of the histogram with the kernel dominating over the data when the sample size is small. For larger templates PPZ and STD exhibit very similar bias values. IPZ outperforms all the above MI methods, due to the uncertainty in the intensity being better modelled. IPZ blurs the joint-histogram less than PPZ while using more of the information implicit to the data than STD or PPZ.

The results of the convergence tests are shown in Figure 3.7. Each similarity

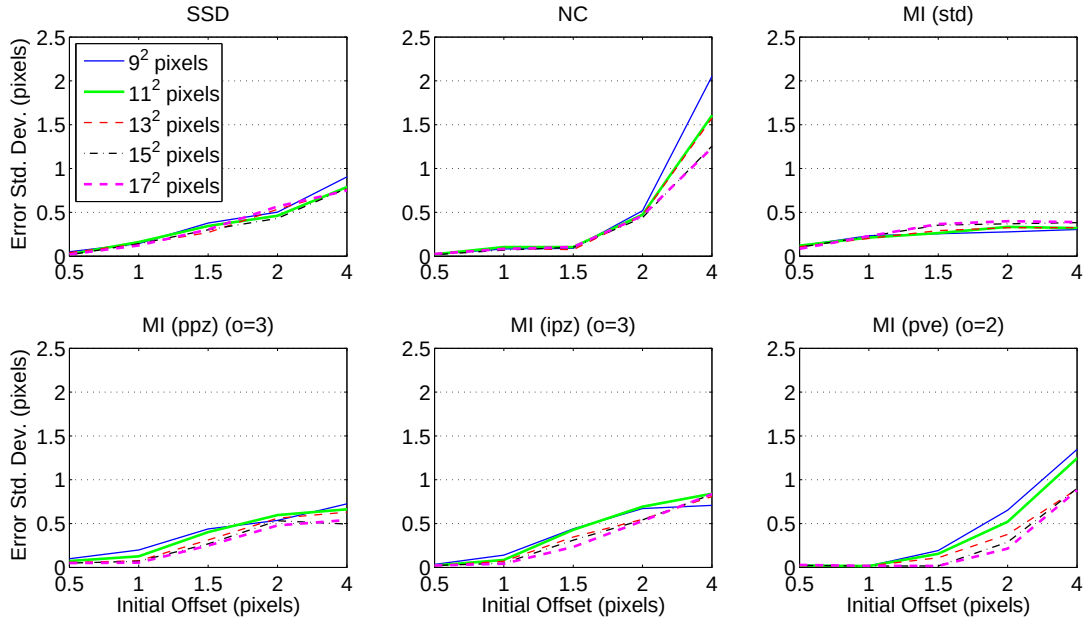


Figure 3.8: Standard deviation of convergence error for each MI family, SSD and NC

metric is shown separately, plotting mean error across all images and tests against the initial offset from the biased ground truth position. As expected the lower the initial offset the lower the final error. Also, in general the larger the template size, the lower the mean error. The exception is SSD, which showed similar performance at all template sizes, because there was insufficient noise to generate significant outliers.

Of all the methods NC performed the best, particularly at large offsets. This is due to the function's large basin size. Again SSD and NC perform well against MI due to the lack of occlusions, noise and variation in lighting between the template and the reference image.

Of the MI methods PVE outperformed SSD and performed better than NC up until 2 pixel offsets for a large templates, because it too has a large basin size. The

larger basin arises from the greater spatial support used in each measurement, which is equivalent to increasing the template size by 2 pixels in each dimension for "free".

PPZ outperformed STD, due to its better population of the joint histogram and the inclusion of uncertainty of intensity. Both of these methods have implicit discontinuities as shown in (3.5a) and (3.5c), which introduce numerous local minima. These local minima (hiss) are discussed in detail in Section 4.2.1. For now it suffices to mention that hiss has little effect while the steps of the convergence algorithm are large. However as convergence is approached, the steps become steadily smaller until the macroscopic variations in the function surface are overwhelmed by hiss. The net result is that convergence is never quite reached. Hence the low but non-zero error at low initial offsets, which is particularly apparent for small templates.

If the low non-zero error of PPZ is considered, it can be seen that the convergence of IPZ is similar to PPZ because the point of inflection in the two convergence performance plots is at the same initial offset (2 pixels). Better performance was expected due to the better use of a Parzen window, but the results seem to indicate that similar orders of performance induce similar convergence rates.

The standard deviations of convergence error are shown in Figure 3.8. These values were obtained by taking the standard deviation of error for all 100 random angles, followed by a mean across images. Most similarity functions showed consistent performance. The exceptions are NC and PVE, because these functions have wide basins of convergence, which not only surround the global minimum, but the local minima as well. Similar standard deviation values were obtained in the testing for Chapters 4 and 5. Standard deviation plots are not shown in Chapters 4 and 5, since this would double the number of figures without aiding

analysis.

3.6 Conclusion

This chapter has introduced a single framework for the four main families of MI, namely: Standard sampling, Partial Volume Estimation, In-Parzen Windowing and Post Parzen Windowing. The analytic Jacobians and Hessians of these methods were derived. A computational cost analysis was performed, which shows that STD MI is not much more expensive to compute than Sum of Squared Differences. The implementation was used to test the convergence of various image metrics using the Levenberg-Marquardt Method on a diverse array of images.

Despite their simplicity, SSD or NC are the methods of choice where the image is not occluded and the intensities of the template and reference image are linearly related. NC performs well even at large initial offsets due its large basin size. Where numerous outliers exist, PVE gives the best convergence performance, but at the cost of a large bias. So once the algorithm has converged it is recommended that IPZ is used to reduce the bias the result.

In Chapter 4 the discussion of the families of Mutual Information that have been introduced is continued. However the focus shifts to consider the artefacts in the function surface of Mutual Information and how these affect convergence.

Chapter 4

Reducing Mutual Information Artefacts with Super-Resolution

(and other methods)

At this point four families for mutual information have been introduced (in Chapter 3) and placed into the same mathematical framework for use in registration. The fact that the function surface of MI is not always smooth has been briefly mentioned and the potentially negative results of these surface irregularities have been demonstrated in the results in Section 3.4. These irregularities are hereafter referred to as *artefacts* and in this Chapter their effects are further considered.

Artefacts occur for all similarity functions, not just MI. At this point a distinction between two types of artefacts is drawn: *hiss* and periodic *glitches*. Hiss appears as random high frequency shifts in the cost function surface. These random shifts are generally small compared to the overall value at each position. Hiss is caused by non-linearities in the function d_{mi}, which cause discrete shifts in histogram bin populations as the warp parameters $\mathbf{v}$ vary. This behaviour is essentially random,

since it depends on numerous local shifts in value for each sample point.

Glitches are a periodic pattern in the cost function surface. They have a larger amplitude than hiss, although this is generally still smaller than the absolute function value. Glitches also have the more insidious effect of shifting global minima away from their "true" positions. This effect is called *positional-bias*. Unless otherwise specified "bias" should be taken to mean positional-bias. Glitches are generally caused by a combination of the synchronisation of the reference lattice and the warped lattice of the template combined with the effect of local correlations in the two sets of data.

In MI, the characteristics of these artefacts are affected by how the joint-histogram of f_R and f_T is sampled. In Figure 4.1, two typical examples of hiss and glitches in MI are shown. First order PVE is well known [66] to have a strong bias towards integral valued translations (lattice alignment), as demonstrated by the cusps at these points. STD has a bias away from lattice alignment, which is demonstrated by the local minima at these points. STD is also a discontinuous function, exhibiting the numerous local extrema typical of hiss.

Following some background in Section 4.1, the causes of artefacts are discussed in Section 4.2. Techniques to reduce the effects of artefacts are presented in Section 4.3. Next, these techniques are compared in terms of registration performance in Section 4.4 before concluding in 4.5.

4.1 Background

The MI value obtained for a given set of warp parameters, is calculated from an estimate of the PDF of intensity values in f_R and f_T. Hence the accuracy of the MI value is dependent on the accuracy of the histogram estimate. Typi-

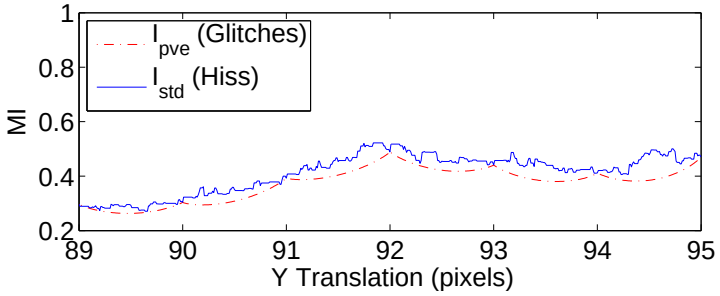


Figure 4.1: Examples of hiss and periodic glitches in function surfaces. The ground truth is a translation of 92 (pixels) in each case. First order PVE has a strong bias towards lattice alignment, exhibiting a regular pattern of cusps. STD is discontinuous, exhibiting the semi-random behaviour referred to as hiss, a noticeable bias *away* from positions of lattice alignment.

cally, the histogram is based on taking one sample per template pixel from each image, so called standard sampling. Three methods have been proposed, which seek to obtain a more accurate histogram, or at least smooth the cost function surface: In-Parzen Windowing [94, 63], Post-Parzen Windowing [81] and Partial Volume Estimation [47, 14]; but none of them eliminate artefacts entirely. Before discussing these methods further a digression is made into the inaccuracy of histogram sampling.

4.1.1 The Inaccuracy of Histogram Estimation

Consider the example of a 2 pixel square region in a larger image, where linear interpolation is used, like that in Figure 4.2a. If it were practical to extract and represent a continuous PDF, it would resemble the PDF shown in Figure 4.3a.

Generally it is not practical to represent a PDF in this way, so a histogram representation with intensity bins of finite size, Δi, is used. This leads to the

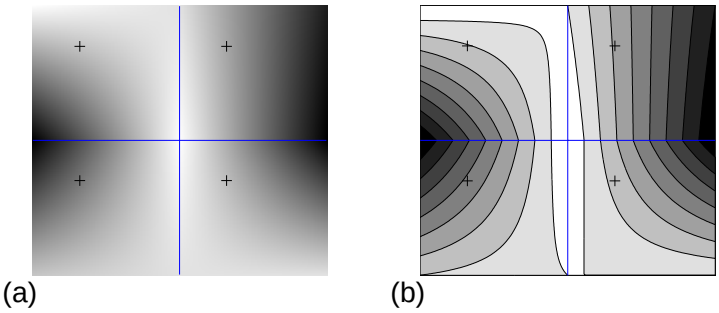


Figure 4.2: Examples of 2×2 pixel images that are (a) continuous in space and intensity (b) continuous in space with discrete intensities. The cross-hairs show four sample points at interpolated positions that are used to form the histogram in Figure 4.3c. Using a higher sampling frequency would provide a better approximation of the histogram (Figure 4.3b). Parzen windowing also gives an improved approximation (as shown in Figure 4.3d).

approximated PDF shown in Figure 4.3b. To illustrate this effect more clearly, each intensity in Figure 4.2b is coloured according to its membership among the bins of fixed size, much like those defined in Figure 4.3b.

Even with this approximation, obtaining an accurate histogram is too expensive for practical use, since it would involve computing the area between each iso-intensity contour in Figure 4.2b. So, a further approximation is made by populating the histogram using one sample per pixel. The four sample points in this example are indicated by the crosses in Figure 4.2a & b. This sampling regime leads to a histogram like that shown in Figure 4.3c. Clearly this is a poor representation of the "true" PDF of Figure 4.3a. However, usually the template is somewhat larger than 2×2, so a much larger number of samples is taken. This improves the statistical relevance of the histogram significantly, but sub-pixel information inherent in the ordering of the pixels is lost. Sampling in this manner also assumes independence and identical distribution (IID) of samples. The fact

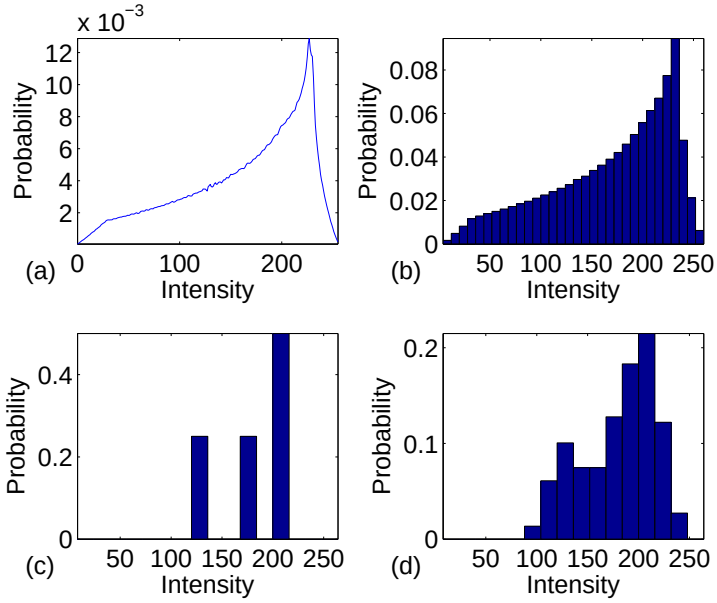


Figure 4.3: Examples of histograms that are (a) continuously sampled in space and in i (b) discretely defined in i but continuously sampled in space (c) discretely sampled in space and i (d) discretely sampled, as in (c), but then convolved with a Parzen Window. The discrete intensity sampling in (b) means that it is an approximation of (a). Normally spatial sampling is discrete as well, resulting in the much worse approximation shown in (c). Parzen windowing ameliorates this effect somewhat in (d).

that these assumptions are seldom correct is generally ignored. Kadir and Brady have proposed a method to more accurately estimate the statistics of images that does not assume IID [37]. This method is extended to obtain MI estimates from image pairs in Chapter 5.

Due to the finite size of the bins and the limited number of samples, MI is generally underestimated (or amplitude-biased). Ji *et al.* [34] and Moddemeijer [56] both discuss amplitude-bias and derive estimates for it. Amplitude-bias is mentioned to differentiate it from the *positional*-bias that is considered in this work. Positional-bias is a shift of the minimum away from its "true" position. In Figure 4.4a-d increasing sample rates are used to obtain progressively better MI estimates and hence lower bias values. These two forms of bias are simply a different way of looking at MI estimation error, albeit on different axes of the function surface plots. However for optimisation methods, it is generally more useful to know the error in the position of the minimum than the error in the function value at the minimum.

In addition to the discretisation of the intensity domain, the histogram probabilities are discretised as well, due to a finite number of samples. This is essentially a non-linearity within the MI function and is the primary cause of hiss.

4.1.2 Interpolation

For all but the most trivial of transformations, values at non-grid positions must be obtained using interpolation. Numerous interpolation techniques exist, of which a few are summarised: Nearest Neighbour, Bi-linear and Bi-cubic. Figure 4.5 illustrates an interpolated position $\mathbf{x}' = (x'_1, x'_2)$ between four pixels $\mathbf{x}_n = (x_1^{(n)}, x_2^{(n)}) \,|\, n \in [1;4]$ and is referred to in the following descriptions. In Nearest Neighbour Interpolation (NNI), the sample point takes its value from nearest

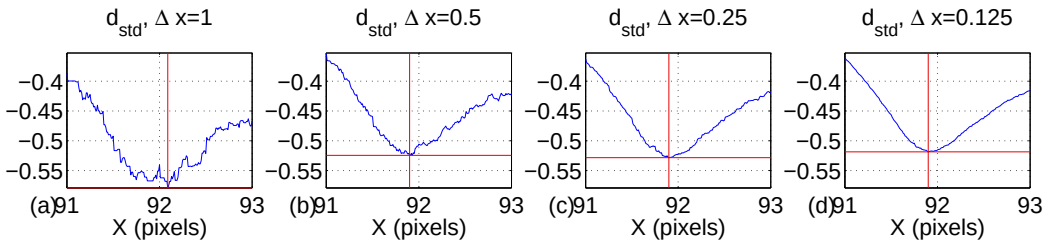


Figure 4.4: Amplitude bias (horizontal red line) and positional bias (vertical red line) at the minimum of a (negative) MI function surface. The amplitude and positional bias are respectively measures of the absolute errors in d and x from the minimum of the true MI of the images. The effect of increasing the sampling rate is shown from left to right: where in (a)-(d), 1, 4, 16 and 64 samples per pixel are taken. As the sampling rate increases, the true MI value is better approximated and both amplitude & positional bias decrease. It should be clear that these are merely two ways of referring to MI estimation error, albeit on different axes.

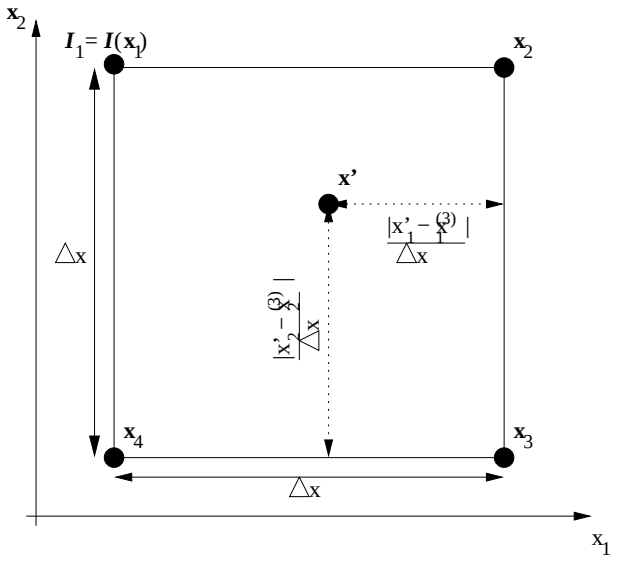


Figure 4.5: An interpolated position between four pixels positions, where the relative distance between point x' and $\mathbf{x}_3$ is shown.

lattice point:

$$f(\mathbf{x}') = f(\mathbf{x}_n) \,|\, n = \arg_n \min |\mathbf{x}_n - \mathbf{x}'|$$

where f is some arbitrary image, $\mathbf{x}'$ is the position of the *sample point* and $\mathbf{x}_n | n \in [1;4]$ are the positions of the *lattice points*. So in the example in Figure 4.5, $f(\mathbf{x}') = f(\mathbf{x}_2)$.

In Bi-Linear Interpolation (BLI), the value of the sample point is a weighted sum of the four closest lattice points. The weightings are inversely proportional to the distances in the x_1 and x_2 directions between the sample and lattice points:

$$\begin{aligned} f(\mathbf{x}) &= \sum_{n=1}^{4} w_n f(\mathbf{x}_n) \\ \text{where} \qquad w_n &= \frac{(\Delta x - |x_1' - x_1^{(n)}|)}{\Delta x} \frac{(\Delta x - |x_2' - x_2^{(n)}|)}{\Delta x} \end{aligned}$$

where $\Delta\mathbf{x}$ defines the spacing between the lattice points.

Bi-cubic interpolation simply extends the spatial support to a wider region and uses cubic functions to combine the values of the 16 closest lattice points. For interest one set of bi-cubic equations is given:

$$f(x_1, x_2) = \sum_{i=-1}^{2} \sum_{j=0}^{3} a_i a_j f_{ij} \tag{4.1}$$

$$a_i = b(|-\dot{x_1} - i|) \tag{4.2}$$

$$a_j = b(|-\dot{x_2} - i|) \tag{4.3}$$

$$b(\epsilon) = \frac{1}{6}(-4(\epsilon-1)^3 + 6(\epsilon)^3 - 4(\epsilon+1)^3 + (\epsilon+2)^3) \tag{4.4}$$

where $\dot{x}$ indicates the fractional part of a value x, and f_{ij} is the intensity of one of the 4×4 neighbourhood surrounding the current position $\mathbf{x}$, in left to right, top to bottom order. This gives a surface that is C_2 smooth, versus the C_0 surface for BLI and discontinuous surface for NNI. Care should be exercised when using Bi-cubic interpolation, since the values obtained with this method may go outside

of the intensity bounds defined by $(\min(f), \max(f))$. Either cropping of values or a histogram with additional bins may be required.

Partial Volume Interpolation or Estimation are often treated as another interpolation method [47, 66]. In this work it is treated as a different *sampling* method, and was discussed as such in Section 3.1.4.

4.2 Artefacts and their Causes

4.2.1 Hiss: Artefacts caused by Non-linearities

The root cause of hiss is non-linearities in the function d_{mi}, which induces discrete shifts in the intensities at sample positions as the warp parameters, $\mathbf{v}$, vary. This is clearly demonstrated by the $\delta()$ functions in the derivative of mutual information in (3.8a). Other similarity metrics also suffer from hiss. For example a common implementation of the Sum of Squared Differences function is:

$$d_{ssd}(\mathbf{v}) = \sum_{\mathbf{x}} [f_T(\mathbf{x}) - \text{floor}(f_R(\mathbf{x_w}))]^2$$

The floor function is explicitly shown indicate the effect of representing interpolated samples as 8-bit integers for gray-scale. The result of this is clearly seen when comparing the two function surfaces shown in Figure 4.6a, where discrete and continuous representation are shown.

To understand how hiss arises, again consider Figure 4.2. In particular refer to the upper left sample point in Figure 4.2b. The point is slightly left of the iso-intensity line (on which all values are 10). Its interpolated value is 10.05, but because of the integer representation it takes a value of 10 exactly. Now, consider a small change to warp parameters $\mathbf{v}$, such that the sample points shift slightly

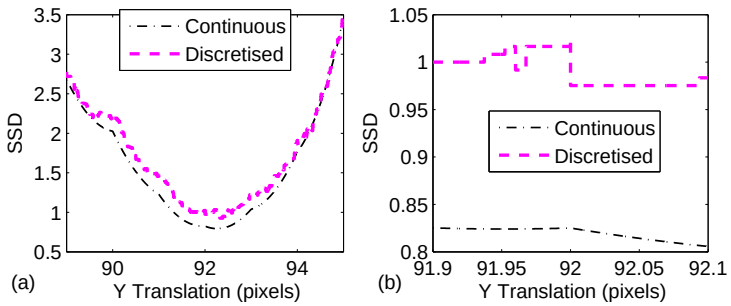


Figure 4.6: Examples of surfaces for SSD functions for discrete and continuous representations. (a) and (b) show the same function, but in (b) the surface is shown over a shorter range of values. The discrete representation results in regions of constant value separated by discontinuities. The frequency of discontinuities is limited by the gradient of the template and reference images. The continuous representation also has regular discontinuous changes in gradient because bi-linear interpolation was used.

to the right. The shift is just sufficient that only the upper left point crosses an iso-intensity line. It now takes an interpolated value of around 9.95, with a new discretised value of 9. This change occurs sharply as the iso-intensity line is crossed, resulting in a discontinuity in the cost function.

For MI with standard sampling, hiss occurs via the same mechanism. In this case, the crossing of the iso-intensity line induces a discrete shift between two bins in the joint histogram. If the trajectories of each sample point are traced for a particular direction in **v**-space, the number of discontinuous changes is finite: equalling the number of times an iso-intensity line is crossed. The limited number of discontinuities is clearly observed if the function surface is plotted over a shorter range of values in Figure 4.6b and resembles NNI. Note that hiss, as defined in this work, is *not* the same as white noise commonly observed in analogue signals, since its frequency is bounded.

Hiss may only be removed if the non-linear parts of the equation are removed: *e.g.* by removing the floor routine implicit to SSD as shown in Figure 4.6. For MI, standard sampling and Post-Parzen windowing generate hiss due to their inherent discretisation of sample values, whereas PVE and In-Parzen Windowing do not. This is clearly seen in Figure 4.7 where the traces show the cumulative sum of histogram bins for each MI family as one component of $\mathbf{v}$ is varied. The traces for standard sampling and Post-Parzen windowing both exhibit discrete shifts as the samples discretely shift between histogram bins. The traces for In-Parzen windowing and PVE vary continuously. This behaviour is predicted theoretically by the Jacobian of the four MI methods in (3.8a)-(3.8d), where standard sampling and Post-Parzen windowing both exhibit a zero plane interspersed with Dirac Delta functions.

4.2.2 Quantifying Hiss

The quantification of hiss is now considered. Although the amplitude changes induced by hiss may be quantified, by evaluating d_{mi} at a fine resolution, this would be computationally expensive and not very useful. However, estimating the *number* of bin shifts that could occur between two $\mathbf{v}$ positions can provide a likelihood of the optimisation algorithm becoming stuck in a spurious local minimum.

Consider the change in image intensity with respect to a change in gradient for a single sample at point $\mathbf{x}_n$, with current warp parameters $\mathbf{v}$:

$$\frac{\partial f(\mathbf{x}_{n\mathbf{w}})}{\partial \mathbf{v}} = \nabla f(\mathbf{x}_{n\mathbf{w}}) \frac{\partial \mathbf{w}}{\partial \mathbf{v}}(\mathbf{x}_n, \mathbf{v})$$

where $\mathbf{x}_{n\mathbf{w}} = \mathbf{w}(\mathbf{x}_n, \mathbf{v})$. Since the number of discontinuities is sought, only those changes sufficient to induce a bin shift matter. If local linearity is assumed and

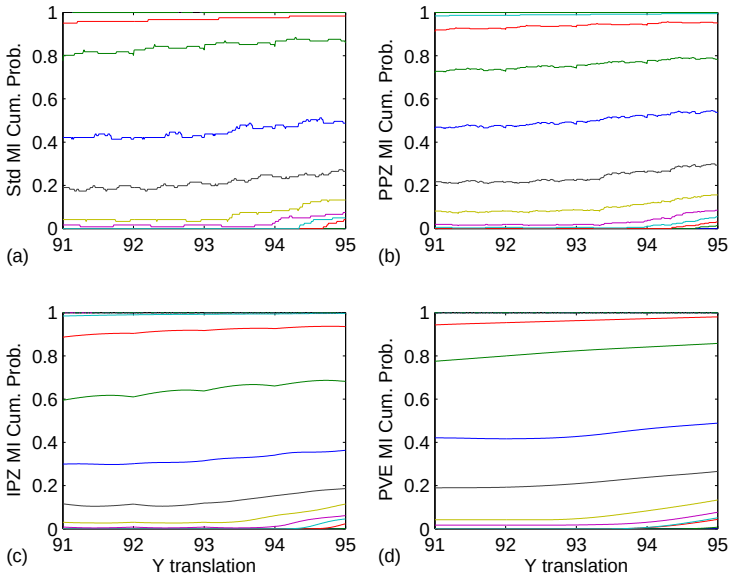


Figure 4.7: Traces of Histograms sampled using (a) Standard Sampling, (b) Post Parzen Windowing, (c) In-Parzen Windowing, and (d) PVE for varying translation. Discrete random shifts in the histogram bins for standard sampling are clearly visible. Post-Parzen windowing gives smoother traces, but the shifts are still discrete. PVE and In-Parzen windowing eliminate the non-linear floor function and hence exhibit smooth traces. For clarity only h_r is shown, rather than h_{rt}.

the sample is assumed to be somewhere between two iso-intensities with uniform probability, then the probability of a bin shift, p_s, is:

$$p_s(\Delta\mathbf{v}) = |\nabla f(\mathbf{x}_{n\mathbf{w}})\frac{\partial \mathbf{w}}{\partial \mathbf{v}}(\mathbf{x}_n, \mathbf{v})| \cdot |\Delta\mathbf{v}|$$

where $\Delta\mathbf{v}$ is a change in warp parameters and $|\cdot|$ indicates the absolute value applied to individual components of a vector. The effects of multiple samples are combined to obtain the total expected number of bin shifts N_s. This gives an equation of the form $N_s \propto \Delta\mathbf{v}$:

$$N_s(\Delta\mathbf{v}) = |\Delta\mathbf{v}| \sum_{n=0}^{N_\mathbf{x}} |\nabla f(\mathbf{x}_{n\mathbf{w}})\frac{\partial \mathbf{w}}{\partial \mathbf{v}}(\mathbf{x}_n, \mathbf{v})| \tag{4.5}$$

Separation of an MI value into a hiss component and an underlying, "true" component is not possible. So it is impossible to tell if an optimisation algorithm has converged to a spurious minimum or a "true" local minimum. This is not serious, since the effects of hiss are generally overwhelmed by the trends in the function surface at the macroscopic level. As the optimisation algorithm approaches convergence the steps taken will shorten. Eventually the steps taken reach a level where the effects of hiss exceed the macroscopic trends in the surface. This is the uncertainty in the distance to the "true" *local* minimum.

4.2.3 Glitches: Artefacts caused by Harmonics

Glitches are a manifestation of harmonics between the lattice points of the reference image and the sample points corresponding to the warped template lattice. Harmonics are also a problem in many physical systems, which are engineered to prevent or eliminate them: *e.g.* the NTSC standard of displaying at 29.997 frames per second to avoid harmonics with 60Hz (US) power supply, and low pass filtering the input to Analogue to Digital converters to prevent aliasing.

Commonly seen results of harmonics (glitches) include the visible scan-line seen when filming a computer screen and Moiré fringes in textured images of insufficient resolution. In all cases, glitches occur because the phases of two frequencies interfere constructively in some regions and destructively in others. In other words this effect is simply a manifestation of beats, with the interfering frequencies arising from the template and reference pixel lattices. Since MI (or any other similarity function for that matter) is based upon intensities within a finite region, a net amount of constructive or destructive interference can occur. This net interference in turn induces a positive or negative bias into MI. As warp parameters vary the net interference varies and hence the bias.

Since the period of the variation from constructive to destructive interference is dependent on the lattice periods, the glitch period, T_c, can be obtained directly:

$$T_c = \min(T_r, T_t)\text{frac}\left(\frac{\max(T_r, T_t)}{\min(T_r, T_t)}\right) \tag{4.6}$$

where frac(.) indicates the fractional component of a real number. Figure 4.8 serves to illustrate how glitches can occur. Consider two sinusoidal signals of differing frequency (an analogue for lattice sample rate). Now consider a small moving window, over which the SSD is taken (an analogue for a similarity metric). The net result is a varying SSD signal depending on the position of the moving window, with a period of variation following (4.6).

Pluim *et al.* made some effort to describe the structure of glitches for two sampling methods (standard sampling and first order PVE) in [66]. For completeness a similar description is included here.

For standard sampling, interpolation methods such as BLI and BCI cause sampled intensities to be averaged when the sample points are not aligned with the reference image lattice. In the histogram, the net effect is a movement from bins at the extremities of the histogram towards regions of high probability density.

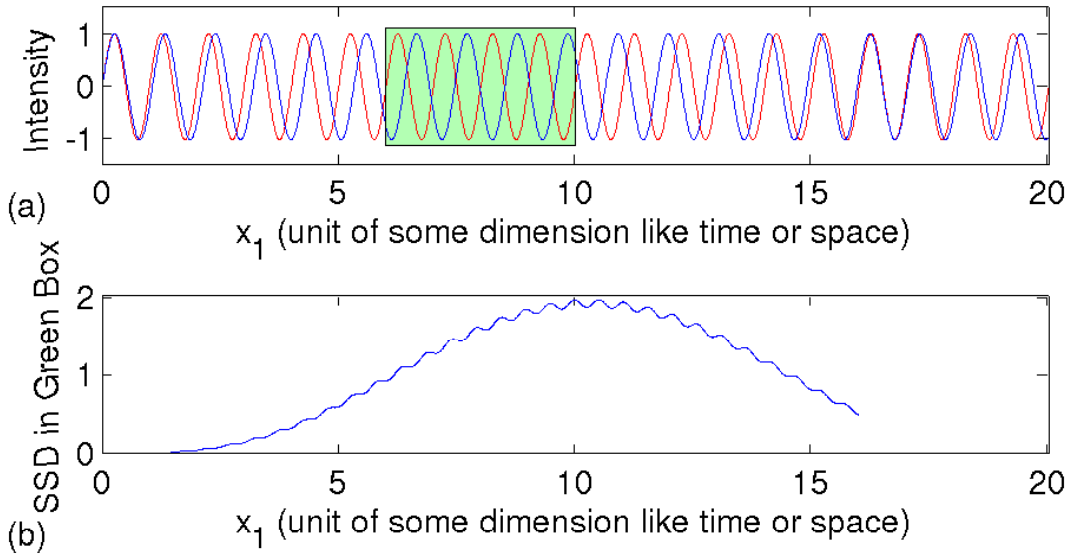


Figure 4.8: Demonstration of how beats in a pair of signals generate glitches. In **(a)** two sinusoidal signals (red and green plots) of differing frequency are shown. In some areas interference is primarily constructive and in others it is primarily destructive. The green box shows a finite region over which the SSD is measured. In **(b)** shows how SSD varies as the region of measurement is shifted across the two signals. The high frequency variation in SSD is referred to as glitches.

The result is the marginalisation of the outlying bins. Usually this means that bins with large populations increase at the cost of bins with small populations. So in the joint histogram, which dominates the MI function, there is a net decrease in entropy, resulting in an overall increase in MI. An illustrative example is shown in Figure 4.9a-c, with the resulting bias *away* from lattice alignment shown in Figure 4.1.

In unusual cases, *e.g.* adjacent mono-intensity regions with a sharp border as in Figure 4.9d-f, a shift from well populated bins to un-populated bins may occur, causing a decrease in MI. MI is similarly affected if Parzen windowing is used, but these effects are partly masked out by the blurring effect of the convolution operation.

For first order PVE, the opposite happens. For a sample point aligned with a lattice point, one histogram is populated per sample. As the sample shifts away from the lattice point, three additional bins are incremented by some fractional value dependent on d. The extreme case is anti-alignment (the sample point is equidistant from the surrounding lattice points). Each of the four bins corresponding to the neighbouring lattice point are incremented by $\frac{1}{4}$.

The net result is a dispersion of the joint histogram, and an increase in entropy for the joint histogram. Since the joint histogram dominates, MI decreases alignment between the reference and warped template lattice decreases. This net bias *towards* lattice alignment is shown in Figure 4.1. As the order of PVE is increased, these effects are greatly reduced, as found by Chen and Varshney[14], since the ratio of bins populated per sample between alignment and anti-alignment decreases from $\frac{1}{4}$ for 1^{st} order, to $\frac{4}{9}$ for 2^{nd} order, to $\frac{9}{16}$ for 3^{rd}, etc.

The glitches caused by PVE are particularly noticeable when the number of samples per histogram bin is low, since an underpopulated histogram offers many

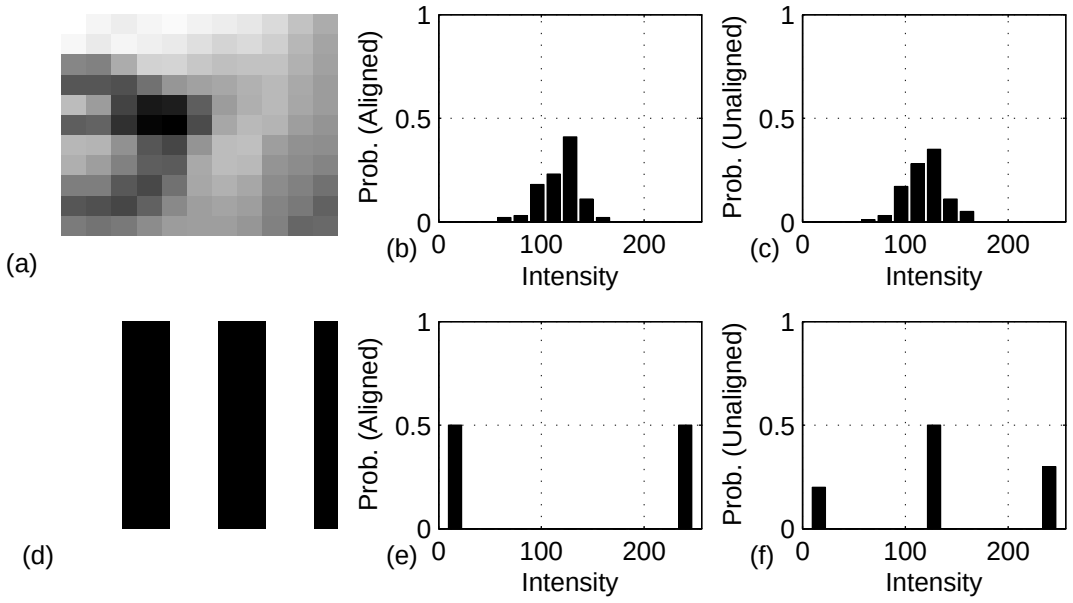


Figure 4.9: The effect of bilinear interpolation is to concentrate probabilities in central bins as shown for two example images (a) and (d). In the middle column: (b) and (e), a histogram extracted from points aligned with the lattice is shown. In the rightmost column: (c) and (f), a histogram is obtained from points sampled exactly out of phase with the image lattice using linear interpolation. The shift from outlying bins to inner bins is clearly visible. The bottom row (d)-(f) shows an unusual case where the entropy will increase as a result. For clarity 1D histograms (rather than 2D joint-histograms) are shown.

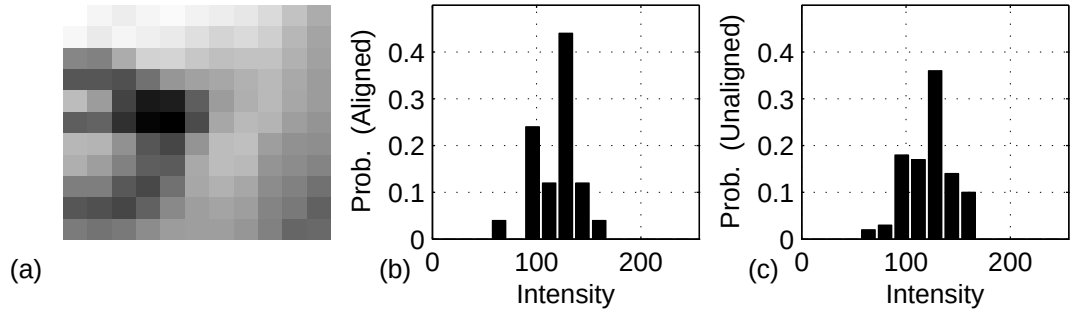


Figure 4.10: Partial volume estimation results in more histogram bins being populated. A histogram of the image in (a) is shown when sample points are aligned with the lattice in (b) and when out of phase with the lattice in (c). For clarity 1D histograms are shown.

opportunities for dispersion. One such example is shown in Figure 4.10.

4.2.4 Quantifying Glitches

As mentioned, apart from generating spurious minima in some cases, glitches have the insidious result of shifting the maximum away from its original position in **v**-space: positional-bias. As for hiss, disambiguating the spurious minima of glitches from the "true" local minimum is not possible, but if their frequency is high the macroscopic trends of the cost function surface due to the image data will dominate. If not, optimisation will fail anyway.

Because the precise effects of glitches are data dependent the author takes the conservative view that the uncertainty in the position of the "true" local minimum is an entire glitch period. Assuming optimisation has converged to the correct minimum, and since one spurious minimum, at most, is generated per glitch, the uncertainty can be no worse than:

$$E_{\mathbf{x}\,\max} = \frac{1}{2}\min(T_r, T_t)\mathrm{frac}\left(\frac{\max(T_r, T_t)}{\min(T_r, T_t)}\right) \tag{4.7}$$

where $T_r = \Delta\mathbf{x}_r$ and $T_t = \mathbf{w}(\Delta\mathbf{x}_t, \mathbf{v})$, assuming that warp parameters $\mathbf{v}$ are such that T_t has a sufficiently low fundamental frequency.

Strong artefacts only occur near "pathological" warps, where the rotated lattice of the template is at an angle and scale such that the period in the x_1 and x_2 is a multiple of the reference lattice-period *i.e.* when a Pythagorean triangle is formed: *e.g.* Rotation angle, θ, is 0, or $\theta = 30°$ and scale of $\frac{2}{\sqrt{3}}$. Away from these positions in $\mathbf{v}$, the warped lattice points of the template have multiple harmonics from the point of view of the reference image axes, resulting in exceedingly weak glitches. Care should be exercised to ensure that solutions near to positions of lattice alignment are not overwhelmed by glitches. This can easily occur in applications like tracking, where translation warps are used.

4.3 Reducing the Effects of Artefacts

Techniques to reduce the amount of hiss, glitches and bias are now considered. The first and most obvious way to eliminate hiss has already been mentioned: reformulate the similarity metric to eliminate discontinuous functions such as flooring, or binning. This will completely remove hiss, but can come at a computational price. In-Parzen windowing [81] and PVE [47, 14] both do this by using a continuous function to populate histogram bins as shown in Figure 4.6 and 4.7.

A second common technique is to break synchronisations between the template lattice and the reference lattice as suggested by Pluim *et al.* [66]. In some applications where only translation warps are used, a slight rotation or rescaling will achieve this. The rotation or scaling should be large enough that several complete glitch periods occur over the region being sampled, to prevent regions with strong harmonics generating glitches. This is the reason that Tsao [88] obtained

a result where rescaling the template by $\frac{\pi}{3}$ gave smaller artifacts than rescaling it by $\frac{128}{129}$. A larger rescaling results in larger glitch periods and uncertainty, but is more likely to break up harmonics in the reference image. For more complex warp functions with rotations, scales and shears, this is less of an issue. The proviso is that solutions are not near pathological warps and the optimisation algorithm should not drift too close to such warps.

Expanding spatial support is also an effective way of reducing glitches. This is equivalent to applying a low pass filter to the data. Since the passband of the filter is at a lower frequency (less than 0.5 pixel^{-1}) than the glitch frequency (greater than 1 pixel^{-1} by definition) the effect of glitches is greatly reduced. The main disadvantage of expanded spatial support is that some loss in the higher frequency information occurs and it is more expensive computationally. An example of increased spatial support is increasing the order of PVE [14].

Expanding intensity support (populating multiple intensity bins per sample) reduces the effect of glitches, since it accounts for the inherent uncertainty in the values of the samples obtained from each image. This has the effect of blurring the histogram. This effect is stronger than the blurring effect due to interpolation, effectively masking it. Examples of increasing intensity support are: increasing the size of a Parzen window (or using a Parzen window at all), and increasing the histogram bin-size as Thevenaz and Unser did using grey-cones [81]. There are no inherent disadvantages to this method apart from increased computation. However large image gradients (in a strong texture for example) can still cause artefacts to be generated.

Noisy data can cause relatively large shifts in intensity for a small shift in $\Delta\mathbf{v}$. This can result in strong glitches from the correlation of local regions where synchronisations occur. Ji *et al.*[34] suggest that smoothing the data initially can

reduce hiss substantially, since there are less outlying bins that are populated. Hence glitches are reduced, since synchronisations must be over a larger region. However, this causes information loss in the original data.

Lastly, the rate at which the images are sampled can be increased. (4.5) and (4.7) show that this reduces the effects of hiss and the number of glitches relative to the sample rate. For hiss, an increased sample rate gives a better populated histogram and more random shifts (since each sample follows a path as $\mathbf{v}$ that crosses iso-intensities) but the size of the shifts is smaller. Moddemeijer showed the variance in MI to be inversely proportional to the number of samples [56]. For glitches, increasing the sample rate results in a lower T_t and a lower uncertainty in the position of the minimum, $E_{\mathbf{x}\,\min}$. The main disadvantage of this method is the increased computational expense. Even so, as shown in (4.5) and (4.7), choosing the sample rate allows fine control of glitch period and hiss frequency, allowing these to be altered on-the-fly during optimisation. Also, hiss only needs to be controlled when close to convergence, allowing computation and accuracy to be explicitly traded-off. No analogy for increased sampling resolution exists for PVE, because PVE is equivalent to the integral of a weighted window over the local region while using Nearest Neighbour Interpolation.

Summary As discussed, many strategies have been proposed to deal with artefacts in the MI function surface. In this section two types of artefacts were considered separately: hiss, which is caused by discontinuities in the function; and glitches, which are caused by harmonics between the lattice spacing of the reference image and the sample point spacing. Generally the above proposals have only been considered so far as the implications for the particular application or choice of sampling method. Certain techniques can then be chosen in a principled manner to give the highest convergence rate and lowest bias at the lowest

computational cost. Six strategies to reduce artefacts have been discussed:

1. Eliminate function discontinuities: hiss has little effect since optimisation methods sample the function surface relatively sparsely.
2. Expand spatial support: This reduces the amplitude of glitches at some computational cost (see Table 3.3).
3. Break synchronisation between lattices: this is only possible in some applications where solutions are far from pathological warps.
4. Expand intensity support: This also reduces the amplitude of glitches at a computational cost.
5. Smooth the data: this results in a sharper peak and smoother surface at the cost of partially compromising the data.
6. Increase the sample rate: At some computational cost, this reduces both the amplitude and increases frequency of glitches, in a controllable manner, making them less visible to an optimisation method that sparsely samples the function surface.

In the next section the above methods are critically compared in terms of convergence rate and bias, allowing a principled choice to be taken.

4.4 Experiments

4.4.1 Experimental Setup

The test setup described in Section 3.4 was used to measure the performance of the various MI estimation methods in terms of:

- Bias: Euclidean distance between the ground truth and global minimum.

- Convergence: The mean distance from the (biased) global minimum.

As in Section 3.4, 600 offsets at random angles from the biased ground truth were used to initialise a Levenberg-Marquardt minimisation algorithm. Six fixed distances were used for the initial offset: 0.5, 1, 1.5, 2, 3 and 4 pixels. The same set of offsets was used for each function to prevent bias due to an unlucky choice of initial position. The offsets were from the biased global minimum, rather than the ground truth, so as to decouple the effects of bias and convergence to spurious minima. The biased global minimum was obtained using a hierarchical brute-force search in the region of the ground truth.

Six sets of experiments were performed, considering each of the strategies discussed in Section 4.2 for reducing the effects of artefacts. Experiments were performed for five template sizes, ranging quadratically from 9^2 to 17^2 pixels, and a fixed histogram bin-size of 32 intensities. In addition, tests for five histogram bin-sizes, ranging exponentially from 4^2 to 64^2 intensities were made, with a fixed template size of 17^2 pixels.

Varying the intensity bin-size is equivalent to varying the amount of spatial support. However, bin-size influences the results to such an extent that it is included with every set of experiments. In all cases BLI was used when interpolating. In all, approximately one million individual tests were performed: 8 images, 5 template sizes + 5 intensity bin-sizes, 6 offset sizes, 21 functions and modifications (like smoothing or lattice alignment), and 100 angles for each offset distance.

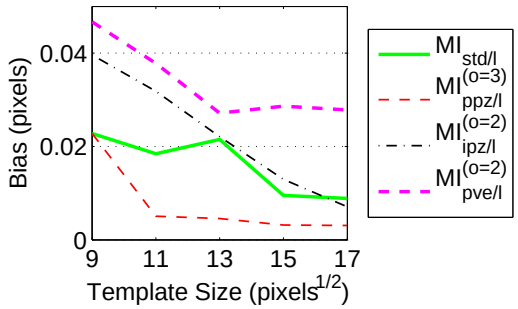


Figure 4.11: Bias when removing function discontinuities for lattice-aligned templates of varying size.

4.4.2 Eliminating Discontinuities

To examine the effect of the removal of discontinuities, four MI functions are compared: STD, 3rd order PPZ, 2nd order IPZ, and 2nd order PVE. For this set of experiments both lattice-aligned templates (*i.e.* where the template lattice exactly aligns with the reference lattice) and lattice-offset templates were used.

The bias obtained for each function is shown in Figure 4.11a. The bias ought to be zero for PPZ and STD, because MI is greatest when the reference and template images exactly match. However in practice, the minimum MI exists for a region rather than a point, due to the functions' discontinuous nature. As a hierarchical search is used to find the biased minimum, and the search algorithm returns the first point for which a minimal MI is found, an erroneous non-zero bias is obtained. The "bias" in this case approximately expresses the size of the region of the minimum. IPZ and PVE populate multiple bins in the histogram, and it is not always possible to obtain an exact match between the reference and template, so non-zero biases are also obtained in these cases. Even so, the bias values are all below 0.05 pixels.

The mean convergence error for each function is plotted for increasing initial

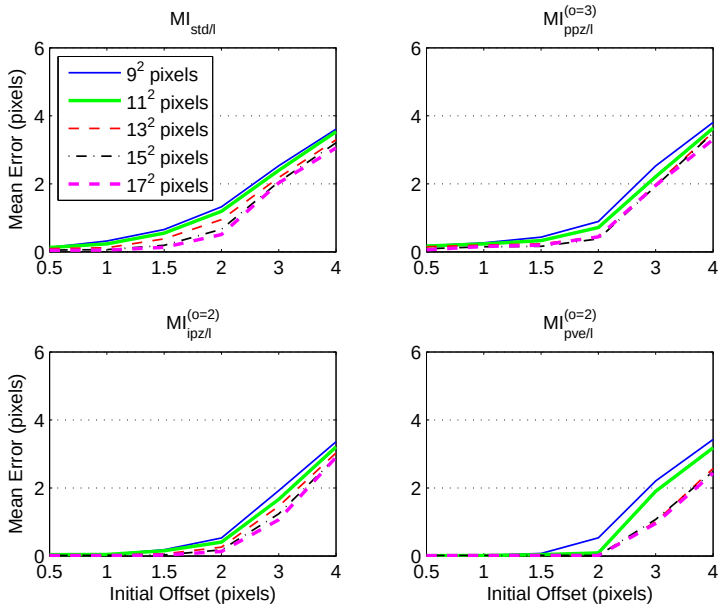
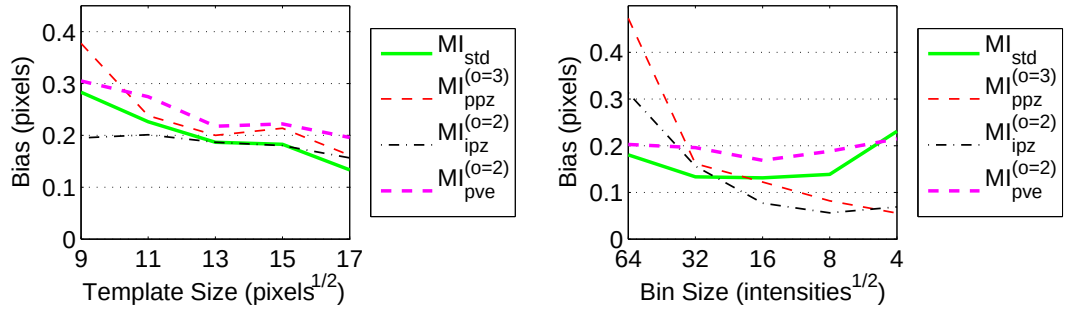


Figure 4.12: Convergence when removing function discontinuities for lattice-aligned templates of varying size.

offsets in Figure 4.12. As expected, the convergence results degrade as the initial offset from the minimum increases. Convergence also improves as the template size increases, because more samples (information) are available, resulting in an increased radius for the basin of convergence.

Lattice-aligned solutions are rare in practice, so the results in Figure 4.11 and 4.12 are given for interest only. More realistic test results are given in Figure 4.13a and 4.14, where the bias and convergence are respectively shown for lattice-offset templates .

The biases for lattice-offset templates, in Figure 4.13a, are larger than those for lattice-aligned templates, in Figure 4.11. This occurs, because the reference image no longer contains a region that exactly matches the template, and the information lost during the blurring and down-sampling, can never be recovered completely. For all the MI functions, bias decreases as the template size in-



(a) (b)

Figure 4.13: Bias when removing function discontinuities for lattice-offset templates when varying *(a)* template size and *(b)* histogram bin-size.

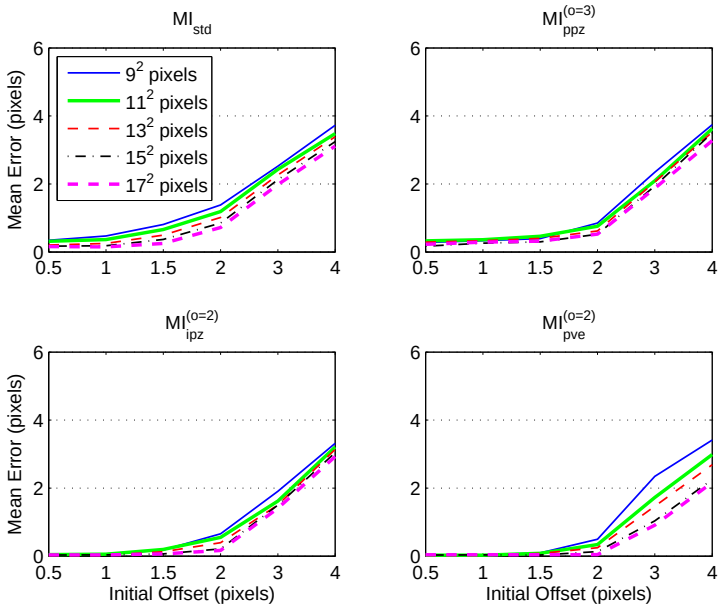


Figure 4.14: Convergence when removing function discontinuities for lattice-offset templates of varying size. MI_{std} and MI_{ppz} are discontinuous, while MI_{ipz} and MI_{pve} are smooth.

creases, because of the additional samples (information) contained within larger templates. PVE has the largest bias overall, due to its favouring of lattice-aligned positions. PPZ performs slightly better. However, due to the large histogram bin-size, PPZ over-blurs the histogram, so STD demonstrates a still lower bias. IPZ yields the lowest bias of all, due to its use of the fractional information contained within intensity values.

As shown in Figure 4.14, all the functions perform similarly well, for all template sizes, when initial positions are close to the minimum, and similarly badly when initial positions are distant from the minimum. This is expected because initial positions close to the minimum are usually within the basin of convergence and are likely to converge. Distant initial positions usually fall outside the basin of convergence and converge to spurious local minima. At intermediate positions, a range of performances are observed, because initial positions are generally on the edge of the basin of convergence. At these positions, slight differences in the algorithm or template size causes large variation in performance. In addition, convergence is generally good, when the initial offset is less than the radius of the basin of convergence. Initial offsets beyond this boundary show an inflection in performance curves, and the error approximately equals the initial offset.

Hence there are two main differentiators of performance:

- The range of performance at intermediate initial offsets.
- The initial offset where an inflection of performance is observed.

Neither STD nor PPZ achieve a mean error of zero when the initial offset is low, despite the initial position being within the basin of convergence. This occurs, because at positions close to convergence, the steps taken by the optimisation

are sufficiently small that hiss overwhelms the macroscopic trends of the function surface. An earlier example shown in Figure 4.4, clearly demonstrates this effect. IPZ and PVE, which are smooth functions, manage to achieve zero mean errors when initial offsets are low. So the main advantage of a smooth function, is that zero convergence errors are possible with Newton type optimisations. Discontinuous functions would need a search algorithm to achieve similar results or a method to reduce hiss. However, at points distant from the minimum, the function surface is sampled sparsely, so hiss has little effect on convergence rate.

STD demonstrated the worst convergence performance when varying template size, because of the simplistic way in which it populates the histogram. PPZ yielded significant improvements, because it accounts for uncertainty in intensity values and populates the histogram less sparsely. IPZ exhibited further improved convergence performance at low initial offsets, but had the same point of inflection in convergence performance (2 pixels) as PPZ. This indicates that the improved performance of IPZ relative to PPZ, is solely due to its lack of hiss. At large initial offsets, PVE has the lowest convergence error out of all the methods in Figure 4.14, because it has larger spatial support than the other methods. This increases the size of the region being considered in the reference image, and has the effect of increasing the radius of the basin of convergence. Hence, results at distant offsets are substantially better for PVE than those for any of the other methods. At larger offsets, PVE exhibits the greatest range of performance, indicating sensitivity to template size.

Figure 4.13b and 4.15 respectively show the results for bias and convergence while varying bin-size for lattice-offset templates. The bias is generally the lowest (best) at intermediate bin-sizes, with large biases when the bin-size is too large or too small. For large bin-sizes, the resolution of the intensity histogram becomes too low to describe the joint statistics, and hence loses the ability to discriminate

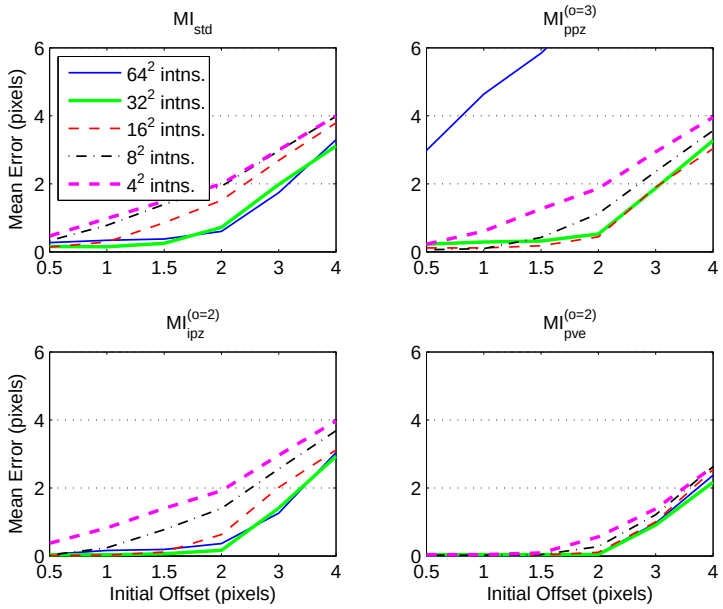
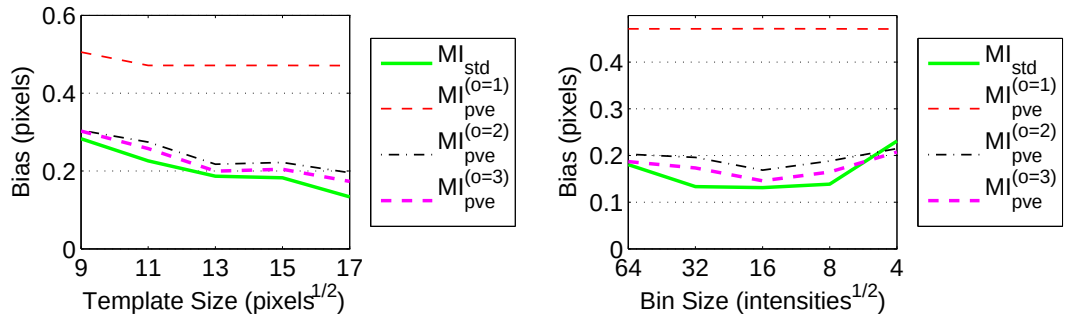


Figure 4.15: Convergence when removing function discontinuities for lattice-offset templates and varying histogram bin-size.

between well matched and poorly matched data. This is particularly apparent for PPZ and IPZ, which blur the histogram with a Parzen window. In these cases the histogram describes the Parzen window statistics rather than those of the data.

For low bin-sizes, there are too few samples to adequately characterise the joint statistics. In this case, all positions in the function surface appear to have equally high MI and again the ability to discriminate is lost. Hence STD, has an optimal bin-size of between 8 and 32 intensities. Methods that populate multiple bins, such as PVE, IPZ and PPZ, are by their nature less affected by lack of samples, so in these cases optimal bias is obtained for lower histogram bin-sizes of approximately 16 intensities.

In the convergence tests, shown in Figure 4.15, PVE again performs the best due its larger basin of convergence. IPZ outperforms STD and PPZ at equivalent bin-



(a) (b)

Figure 4.16: Bias when increasing spatial support for lattice-offset templates when varying *(a)* template size and *(b)* histogram bin-size.

sizes due to its continuous nature and the fact that it uses all available intensity data, including fractional data. When using optimal bin-sizes, PPZ outperforms STD, because PPZ models the uncertainty in intensity better. However, PPZ catastrophically fails once the histogram starts to become over-blurred.

To summarise, the advantage of smooth functions is that they allow convergence errors of zero when using Newton optimisation algorithms and the initial position is within the basin of convergence, without the use of techniques to reduce hiss.

4.4.3 Increasing Spatial Support

In this set of tests, the effects of increasing spatial support were considered, *i.e.* widening the size of the region used for PVE. The bias is respectively plotted in Figure 4.16a and b for varying template size and intensity bin-size, for four MI functions: STD and PVE of first, second and third order.

First order PVE gave a large constant bias of 0.467 pixels, because of its strong bias towards positions of lattice alignment. Recalling from Section 3.4 that the lattice-offset template was offset by $\frac{1}{3}$ of a pixel in each direction, a bias of

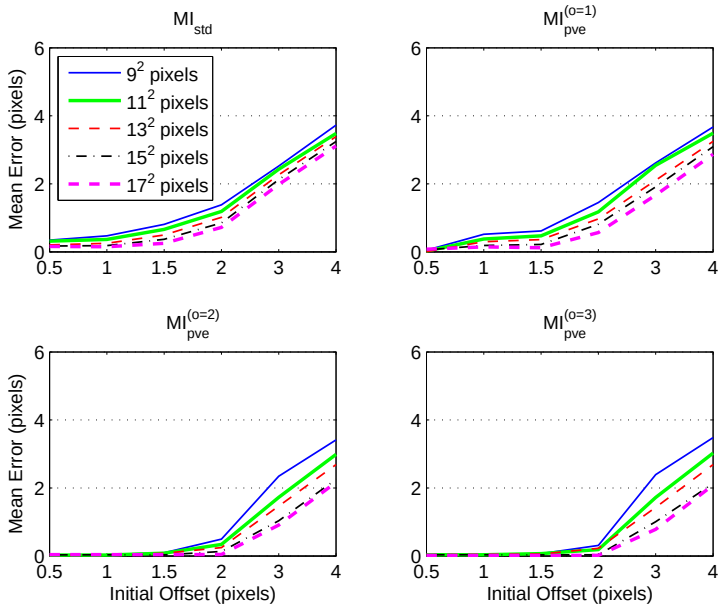


Figure 4.17: Convergence when increasing spatial support for lattice-offset templates of varying size.

$\sqrt{2.\frac{1}{3}^2} = 0.467$ pixels was expected, and corresponds to the results.

Increasing the spatial support to second order reduced the bias to approximately 0.2 pixels. A further increase to third order gave additional, but less significant, reductions in bias. This improvement in bias occurred because the difference between histogram dispersion at lattice-aligned and lattice-offset positions decreased, as discussed in Section 4.3. The same pattern of decreasing bias was observed for larger templates at optimal bin-sizes. STD outperformed PVE in every case, since it has no bias towards lattice alignment.

Figure 4.17 and 4.18 plot the convergence results when varying template size and bin-size. First order PVE outperforms STD at positions close to the minimum, when the initial offset increases to 1 pixel. However, convergence performance quickly degrades because other local minima, caused by glitches, capture the

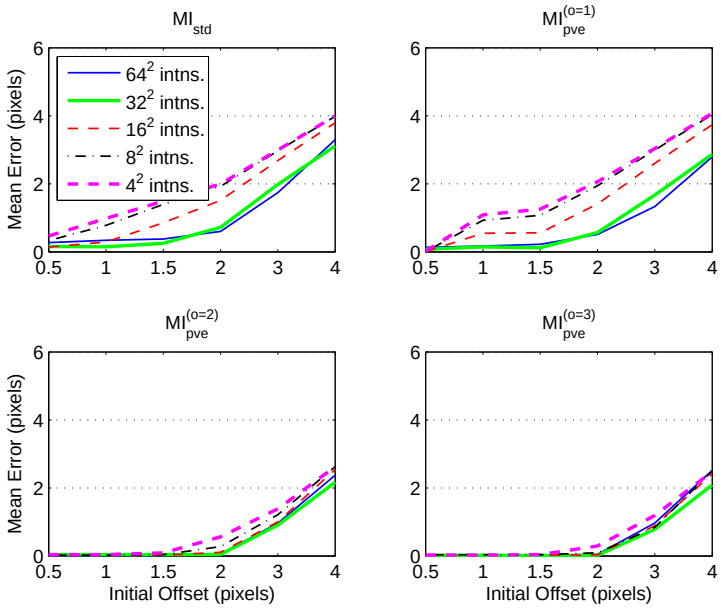
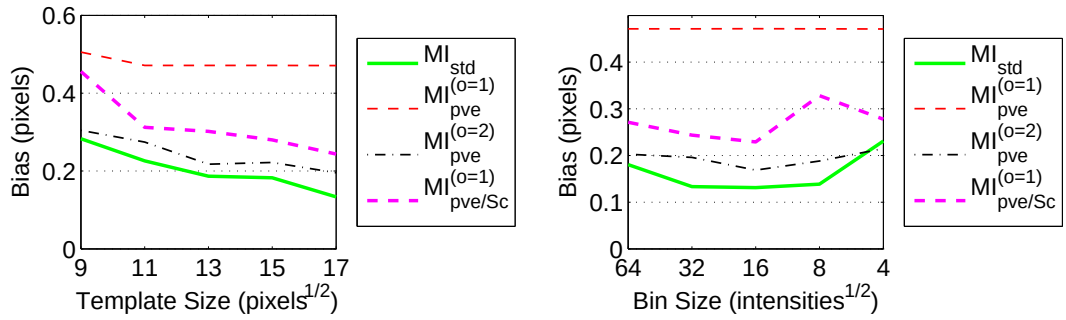


Figure 4.18: Convergence when increasing spatial support for lattice-offset templates and varying histogram bin-size.

optimisation algorithm. Second order PVE outperforms both of these functions, because it is smooth and does not suffer from a strong pattern of glitches. No further improvement was observed for third order PVE, except when using low bin-sizes, because higher orders of PVE populate more bins per sample. This indicates that increasing the PVE order beyond two does not significantly increase the radius of the basin of convergence.

In summary, increasing PVE order reduces the size of function glitches and hence reduces bias and improves convergence. However, beyond second order the performance improvements become increasingly marginal.



(a) (b)

Figure 4.19: Bias when desynchronising template and reference pixel-lattices for lattice-offset templates when varying *(a)* template size and *(b)* histogram bin-size.

4.4.4 Breaking Lattice Synchronisation

In this set of experiments, the effect of breaking lattice synchronisation was considered. To do this, the scale of the template was increased by 7.7%. Glitches in the function surface have a large effect on first order PVE, as has been noted [66], and a suggested solution is to prevent synchronisation of the template and reference lattice points by scaling the image slightly [48, 88].

The bias when varying bin-size and template size are respectively plotted in Figure 4.19a and b for STD, 1st order PVE, 2nd order PVE and 1st order PVE with a scaled template. A scale of 1.077 was chosen so that the pixels in the middle sized (13×13) template would have a range of phases (relative to the reference lattice) that was distributed over 360°.

Using an upscaled template reduced the bias substantially, but increasing the PVE order to two yielded larger reductions. Despite the scale being "tuned" to the 13×13 template, the bias continued increasing for larger template sizes. Bias for smaller templates improved less than it did for larger templates, implying that the distribution of phase should be at least 360° to disrupt the pattern of

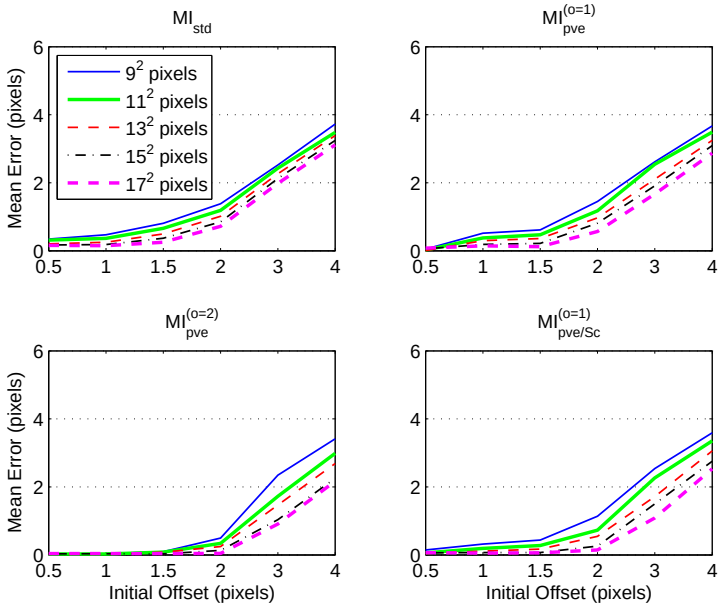


Figure 4.20: Convergence when desynchronising template and reference pixel-lattices for lattice-offset templates of varying size.

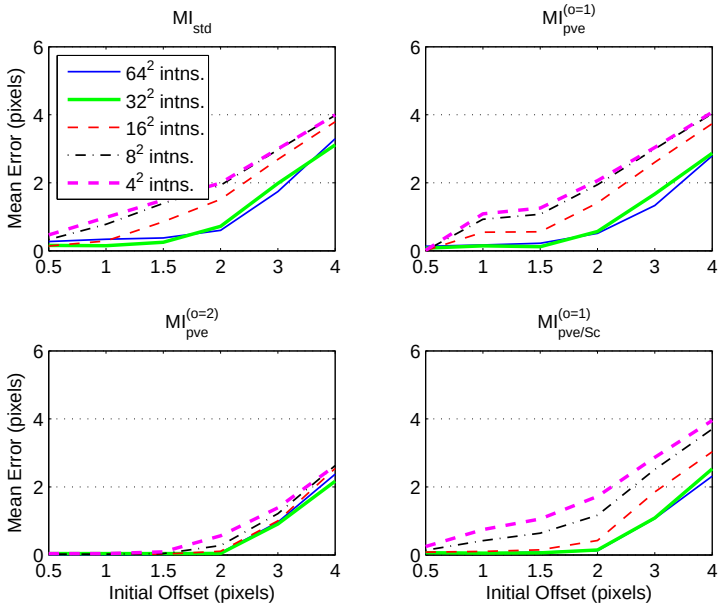
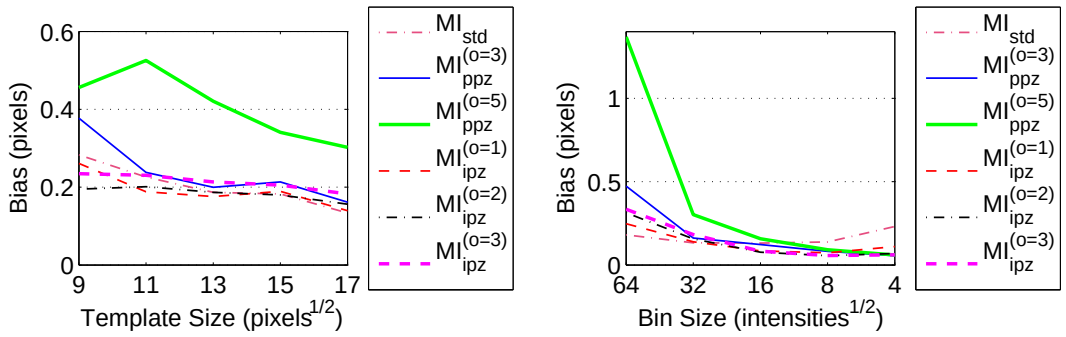


Figure 4.21: Convergence when desynchronising template and reference pixel-lattices for lattice-offset templates and varying histogram bin-size.



(a) (b)

Figure 4.22: Bias when increasing intensity support for lattice-offset templates when varying *(a)* template size and *(b)* histogram bin-size.

glitches. Similarly reduced biases were obtained when varying bin-size.

The convergence results, plotted in Figure 4.20 and 4.21, demonstrate that upscaling the template yields significant improvements in convergence for larger templates. Again, the performance improvement for templates below 13×13 in size is minimal, bearing out the conclusion that to disrupt glitch patterns, upscaling should be sufficient to give a distribution of phase over 360°. Again increasing PVE order to two gave greater improvements in convergence rates.

4.4.5 Increasing Intensity Support

This set of tests sought to examine the effects of increasing intensity support, where Parzen windows are used to blur the histogram with kernels of successively increasing size. The bias is respectively plotted in Figure 4.22a and b, for varying template size and intensity bin-size, for six MI functions: STD, second and fourth order PPZ, first, second and third order IPZ.

The advantages of using second order PPZ over STD have already been discussed. In Figure 4.22a, where a fixed bin-size of 32 is used, increasing the PPZ order to

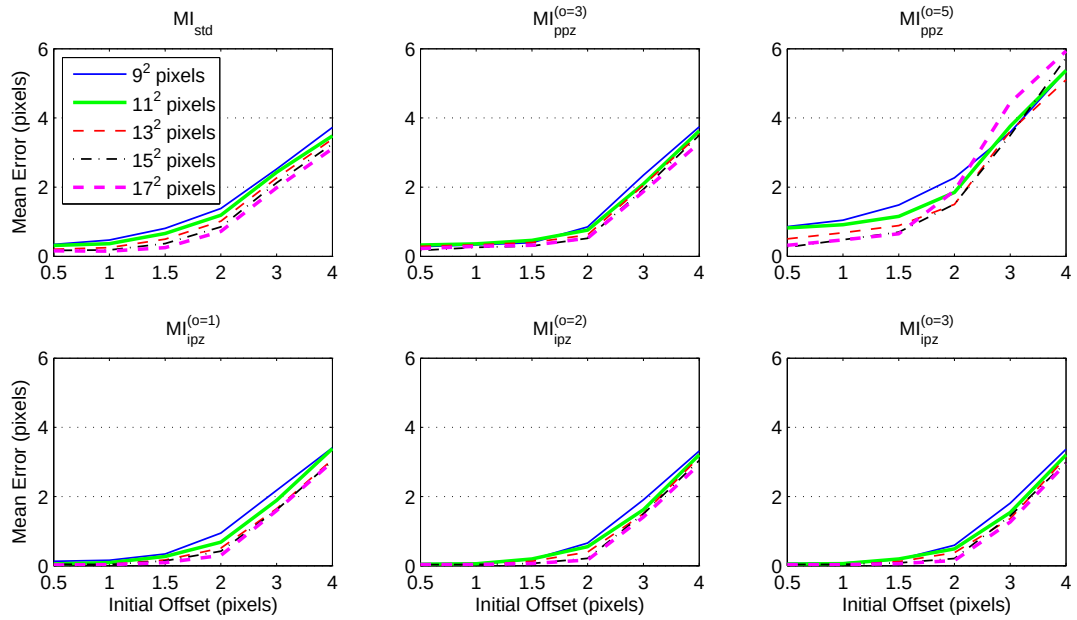


Figure 4.23: Convergence when increasing intensity support for lattice-offset templates of varying size.

four increases the bias, because the joint-histogram becomes over-blurred. However, Figure 4.22b shows that fourth order PPZ compares comparably to, but no better than, other methods for low bin-sizes. IPZ has a generally lower bias than PPZ, due its better use of (fractional valued) sample data. For IPZ, the bias also increases with PVE order at large bin-sizes, due to over-blurring. However, this trend is reversed at low bin-sizes, due to the sparsity of the histogram having an effect. The improvement is less significant between second and third orders.

The convergence results plotted in Figure 4.17 and 4.18, lead to similar conclusions where IPZ outperforms PPZ and STD at low initial offsets because it does not suffer from hiss. Again, increasing the IPZ order from one to two yields significant improvements for high resolution histograms, with a more marginal improvement thereafter.

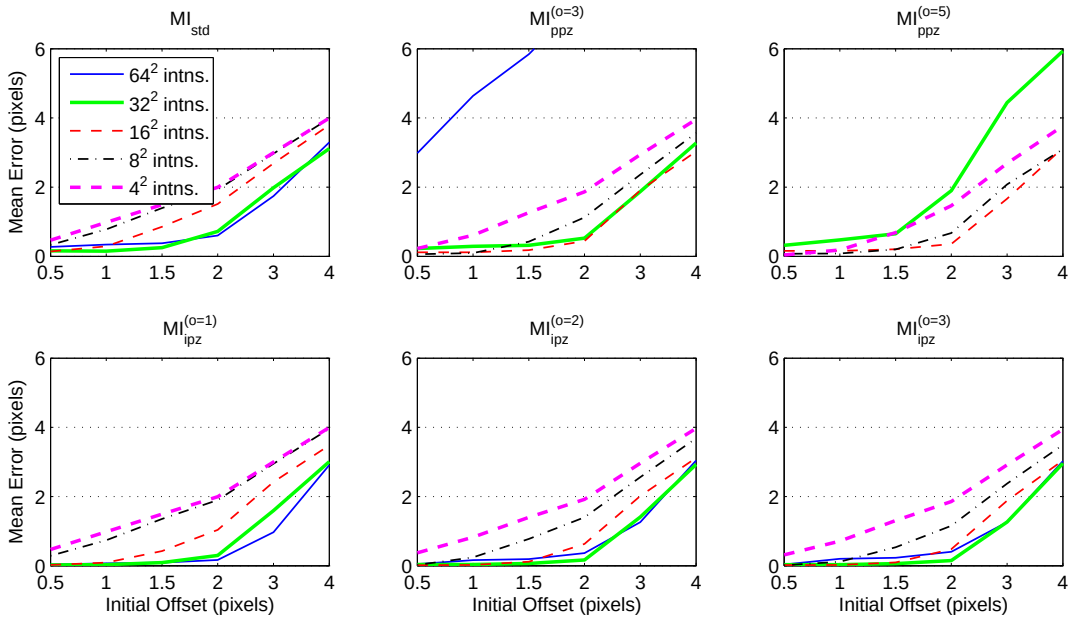
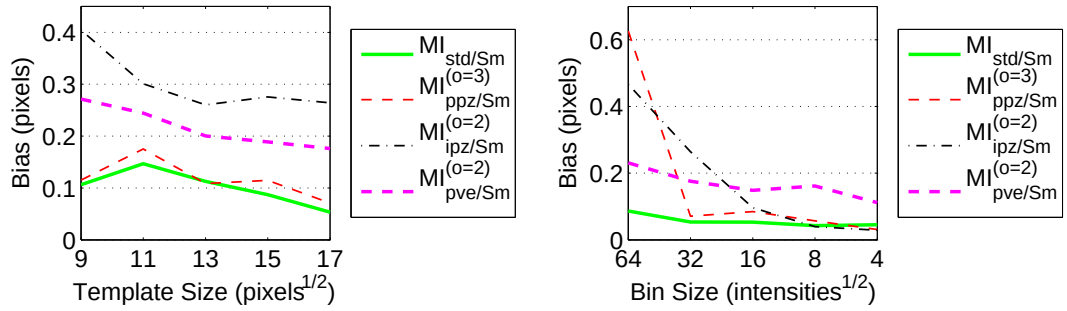


Figure 4.24: Convergence when increasing intensity support for lattice-offset templates and varying histogram bin-size.

In summary, increasing the intensity support can improve both convergence and bias, but successive increases in PVE order have a decreasing effect, as ever finer descriptions of the kernel and histogram add little new information. For large kernels, care should be exercised not to over-blur the histogram, as this induces ringing.

4.4.6 Smoothing the Images

The use of pre-smoothed images was also examined for possible improvements in performance. Smoothing was applied to the reference image and the template, before sub-sampling and extraction of the template. This approach was used to avoid negatively affecting the results by aliasing the intensities at the borders of the images. For the smoothing experiments, a 36×36 smoothing window was used



(a) (b)

Figure 4.25: Bias when smoothing images for lattice-offset templates when varying *(a)* template size and *(b)* histogram bin-size.

on the high resolution images: equivalent to a 3×3 window in the sub-sampled images.

The bias is respectively plotted in Figure 4.25a and b, when varying template size and intensity bin-size, for four MI functions: STD, 3rd order PPZ, 2nd order IPZ, and 2nd order PVE. Comparing Figure 4.25 to Figure 4.13, it is shown that smoothing generally reduces bias for STD, and PPZ. PVE is less affected, while for IPZ there is a large increase in bias.

For STD and PPZ an improvement in bias occurs, because smoothing reduces noise, while allowing each pixel to contain some information of a more global nature. This results in a better joint PDF estimation. IPZ uses the factional intensity data of every pixel already, so smoothing causes a net loss of information contained within each sample, resulting in an increase in bias. PVE shows little variation in bias, because second order PVE already has large spatial support, so smoothing gives no new information to the function.

Convergence when using smoothed data and a fixed bin-size of 32, plotted in Figure 4.26, is significantly improved for STD and PPZ, because the reduction of noise in the images also results in a reduction of hiss. For IPZ, a substantial

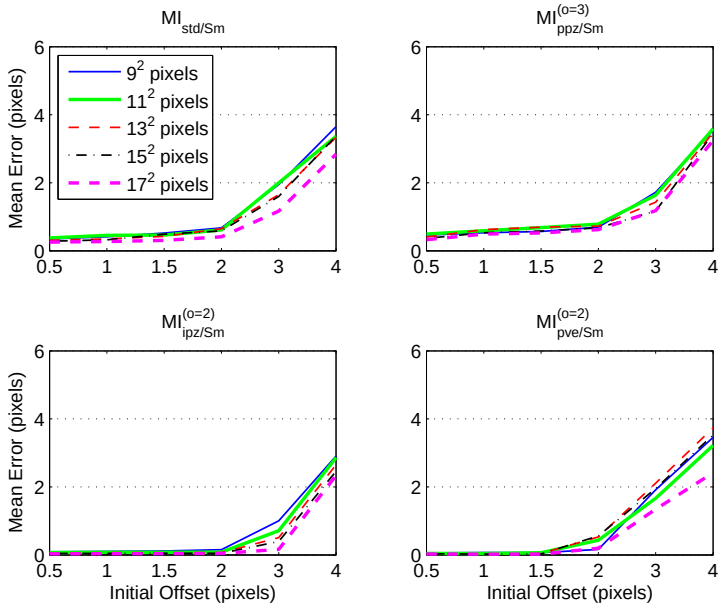


Figure 4.26: Convergence when smoothing images for lattice-offset templates of varying size.

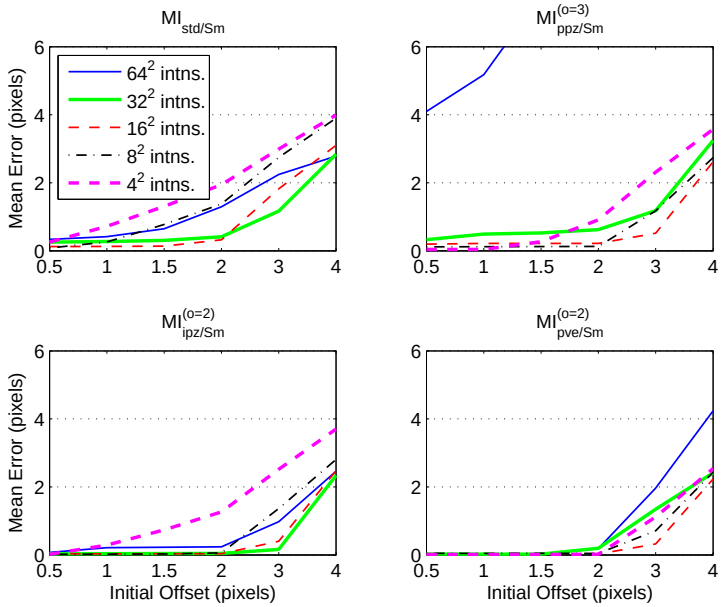


Figure 4.27: Convergence when smoothing images for lattice-offset templates and varying histogram bin-size.

improvement is also observed, because smoothing near the border is equivalent to increasing the size of the template slightly. The result is an increased radius for the basin of convergence, which is seen by the shift of the point of inflection to an initial offset of 3 pixels. The convergence performance of PVE shows little change, due to its already wide spatial support.

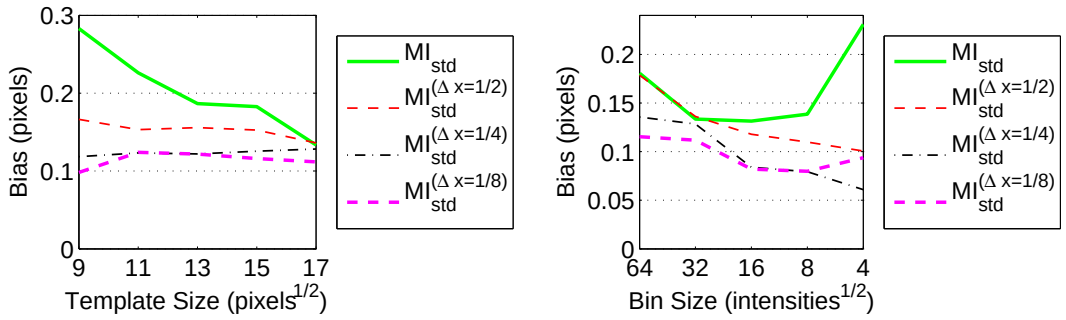
The convergence results, when varying the bin-size (plotted in Figure 4.27) allow similar conclusions to be drawn. However, the optimal histogram bin-size decreases for all the methods, because the sample data is less local. Over-blurring of the histogram also occurs more easily for smoothed data, as demonstrated by the reduction in performance for 64^2 pixel sized bins.

In summary, smoothing improves convergence, because of the inclusion of global data, unless the global data is already used, as it is in PVE. Smoothing also reduces hiss for STD and PPZ leading to a significant improvement in convergence. In general, bias is also reduced, unless local intensity information that is being used, *e.g.* for IPZ, is lost.

4.4.7 Increasing Sample Rate

In these tests, the effects of increasing the sampling rate are examined by successively increasing the number of samples per template lattice point. The bias is plotted in Figure 4.28a and b, when varying template size and intensity bin-size respectively, for four MI functions: STD with sampling rates of $1,2^2,4^2$ and 8^2 samples per template pixel.

As expected, increased sampling rates decreases the bias, since more of the information implicit in the 2D lattice structure of the images is used. Super-sampling also reduces the inaccuracy of the histogram estimate, as discussed in Section



(a) (b)

Figure 4.28: Bias when super-sampling for lattice-offset templates when varying *(a)* template size and *(b)* histogram bin-size.

4.1.1, and allows lower histogram bin-sizes to be used as shown in Figure 4.28b. The improvements in performance become increasingly marginal, since the information that may be extracted from sampled data is limited. Despite the decreasing rate of improvement, lower biases are obtained than for any of the functions tested thus far. Moreover, the bias may be controlled more finely by the choice of the sample rate, since the lattice structure implicit to the images is used, partially eliminating the assumption of independence and identical distribution.

Increasing the sample rate also results in improved convergence rates, plotted in Figure 4.29 and 4.30. In particular, the mean convergence error at low initial offsets approaches zero, as the amplitude of hiss relative to the macroscopic trends in the function surface decreases. The fact that hiss and convergence error may be finely controlled by the sample rate is clearly demonstrated in Figure 4.29 and 4.30, by the steady improvement in convergence and bias as the sample rate is increased. The performance of STD also improves substantially, at high histogram resolutions, because of the increased number of available samples. Hence, the optimal histogram bin-size decreases as well.

In summary, the use of increased sample rates can improve performance, like

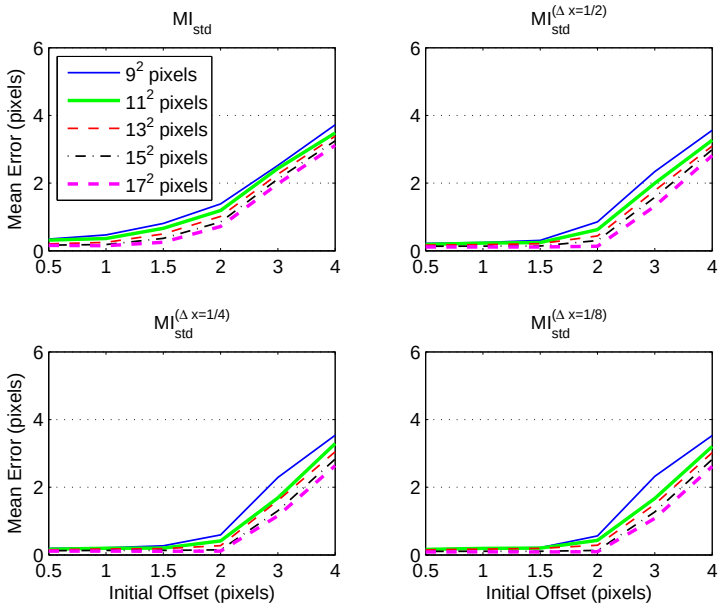


Figure 4.29: Convergence when super-sampling for lattice-offset templates of varying size.

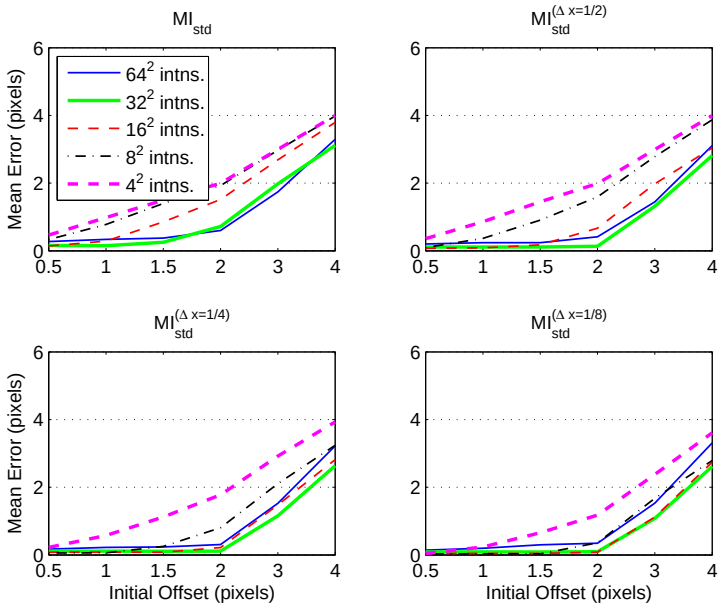


Figure 4.30: Convergence when super-sampling for lattice-offset templates and varying histogram bin-size.

all the methods discussed above. However, unlike other methods, controlling the sample rate allows *fine* control of the bias and mean convergence error. The sample rate can be varied according to the step size in an optimisation algorithm, because the shape of the function surface does not change substantively if the sample rate is varied slowly. Other methods, such as changing the order of the function, have too coarse an effect to be used in this way. Moreover, the effects of hiss may be almost entirely eliminated. STD is generally substantially cheaper than IPZ or PVE to compute and has lower bias. PPZ can improve the results of STD at low sample rates.

4.5 Conclusions

In this Chapter, it was shown that all sampling methods give an approximate joint PDF of intensity. This approximation causes two types of artefacts: hiss (caused by function discontinuities) and glitches (caused by harmonics between the template and reference image lattices). Hiss and glitches affect registration performance in two ways: inducing spurious local minima and shifting the global minimum away from its "true" position (bias). Many techniques have been suggested to improve registration performance, all of which were compared using the theory established in Chapter 3. Comparisons in terms of both bias and convergence rate were made, where previous work has considered convergence rate only, using a comprehensive array of tests (1.0 million).

Bias turned out to be an important consideration, since in many cases it was a substantial fraction of a pixel or more. As expected, increasing template size improves convergence rate and bias, since more samples give a more representative histogram. Most functions have an optimal bin-size of intermediate value, since

low bin-sizes lead to overly sparse histograms and poor statistical models, and large bin-sizes result in over-blurring when using kernel functions.

PVE generally gives the best convergence due to its wider basin of convergence. However it also exhibits a large bias giving inaccurate solutions, even when the spatial support is increased. The large glitches of first order PVE were greatly reduced by upscaling the template by 7.7%, but this yielded less of an improvement than increasing the spatial support. Moreover, scaling allows strong features near the edge of the template to bias the position of the minimum, because their relative shift in position is greater. Other methods do not have this disadvantage.

Pre-smoothing the images allows STD and PPZ to achieve similar convergence performance to IPZ. In the case of STD this comes at much lower computational cost. For IPZ the bias increases for smoothed data, because the loss of local intensity information that IPZ already uses, while PVE is unaffected by smoothing.

Discontinuities in STD and PPZ prevented convergence errors of zero at positions close to the function minimum, because hiss overwhelms the macroscopic trends in the function surface. However, increasing the sample rate allows hiss to be reduced, with the added advantage of reducing the already low bias of STD as well. Moreover the improvements in hiss and bias may be finely controlled, so they may be applied during function optimisation. Other parameters, such as function order, are too coarse to be used in this manner. The advantage of using super-sampled STD is that it often comes at lower computational cost than other methods and allows the bias the final solution to be controlled as well, because it partially eliminates the assumption of independence and identical distribution.

In Chapter 5 the idea of super-resolution is followed to its logical conclusion where an algorithm has been developed that integrates the effect of taking infinite samples when obtaining a histogram.

Chapter 5

NP-windowing applied to Mutual Information

Despite its benefits, the registration accuracy for MI may still be improved upon. Due to the limited number of samples available in some applications like tracking, the resolution of the intensity-histogram is limited by the relationship given in Section 3.1.3 to avoid overly sparse histograms becoming statistically meaningless [81, 27]. Even then, the MI value has a variance dependent on the histogram bin-size and number of samples [56, 34]. Also, the position of the image pixels relative to the world is arbitrary, as are the histogram bin boundaries relative to the light spectrum. So small changes in position, can result in unpredictable changes in the PDF and instability in the MI value, as was shown in Chapter 4. Finally, using each pixel as a single sample assumes independence and identical distribution. The fact that these assumptions are often incorrect, is generally ignored.

Parzen windowing [81] and Partial Volume Estimation (PVE) [49, 14] have been proposed to improve PDF estimation and MI stability. Parzen windowing accounts for uncertainty in intensity by blurring the histogram slightly. PVE ex-

plicitly smooths MI values for small shifts by using distance values to weight the relative effects of nearby pixels. However, both methods require sufficient samples to populate the histogram and may suffer from artefacts and bias [66]. Parzen windowing requires a kernel to be specified beforehand, and the histogram bin-size must be specified for PVE and Parzen Windowing depending on the number of samples.

A method addressing these shortcomings was proposed by Kadir and Brady, where the statistics in arbitrary regions in an image were estimated [37, 38]. This is the foundation of this chapter. However, in this chapter the more complicated case of a pair of 2D images, is considered. In addition, substantial simplifications to the theory are made using Green's Theorem and standard polygon rendering methods, which avoids having to consider multiple geometric cases. This method is referred to as NP-windowing.

Essentially, NP-windowing is just another method for obtaining the intensity PDF for images and joint-images. But it is equivalent to sampling the images at infinite resolution. Unlike more traditional sampling methods, arbitrary selection of a kernel is not required. Moreover, the spatial structure inherent to images, specified as a lattice of pixel values, *is* used.

In this chapter, the joint-distributions obtained from NP-windowing were used to obtain better estimates of Mutual Information. In the registration framework established in Section 2.2, NP-windowing yielded improved convergence rates with less bias. NP-windowing also has a broader application to estimating image statistics and the measurement of entropy *e.g.* in saliency detection [36]. Importantly, the implementation of NP-windowing is sufficiently rapid in terms of computation for practical use. This has two important implications: MI may be used in applications where few samples are available, such as tracking small im-

age patches; and the position of maximum MI is closer to ground truth position, resulting in improved registration accuracy.

Independently, Rajwade *et al.* have also developed a similar density estimation method [69] for joint-distributions. Rajwade's method works slightly differently to this method, in that it assumes constant gradient over each pixel. The gradients from each image form pairs of bounding parallel lines in intensity space allowing the histogram to be incrementally populated by parallelograms. The constant gradient assumption is not true and generates geometric singularities that need to be solved by arbitrary limiting factors.

Moreover, Rajwade's method is never extended to actually registering images, and geometric singularities are solved by limiting the extent of the parallelograms, which decreases accuracy. The methods discussed in this chapter are more general, and unlike Rajwade's method, the geometric constraints used are explicitly derived from the problem formulation and interpolation method.

The rest of the chapter is organised as follows. In Section 5.1, the method for constructing PDFs from neighbourhoods of intensity values using the proposed NP-windowing method is presented. Section 5.2 discusses the combining of multiple neighbourhoods into a complete histogram, and the subsequent computation of MI. A resolution aware version of the proposed NP-windowing method is also presented. Next, in Section 5.3, the special case of interpolation methods with non-constant PDFs within each neighbourhood is considered. Section 5.4 discusses some experiments and results, before the Chapter concludes in Section 5.5.

Please note that in this Chapter a slight notational digression from the convention in Section 2.2 is made. Previously the reference and template images were denoted f_r and f_t. In this chapter they are instead respectively referred to as f_1 and f_2

and treated as two interchangeable signals, which makes the explanation clearer.

5.1 MI from NP-windowed PDFs

This section describes the proposed NP-windowing method used to obtain a PDF, and hence an MI value. To begin with, recall from standard probability theory [62] that a function of a random variable creates another random variable whose distribution may be determined by the *transformation formula* between the two.

To illustrate, consider a group of adjacent pixel values in a template image (*e.g.* α_1, β_1, and γ_1 in Figure 5.1a). In a registration application, the pixel values have corresponding values in a second reference image (*e.g.* α_2, β_2, and γ_2 in Figure 5.1b). Intensity f at any point $\mathbf{x} = (x_1, x_2)$ may be computed using an interpolation method utilising the α, β and γ values. In Figure 5.1a and b colour is used to indicate the half bi-linear interpolated (Half-BLI) intensity of two images as it varies over a 1×1 pixel region. Half-BLI uses three values from a triangular-shaped region for interpolation.

Each pixel is split into two triangles, denoted (α_i, β_i, γ_i) and (α_i, β'_i, γ_i) in Figure 5.1a and b. The two corresponding sets of intensity triplets from corresponding positions in the template and reference image form a *neighbourhood* of six values. The size of the neighbourhood varies with the interpolation method used, *e.g.* for bi-cubic interpolation the neighbourhood-size is 32: 2 images x 16 points; and for bi-linear interpolation the neighbourhood-size is 8: 2 images x 4 points.

The position of the lattice points relative to the world is arbitrary, so $\mathbf{x}$ is treated as a random variable with bounded uniform distribution: $0 \leq x_1 < 1$ and $0 \leq x_2 < 1$. The bounds arise naturally from the spacing between the pixels (also referred to as the *lattice-spacing*). This implies that the intensities of the two

images, $\mathbf{f} = (f_1, f_2)$, are also a random joint variable, with a distribution that may be found using a transformation formula. The bounds in $p_{\mathbf{x}}$ are also transformed to form bounds in $p_{\mathbf{f}}$. The bounded region of intensity probabilities is called the *intensity polygon*.

Notice how the intensity polygon for a square pixel, consists of two triangles, corresponding to each pair of triangles in f_1 and f_2. The vertex coordinates in the joint-PDF directly correspond to the intensity values in the two images, and each triangle corresponds to a single neighbourhood. The probability over each triangle should sum to one, hence the smaller triangle has a higher weighting in Figure 5.1c. Finally, notice that for other sampling methods, the choice of diagonal can critically affect the result, while for NP windowing it merely has the effect of flipping the shared edge between the two triangles in the joint-PDF.

This leads to a four step approach:

1. Obtain all neighbourhoods in image 1 and 2 for a given warp and interpolation method.

2. Calculate the intensity polygon for each set of samples.

3. Render the intensity polygons to a histogram. Alternatively, locate all polygon intersections and generate a new list non-overlapping intensity polygons.

4. Calculate the MI.

Step 1 simply requires the extraction of all neighbourhoods. This step is not discussed further except to mention that interpolation methods will, in general, only approximate the true band-limited signal and hence the true PDF. Better

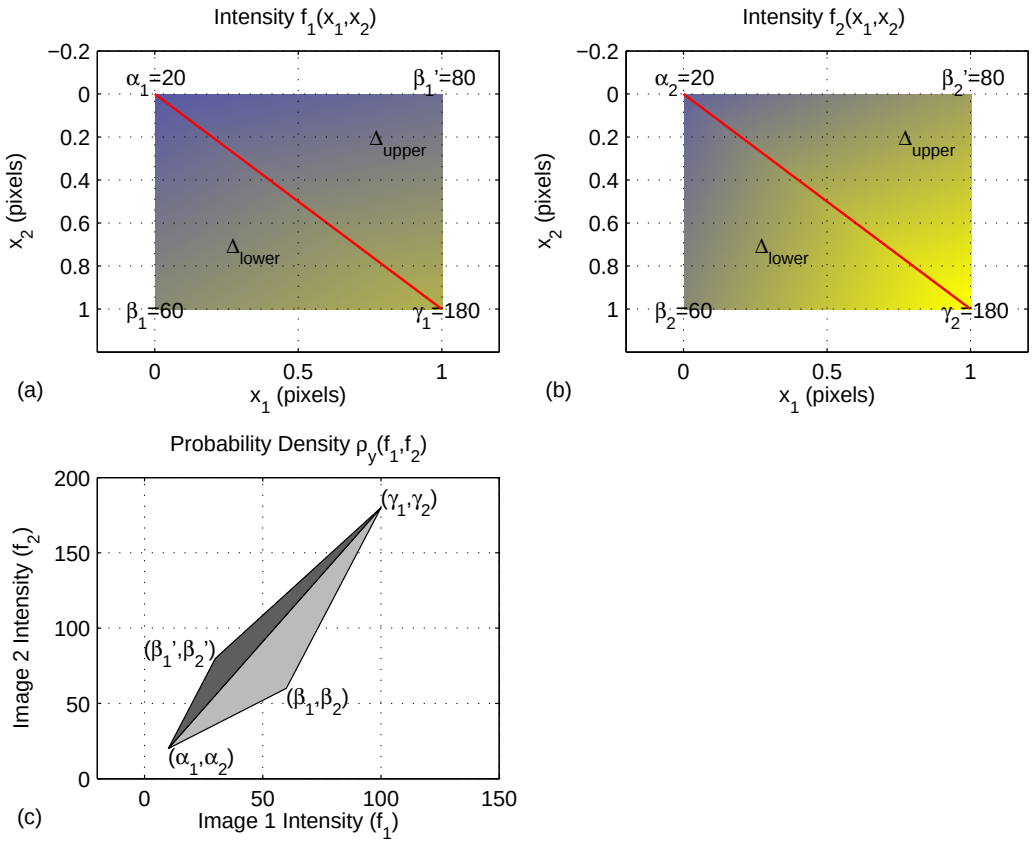


Figure 5.1: *(a)* & *(b)* Interpolated intensities for two images: respectively $\mathbf{f}_1(\mathbf{x})$, and $\mathbf{f}_2(\mathbf{x})$, where blue indicates low intensity and yellow indicates high intensity. Half bi-linear interpolation is used, requiring each pixel to be split into two triangles (Split by the red line). Each corresponding triangle pair in (a) and (b) describes a triangle of uniform probability in the joint-PDF of intensity, shown in *(c)*. For each triangle in the joint-PDF, the probability must integrate to one, hence the smaller triangle has a higher weight, indicated by the darker shade of gray. Note how the vertices of the two triangles in (c) correspond to intensity values in (a) and (b) respectively. Note also, that the choice of diagonal in (a) and (b), will simply change the shared diagonal between the two triangles in the joint-PDF, *i.e.* this choice negligible effect on the NP-windowing method.

approximations of signal may be obtained if information about the sampling pre-filter is available, or has been estimated using empirical methods [87]. For nearest neighbour interpolation this method breaks down to first order PVE as defined in (3.5b).

5.1.1 Single linear signal

For illustrative purposes step 2 is first discussed for the case of a single linear interpolated 1D signal. Pairs of neighbouring samples are required for linear interpolation, so the neighbourhood is of size two. Each pair of adjacent samples: α and β defines a straight line, denoted as

$$f = ax + b \quad | \quad 0 \leq x < 1 \tag{5.1}$$

where x and f are simply the scalar (1D) forms of spatial position $\mathbf{x}$ and intensity $\mathbf{f}$. The symbols a and b allow a simplified equation to used, and relate to the neighbourhood as follows: $a = \beta - \alpha$, and $b = \alpha$.

As discussed, x is treated as a uniformly distributed random variable: $p_x(x) = 1, \quad 0 \leq x < 1$. The PDF of f, $p_f(f)$ is simply a transformed version of p_x, obtained using the inverse function $x(f) = \frac{f-b}{a}$.

$$p_f(f) = |J(f)| p_x(x(f)) \tag{5.2}$$

where $|J| = |\frac{\partial \mathbf{x}}{\partial \mathbf{f}}|$ *i.e.* the determinant of the Jacobian of $\mathbf{x}$ with respect to $\mathbf{f}$. Conveniently in this case $|J(f)| = |a|$, which is constant, so

$$p_f(f) = \begin{cases} \frac{1}{|a|} & f \in [a; a+b) \\ 0 & \text{elsewhere} \end{cases} \tag{5.3}$$

p_x acts as a bounding function, giving the bounds on p_f shown in (5.3).

5.1.2 Joint linear signal

For a *joint* linearly interpolated signal, two equations describe the signals: $f_1 = a_1x_1 + b_1$ and $f_2 = a_2x_2 + b_2$. Both signals are 1D but to keep the Jacobian square, separate variables for the spatial position (x_1 and x_2) are used. The inverse equations are $x_1 = \frac{1}{a_1}(f_1 - b_1)$ and $x_2 = \frac{1}{a_2}(f_2 - b_2)$. The determinant of the Jacobian becomes:

$$|J| = \begin{vmatrix} \frac{\partial}{\partial f_1}x_1(f_1, f_2) & \frac{\partial}{\partial f_1}x_2(f_1, f_2) \\ \frac{\partial}{\partial f_2}x_1(f_1, f_2) & \frac{\partial}{\partial f_2}x_2(f_1, f_2) \end{vmatrix} = \begin{vmatrix} \frac{1}{a_1} & 0 \\ 0 & \frac{1}{a_2} \end{vmatrix} = \frac{1}{a_1a_2} \tag{5.4}$$

As this is a joint case, the PDF has two dimensions and the boundary becomes a 2D polygon. In this simple linear case, the polygon is a rectangle bounded by $0 \leq x_1 < 1$ and $0 \leq x_2 < 1$, *i.e.* $a_1 \leq f_1 < a_1 + b_1$ and $a_2 \leq f_2 < a_2 + b_2$.

5.1.3 Joint half-bilinear image

Compared to the single and joint 1D case above, the derivation of $|J|$ and the subsequent evaluation of the PDF is rather more complex for pairs of 2D images. Initially the simplest useful 2D interpolation method is considered: half Bi-Linear Interpolation (Half-BLI). Half-BLI uses neighbourhoods of three values from each image: interpolating within two triangular areas. Figure 5.1 shows two such neighbourhoods where Half-BLI is used. This yields two equations of the form

$$f_i = a_ix_1 + b_ix_2 + c_i \tag{5.5}$$

where $i \in \{1; 2\}$ is the index of each image. Again, a, b, and c are calculated from α, β, γ for the simplified form used in (5.5). These values are calculated as

follows:

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} -1 & 1 & 0 \\ 0 & -1 & 1 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix} \tag{5.6}$$

In this case the Jacobian is naturally square so a determinant may be obtained directly. In other cases, discussed in [62], a dummy variable is required to make the Jacobian square. The pair of inverse functions is:

$$x_1 = \frac{c_1 b_2 - b_1 c_2 - b_2 f_1 + b_1 f_2}{b_1 a_2 - a_1 b_2} \quad x_2 = \frac{c_1 a_2 - a_1 c_2 - a_2 f_1 + a_1 f_2}{-b_1 a_2 + a_1 b_2} \tag{5.7}$$

So $|J|$ is:

$$|J| = \begin{vmatrix} \frac{\partial x_1}{\partial f_1} & \frac{\partial x_2}{\partial f_1} \\ \frac{\partial x_1}{\partial f_2} & \frac{\partial x_2}{\partial f_2} \end{vmatrix} = \begin{vmatrix} \frac{b_2}{a_1 b_2 - b_1 a_2} & \frac{-a_2}{a_1 b_2 - b_1 a_2} \\ \frac{-b_1}{a_1 b_2 - b_1 a_2} & \frac{a_1}{a_1 b_2 - b_1 a_2} \end{vmatrix} = \frac{1}{a_1 b_2 - b_1 a_2} \tag{5.8}$$

The Jacobian for each neighbourhood is twice the inverse area of the intensity polygon (triangle) defined by the points (α_1, α_2), (β_1, β_2), and (γ_1, γ_2). This is exactly what one would expect for a constant $|J|$, since each triangle represents half a sample (pixel), so each pixel integrates to one in the histogram.

5.2 Obtaining the PDF estimate

Having obtained the intensity polygon for each neighbourhood, step 3 may now be considered: obtaining the PDF estimate for the entire image. This simply consists of summing the distributions for each neighbourhood together:

$$\rho_f(\mathbf{f}) = \frac{1}{N_n} \sum_{n=1}^{N_n} \frac{1}{|J_n(\mathbf{f})|} f_{xn}(\mathbf{x}_n(\mathbf{f})) \tag{5.9}$$

where n indexes each neighbourhood. N_n is the total number of neighbourhoods and is used to normalise the values so that the the PDF integrates to one. (5.9) is

expressed in a general form to indicate that the signals may have more than one spatial dimension $\mathbf{x}$ and that there may be more than one signal with co-varying amplitudes, $\mathbf{f}$.

Two methods for evaluating (5.9) were considered. First, the list of polygons may be intersected, with contributions from each intersecting region summed, to obtain a new list of non-overlapping intensity polygons. Even in the relatively simple case of half-BLI, which has constant distribution within each polygon, this is time consuming, because the process is $O(N_n!)$. So intersection and summing is only practical for small numbers of polygons: 40-50 polygons in the implementation tested for this work. For more complex interpolation methods, where the distribution varies within each polygon, intersection and summing becomes even less practical.

The second option, favoured in this work, is to use existing polygon drawing methods to "render" each polygon to an approximate PDF: *i.e.* a histogram. Unlike standard rendering algorithms, the drawing routine sums rather than replaces "pixel" values. The downside with this approach is a slight loss in accuracy, dependent on the histogram bin-size. Distributions that vary within a polygon may be represented as a "texture" to be rendered.

One could expect that the cost of evaluating the histogram is dependent on the number of neighbourhoods, N_n, and the average image gradient, *i.e.* approximately $O(\overline{\frac{\partial \mathbf{f}}{\partial \mathbf{x}}} N_n)$, since the larger the gradient the larger the area of the intensity polygon to be rendered. However, the costs are relatively constant since intensity polygons are added to the histogram using a standard rendering routine with flood filling at the polygon centre and (precise, intersection based) anti-aliasing at the edges. The anti-aliasing at the polygon edges uses the bulk of the computational cost of rendering.

5.2.1 Special geometric cases

Special cases where polygons collapse to lines or points are easily detectable, because their area is zero, and points have a perimeter of zero. Points are modelled as unit impulses in the PDF.

Since the intensity distribution is constant, lines are normalised by their length and split into segments wherever they intersect bin boundaries. Bins are incremented by the proportion of each line's length they contain. Non-varying distributions would use the same procedure but require explicit integration over each segment.

5.2.2 Computation of MI

MI may be obtained directly from the PDF estimate using the standard equation:

$$d_{mi} = \int_{\forall \mathbf{f}} \rho_{\mathbf{f}}(\mathbf{f}) \log \left(\frac{\rho_{\mathbf{f}}(\mathbf{f})}{\rho_{f_1}(\mathbf{f}) \rho_{f_2}(\mathbf{f})} \right) \mathrm{d}\mathbf{y} \tag{5.10}$$

where $\rho_{f_1} = \int \rho_{\mathbf{f}} \mathrm{d}f_2$ and $\rho_{f_2} = \int \rho_{\mathbf{f}} \mathrm{d}f_1$. In this case, since the PDF estimate is actually a histogram, the integral becomes a summation.

Due to the finite size of the bins and the limited number of samples, MI is generally under-estimated. The residual error was found by Moddemeijer to be approximately proportional to the square of the bin size [56] and inversely proportional to the number of samples. Since NP-windowing is equivalent to taking infinite samples, a major source of MI estimation error is completely eliminated by NP-windowing. Accurate PDFs may be obtained even when there are few (tens of) samples *e.g.* for small image patches.

The residual error due to bin size may also be eliminated by estimating MI directly from $\rho_{\mathbf{y}}$ *i.e.* without constructing a histogram. However, as already discussed,

this would require the intersection and summing of intensity polygons, which is too slow for practical use.

5.2.3 Resolution Aware PDF estimation

In the above NP-windowing implementation, the neighbourhood values for the template were obtained directly from the template lattice. However the warped template lattice points will generally not align with the reference lattice, as shown in Figure 5.2a, where the warped template coordinate system is shown in green in the reference image coordinate system.

This means that reference intensities are interpolated. So long as the scales of the template and reference pixel-lattices are similar, the underlying reference image will be well modelled. However as the relative scale increases, the reference image is sampled ever more sparsely, and the interpolated image models the original reference image ever more inaccurately. Naturally, this has a negative impact on the PDF estimate. To overcome this problem, a *resolution aware* version of the NP-windows algorithm is proposed.

The resolution aware method is principally the same as the existing approach, except that a locally irregular lattice is used to obtain each neighbourhood, like that shown in 5.2b. The lattice is chosen so that each point in both the reference and the template intersects the irregularly sampled grid, while maintaining rectangular regions. The triangles defined by each neighbourhood are normalised by the relative area of the rectangular region to preserve statistical accuracy, so (5.9) becomes:

$$\rho_y(\mathbf{y}) = \sum_{n=1}^{N_n} A_n \frac{1}{|J_n(\mathbf{y})|} f_{xn}(\mathbf{x}_n(\mathbf{y})) \tag{5.11}$$

The computational cost is approximately quadrupled when the relative scale of

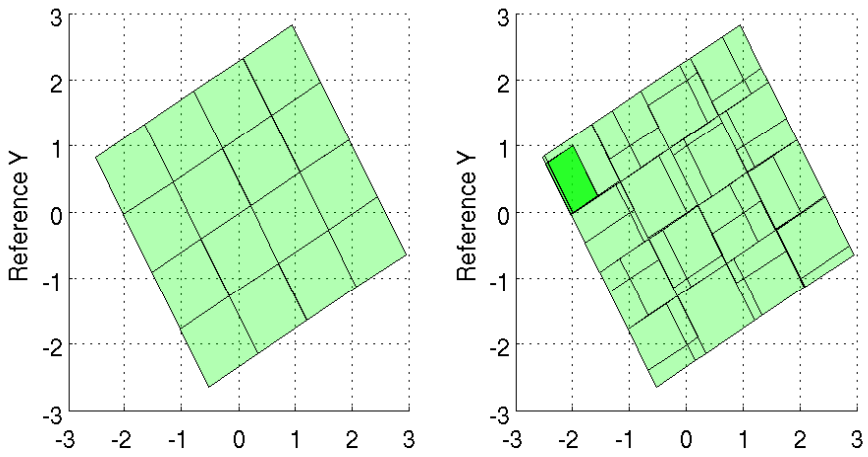


Figure 5.2: Template coordinate system (green region) warped into the reference image coordinate system. As shown the lattice points of each coordinate system generally will not line up, so interpolation is required to obtain the reference intensities. **(a)** In standard NP-windowing, a regular lattice is used for sampling, and the underlying reference image will be well modelled. However if the relative scale becomes too large, the reference image becomes too sparsely sampled to accurately model its statistics. **(b)** The resolution aware implementation uses a locally irregular lattice to ensure that every reference pixel within the template's bounds is sampled. Hence the reference image statistics are always correctly modelled.

the template and the reference image is one. Otherwise, the cost is dependent on the image with the higher resolution spacing in the reference coordinate system.

5.3 Bi-Linear Interpolation: A Varying Distribution

For some kinds of interpolation the distributions in histogram space are *not* constant within each polygon. One such example is considered: (full) bi-linear interpolated images. These distributions are more difficult to deal with, since they require explicit integration over their area to obtain a PDF.

Considering the example just given, the equations for a pair of bi-linearly interpolated images is:

$$f_i = a_i x_1 x_2 + b_i x_1 + c_i x_2 + d_i \tag{5.12}$$

where i indexes each image. This yields two sets of inverse functions of form $x_i(f_1, f_2) = P(f_1, f_2, 1) + \sqrt{P(f_1, f_2, 2)}$, where $P(..., \epsilon)$ indicates a polynomial of degree ϵ. This concise form is used because some of the following equations are difficult to read otherwise. Both sets of solutions give the same solution for $|J|$:

$$\begin{aligned} |J(f_1, f_2)| = \big[&(b_2c_1 - b_1c_2 + a_2(d_1 - f_1) - a_1(d_2 - f_2))^2 \\ &-4(a_2c_1 - a_1c_2)(b_2(d_1 - f_1) - b_1(d_2 - f_2))\big]^{-\frac{1}{2}} \end{aligned} \tag{5.13}$$

As for the half bi-linear case, the valid region is bounded by a polygon with vertices corresponding to the pixel values of the neighbourhood: (α_1, α_2), (β_1, β_2),

(γ_1, γ_2), and (δ_1, δ_2), with the relationship:

$$\begin{pmatrix} a_i \\ b_i \\ c_i \\ d_i \end{pmatrix} = \begin{pmatrix} 1 & -1 & 1 & -1 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \alpha_i \\ \beta_i \\ \gamma_i \\ \delta_i \end{pmatrix} \tag{5.14}$$

Evaluating $\int |J| \mathrm{d}f_2 \mathrm{d}f_1$ between such complicated boundaries is difficult, because multiple geometric cases must be considered [37]. This can be avoided by splitting the quadrilateral into two simple polygons (triangles) and using Green's Theorem to re-parameterise the problem.

Green's Theorem states $\int\int_R (\frac{\partial J_1'}{\partial f_1} - \frac{\partial J_2'}{\partial f_2}) \mathrm{d}A = \oint_C J_2' \mathrm{d}f_1 + J_1' \mathrm{d}f_2$, where C defines the curve around a region R. Letting $J_2' = 0$ and $J_1' = \int |J(f_1, f_2)|^{-1} \mathrm{d}f_2$ yields:

$$\rho = \oint_C J_1'(f_1, f_2) \mathrm{d}s \tag{5.15}$$

Now, $\mathbf{f}'(\mathbf{s}) = (m_j s + c_j, s)$ is substituted for $\mathbf{f} = (f_1, f_2)$, where m_j and c_j are respectively the gradient and f-intercept defining line segment j of the intensity polygon. The rules of integration by substitution are applied to (5.15) and (5.13):

$$\rho = \oint_C J_1'(m_j s + c_j, s) \mathrm{d}s \tag{5.16}$$

The result of the above (double) integration is an equation containing more than 100 terms and factors. For interest the form of the equation is:

$$\begin{aligned} \rho = k_1 \sqrt{P(s,2)} + P(s)(k_2 + \log[P(s) + k_3 \sqrt{P(s,2)}] \\ + k_4 \log[P(s) + k_5 \sqrt{P(s,2)}] + k_6 \log[\frac{P(s) + k_7 \sqrt{P(s,2)}}{P(s)}] \end{aligned} \tag{5.17}$$

Clearly the implementation of (5.17), although giving a precise result, would be too slow for practical use. Several hundred operations would be required just to

initialise the constants for each polygon. In addition, a few tens of operations would be required to calculate *every* bin overlapped by the polygon. This is in contrast to half bi-linear interpolation, where nine operations (4 subtractions, 2 multiplies, 3 additions) are required to calculate the area of each triangle. A flood fill routine (one add per bin) is used over most of the overlapped region, *i.e.* Half-BLI is about two orders of magnitude faster than BLI. Using bi-cubic interpolation (BCI) in an NP-windowing framework turns out to be even more problematic. The intensity equations for BCI have form $f_i = P(x_1, x_2, 3)$, and these are not invertible.

For BLI, the potential improvement in accuracy is limited and comes at too great a computational cost for practical use. Hence, an NP-windowing technique for BLI and a pair of images was not implemented, because Half-BLI was found to work well, as shown in the results section. BLI for a single image is far simpler to implement and has practical use in applications such as saliency detection and estimation of image statistics. This method *was* implemented as is now discussed.

5.3.1 Applying Green's Theorem to BLI for a single image

A more applicable example, using Green's theorem to obtain the PDF of a *single* bi-linearly interpolated image is proposed in this section. This method would be useful for measuring local image statistics and interest point detection [36]. BLI has the following form:

$$y_1 = ax_1x_2 + bx_1 + cx_2 + d \tag{5.18}$$

.

To obtain the square matrix necessary for obtaining the determinant of the Jacobian a dummy variable is used: $y_2 = x_1$, which is integrated out in the final

step. The inverse equations of x_1 and x_2,in terms of y_1 and y_2, are:

$$x_1(f_1, f_2) = f_2 \tag{5.19}$$

$$x_2(f_1, f_2) = \frac{f_1 - bf_2 - d}{af_2 + c} \tag{5.20}$$

yielding the Jacobian:

$$|J| = \begin{vmatrix} 0 & 1 \\ -\frac{1}{af_2+c} & \frac{-b(af_2+c)+a(-bf_2-f_1+d)}{(af_2+c)^2} \end{vmatrix} = \frac{1}{af_2 + c}$$

Solving the area integral in the general case simply requires integrating with respect to f_2:

$$\int \frac{1}{af_2 + c} \mathrm{d}f_2 \mathrm{d}f_1 = \int \frac{1}{a} \log(|af_2 + c|) \mathrm{d}f_1 \tag{5.21}$$

.

Again the neighbourhood defines an intensity polygon in the intensity PDF, where each edge of the polygon denotes a linear relationship between f_1 and f_2: $f_2 = m_j f_1 + n_j$, where k indexes the current edge. Substituting this and integrating yields:

$$\int \frac{1}{a} \log(a(m_j f_1 + n_j) + c) \mathrm{d}f_1 \tag{5.22}$$

$$= \frac{1}{m_j a^2} [(a(m_j f_1 + n_j) + c) \log(|a(m_j f_1 + n_j) + c|) - m_j a f_1]$$

However there are some geometric special cases that need to be considered, since without care (5.22) can become indeterminate. These cases are tabulated in Table 5.1. The actual derivations are trivial using L'Hopitals rule and are not discussed further.

The above method yields the same results as Kadir and Brady's earlier work [37], but with lower algorithmic complexity. In the original approach [37], 24 individual cases needed to be considered. The above method, utilising Green's theorem,

Description	Mathematically	Derivation
Intensity Polygon is a parallelogram	$a = 0$	$\int \frac{1}{2c} f_1(2n_j + m_j f_1)$ (*i.e.* same as half-BLI case)
Constant Intensity	$a = c = 0$	This is a point. Use half-BLI.
Constant Gradient		This is a line. Use half-BLI.
Horizontal Line	$m_j = 0$	$\frac{1}{a} f_1 \log(\lvert ak_{f2} + c \rvert)\, k_{f2} = 0$ or 1
Vertical Line	$m_j = \infty$	0
Singularity	$a(m_j f_1 + n_j) + c = 0$	0

Table 5.1: Special cases when integrating Jacobian for a single BLI image

considers only the special cases in Table 5.1. The Matlab code of the above implementation consists of 250 lines, most of which were required for polygon slicing routines. An example, comparing the histograms obtained using various sampling methods, is shown in Figure 5.3.

5.4 Experiments

To demonstrate the superiority of NP-windowing when estimating MI it was compared to the current state of the art MI registration methods. In all, four methods were compared: NP-windowing, standard sampling (STD) (One sample per pixel), third order partial volume estimation (PVE) [14] and second order B-spline Parzen windowing (IPZ) [81]. Since an analytic derivative for NP-windowing is beyond the scope of this work, optimisation was performed using Powell's Direction Set method [68]. In all cases MI was maximised for 2DoF translation warps only. BLI was used to obtain reference image values corresponding to template lattice points. Half-BLI was used as the interpolation model to populate the histogram for NP-windowing. The test setup described in Section 3.4 was used.

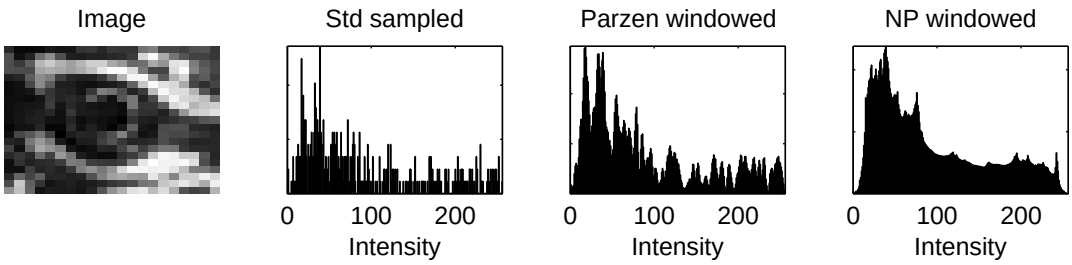
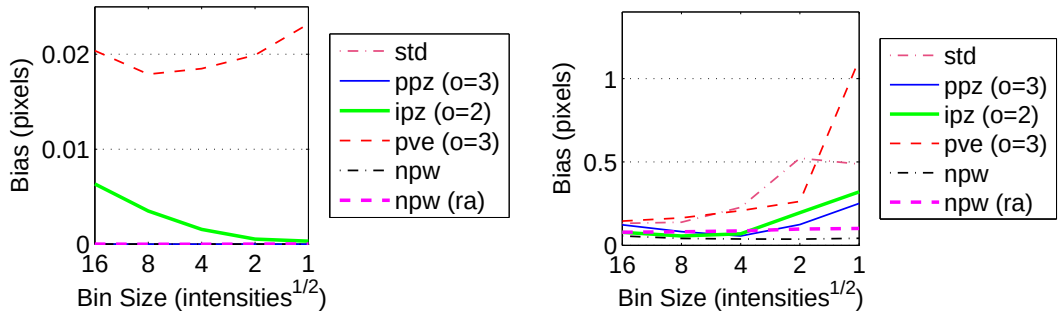


Figure 5.3: Some examples of histogram obtained from the image on the far left using different sampling methods. From left to right these are: standard sampling, Parzen windowing and NP-windowing. Notice the discrete changes between bins in standard sampling, where one expects a PDF to be smooth. Parzen windowing gives an improved estimate by explicitly accounting for the uncertainty in intensity samples, but this is simply a convolved version of the standard sampled histogram, so it still exhibits sudden changes and under-sampling in certain regions. NP windowing on the other hand exhibits a smooth nature where the statistics in all regions are well described.



(a) (b)

Figure 5.4: Bias when varying *(a)* template size and *(b)* histogram bin-size, for lattice-aligned templates.

Again performance was measured in terms of bias and convergence, for five template sizes ranging from 9^2 to 17^2 pixels, with a constant histogram bin-size of 16^2 intensities. Performance was also measured as the bin-size in the histogram varied exponentially from 16^2 to 256^2 bins, with a constant template size of 17^2 pixels.

The results for bias while varying the number of intensity bins is given for both lattice-offset and lattice-aligned templates in Figure 5.4. The convergence results, when varying bin-size, are plotted in Figure 5.5 and 5.6 for lattice-aligned and lattice-offset templates respectively. Not unexpectedly the bias is low, if not zero, for the lattice-aligned templates, as the lattice-aligned template has a strong global minimum. Both NP-windowing methods and PVE give zero bias in these cases respectively, because of the high accuracy of NP-windowing and the biasing of PVE towards lattice-aligned positions [66]. However, lattice-alignment is unusual and these results are given to contrast them with results from a lattice-offset aligned template.

As demonstrated in Figure 5.4 bias is larger in the common lattice-offset case. In the lattice-offset case, notice that NP-windowing is again the best performer

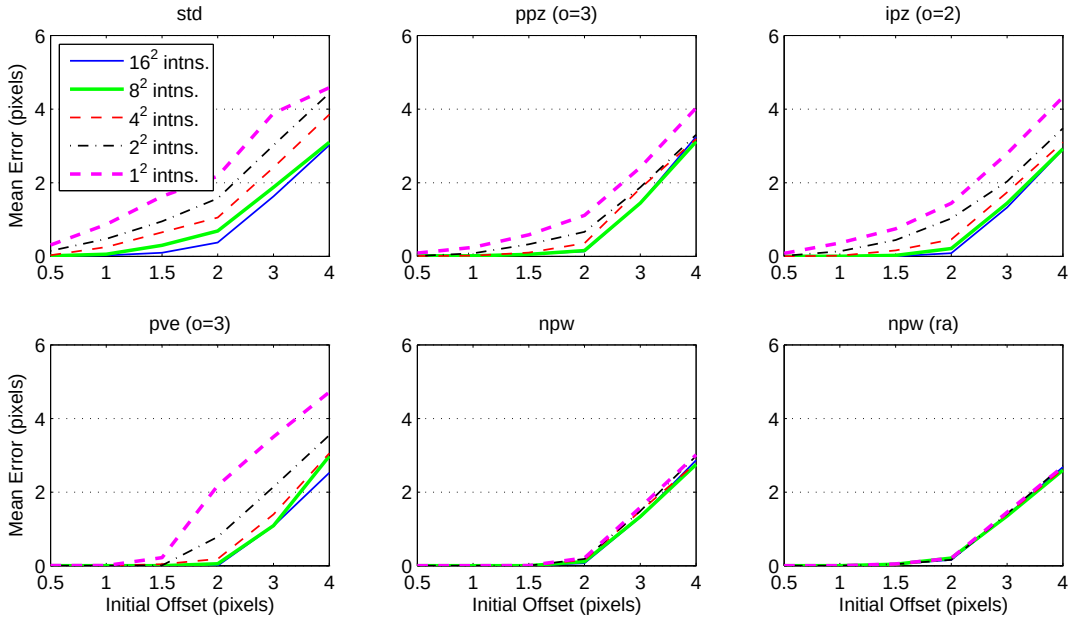


Figure 5.5: Plots showing convergence for varying *histogram bin-size* when using lattice-aligned templates

and unlike the other methods is almost completely unaffected by the number of bins in the joint histogram. For other methods, performance steadily drops with increasing histogram size, due to the lack of samples. PVE performs the worst in terms of bias. Again, this is due to the bias of PVE towards lattice-alignment.

Similar conclusions may be drawn from the convergence results in Figure 5.5 and 5.6. The number of bins has less influence on NP-windowing than any other method. For STD, PVE and IPZ there is similarly good performance at low initial offsets from the minimum, whatever the histogram bin-size. Likewise, at distant initial offsets performance is poor in all cases. At intermediate distances however, there is a range of performances, and it is at intermediate distances where success or failure are similarly likely, that the influence of an algorithm's parameters is clearest.

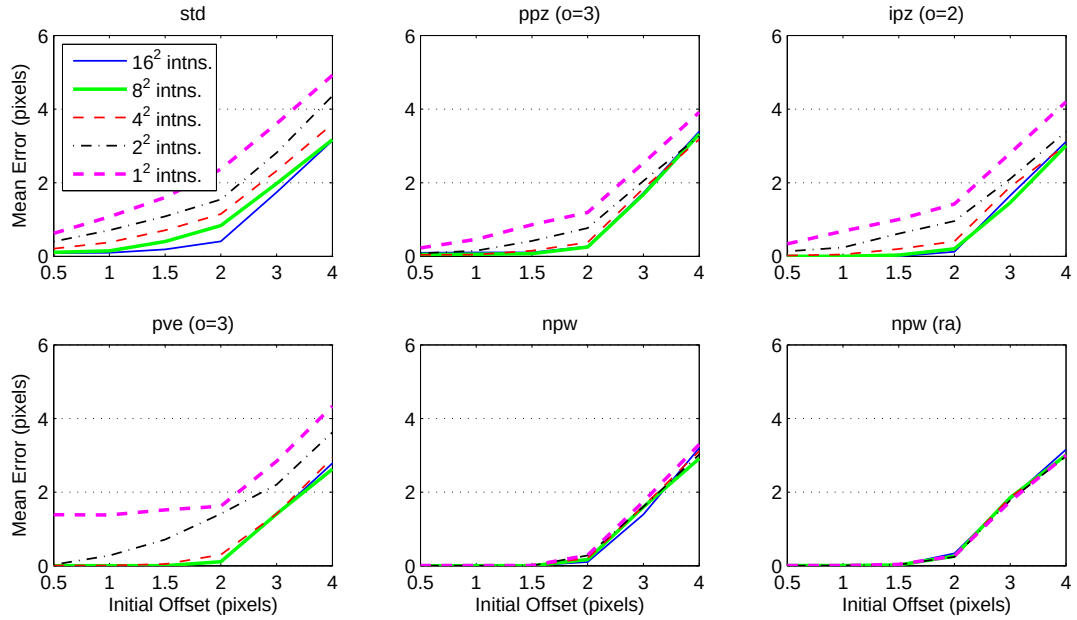
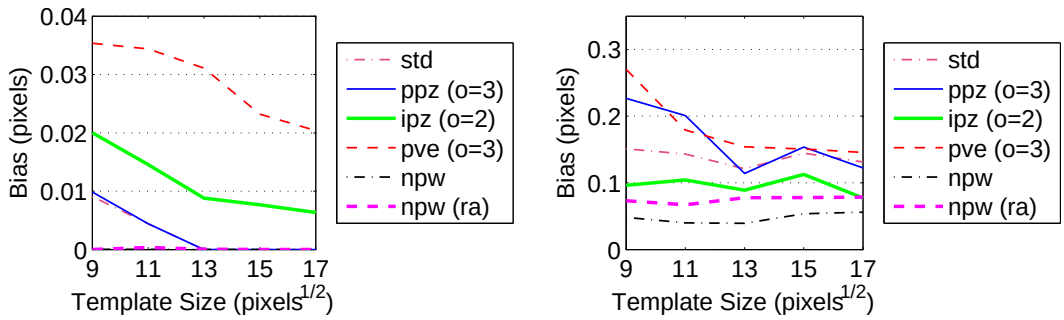


Figure 5.6: Plots showing convergence for varying *histogram bin-size* when using lattice-offset templates

PVE was the second best performer in terms of convergence, because of a larger basin size arising from extended spatial support. Apart from PVE, the basin of convergence is neither affected by the number of histogram bins, nor by the sampling method used.

The results for bias while varying template size are given for both lattice-aligned and lattice-offset templates in Figure 5.7. The convergence results, when varying template size, are plotted in Figure 5.8 and 5.9 for lattice-aligned and lattice-offset templates respectively. In Figure 5.7, the bias generally decreases as the template size (and number of available samples) increases, because the statistics represent the data better and tend to dominate over the local blurring effects of the kernel (if one is used). This is particularly evident for PVE. However, the decrease in bias is negligible for NP-windowing, since it makes maximal use of available information, and additional pixels add mainly redundant data.



(a) (b)

Figure 5.7: Bias when varying *(a)* template size and *(b)* histogram bin-size, for lattice-offset templates.

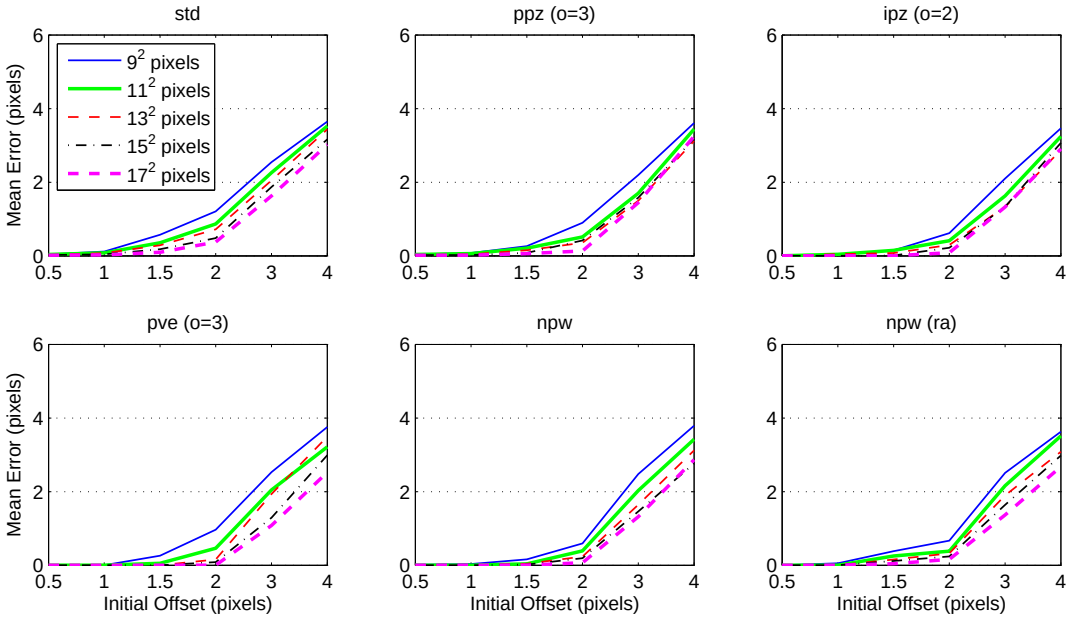


Figure 5.8: Plots showing convergence for varying *template sizes* when using lattice-aligned templates

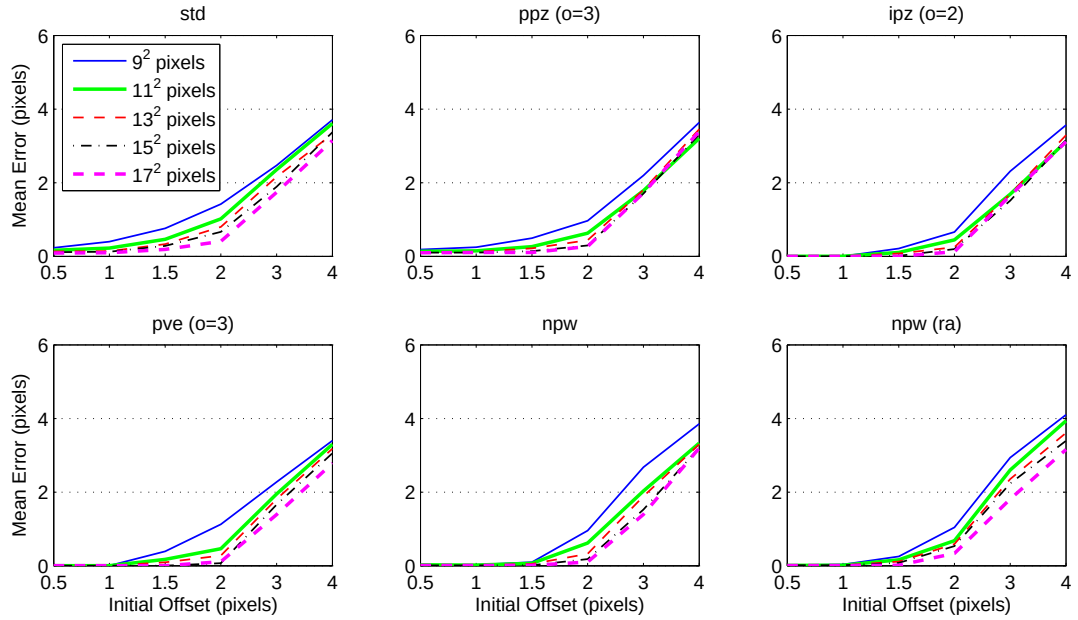


Figure 5.9: Plots showing convergence for varying *template sizes* when using lattice-offset templates

In Figure 5.9a, the point of inflection for convergence tests shifts rightwards as the template size increases. This indicates that additional data can increase the size of the basin of convergence despite adding only redundant information. Parzen windowing is least affected by the number of samples (lowest range at intermediate values), and PVE gives best performance for the 17x17 template due to its wide spatial support. However, NP-windowing gives a better mean performance than either method and outperforms other methods in the most difficult case (9x9 template).

Interestingly, the use of a resolution aware approach to NP-windowing degraded performance, although it did compare favourably to other MI estimation methods. This occurred because the position of the locally regular lattice points is dependent on the warp parameters resulting in beats and hence glitches. The induced glitches reduce the convergence rate and increase bias relative to the

standard NP Windows implementation.

5.5 Conclusions

In this chapter the NP-windowing method for obtaining the joint distribution of a pair of images was introduced. Unlike existing methods the proposed approach neither assumes sample independence nor identical distributions. Moreover, the method does not require arbitrary parameters like kernel size to be chosen. NP-windowing effectively eliminates the error caused by having a finite number of samples. NP-windowing is a general case for both STD and for first order PVE.

NP-windowing for half-bilinear and bilinear interpolation schemes was developed theoretically. Because of its simplicity, relative accuracy, and speed, the half bilinear approach was implemented, using standard polygon rendering routines to generate joint histograms.

The histograms obtained using NP-windowing were used in registration applications to maximise the Mutual Information between a reference image and a template with respect to some warp parameters. The registration accuracy was compared to existing state of the art registration methods using MI. Performance was measured in terms of bias and convergence using a set of eight images at a range of initial starting points. In all, more than half a million tests were performed.

In general, NP-windowing had less bias than any of the methods it was compared to (more than 40% less than its nearest competitor in some cases). In addition, the number of histogram bins and sample size did not significantly affect NP-windowing, implying that it makes maximal use of available data. NP-windowing typically demonstrated the best convergence properties, although the increased

spatial support of 3rd order PVE gave advantages when converging from initial positions distances far from the function minimum. However, NP-windowing can tolerate decreased sample sizes better than PVE. The radius of the basin of convergence increases with sample size, but is unaffected by histogram bin-size. Resolution awareness did not improve performance, since the position the resolution aware sampling points is coupled to the position of the template and reference pixel-lattices. The result of this is beats and hence glitches, which generate local minima, reducing the convergence rate and increasing in bias.

NP-windowing is recommended for applications where bin-size should be a non-critical choice and where the number of samples is limited or unknown. Moreover, in domains where the bias (shift in true global minimum) must be kept to a minimum, it can be concluded that NP-windowing should be used.

The effects of sampling methods on the Mutual Information and artefacts in its function surface were discussed in the Chapters 3 and 4, and increasing the sampling rate was shown to provide the best results. In this Chapter, the proposal of sampling at a higher resolution was taken to its logical conclusion by sampling at infinite resolution. A return to the general registration problem is made in Chapter 6: where a method is proposed to obtain substantial improvements in speed.

Chapter 6

MILK: Mutual Information in a Lucas-Kanade Framework

The previous three Chapters considered various methods for sampling images to obtain accurate values for Mutual Information with the aim of reducing the effects of artefacts in the function surface. In this Chapter, the more practical issue of formulating the MI optimisation function in a Lucas Kanade framework is considered, with the aim of reducing the computation required for registration. In particular, an inverse compositional formulation for aligning a template and a reference image using mutual information is proposed.

Lucas and Kanade made one of the earliest practical attempts to efficiently align a template image to a reference image [45], using the Sum of Squared Difference similarity metric. Processing was limited by using a Newton-Raphson method to traverse the space of warp parameters. In Newton-Raphson optimisation, iterative parameter updates to alignment parameters are obtained by multiplying the Jacobian by an inverse Hessian. Lucas and Kanade mainly considered translations, but they demonstrated that any linear transformation could be used.

Later research considered more complex transforms and attempted to reformulate the similarity metric allowing pre-computation of some terms. In particular, Hager and Belhumeur [28] proposed inverting the roles of the reference and template at a strategic point in the derivation, and Shum and Szeliski [73] constructed the warp as a composition of two nested warps. In a general treatise on Lucas-Kanade (LK) techniques [3], Baker and Matthews combined these ideas to formulate the inverse-compositional method.

An approach similar to that used by Baker [3] is used, in this Chapter, to obtain a constant Hessian for the MI similarity metric. To maintain the convention established by Baker, the derived formulation is also referred to as an inverse compositional formulation. This uses two techniques: first, the alignment function is composed as a base warp and a warp variation; and second, the roles of template and reference image are inverted or exchanged. This is difficult in the case of MI because the template and reference values are not separable into two terms. But with some limited assumptions of constancy, a speed improvement is obtainable.

The remainder of the chapter is arranged as follows. After providing some background in Section 6.1 the inverse compositional formulation of MI is presented in Section 6.2. The derivation obtained is compared to existing methods in terms of convergence and speed in Section 6.3 before the conclusions are given in Section 6.4.

6.1 Background

Many optimisation algorithms exist, but LK methods use a subset of these: the so called Newton-type methods *i.e.* methods, which assume locally parabolic

Name	Update
(Full) Newton Descent	$\mathbf{v}^{(k+1)} \leftarrow \mathbf{v}^{(k)} - H^{-1}G$
Quasi-Newton Descent	$\mathbf{v}^{(k+1)} \leftarrow \mathbf{v}^{(k)} - \tilde{H}^{-1}G$
Steepest Descent	$\mathbf{v}^{(k+1)} \leftarrow \mathbf{v}^{(k)} - \lambda G$
Levenberg Marquardt	$\mathbf{v}^{(k+1)} \leftarrow \mathbf{v}^{(k)} - ((\mathbf{1} + \lambda\mathbf{I})\tilde{\mathbf{H}})^{-\mathbf{1}}\mathbf{G}$

Table 6.1: Updates for four Newton-type optimisation Methods. (**1** is a matrix of ones). Although not explicitly indicated, several λ values may be tested for the Levenberg-Marquardt and Steepest Descent methods.

topology and "jump" to the minimum using gradient information to update the warp parameters, $\mathbf{v}$, as follows: $\mathbf{v}^{(k+1)} \leftarrow \mathbf{v}^{(k)} - H^{-1}(\mathbf{v}^{(k)})G(\mathbf{v}^{(k)})$, where H is the Hessian of an image similarity function, d, and G is the Jacobian of d. Newton methods should be contrasted with methods that choose a direction, bracket the minimum, and minimise along the line using Brent's algorithm [10], *e.g.* Powell's Method or Variable Metric Methods [68]. Bracketing methods are more stable than Newton methods, but somewhat slower, since more function evaluations are performed. Speed is often more important than accuracy in image alignment where minima are numerous and closely spaced, because stable methods will often fail despite their robustness.

Generally LK type methods apply Quasi-Newton optimisation, *i.e.* an approximate Hessian, $\tilde{H}$, is used. In general, Newton and Quasi-Newton only perform well when near to the minimum. Steepest Descent methods, which ignore local curvature and instead multiply G by a scalar *step-size* value, λ, perform better at points further from the minimum. The Levenberg-Marquardt [50] method combines these two methods for optimal performance. A summary of these methods is given in Table 6.1, which follow the conventions of the registration framework established in Section 2.2.

6.1.1 Lucas-Kanade Framework

The Lucas-Kanade (LK) framework uses the sum of squared differences metric defined for a warp parameter, $\mathbf{v}$, and a change in the warp $\Delta\mathbf{v}$:

$$d_{ssd}(\mathbf{v}+\Delta\mathbf{v}) = \sum_{\mathbf{x}}[f_r(\mathbf{w}(\mathbf{x},\mathbf{v}+\Delta\mathbf{v})) - f_t(\mathbf{x})]^2 \tag{6.1}$$

Following the derivation of Lucas and Kanade [45], a first order Taylor expansion is applied to the function within the brackets:

$$d_{ssd}(\mathbf{v}+\Delta\mathbf{v}) = \sum_{\mathbf{x}}[f_r(\mathbf{w}(\mathbf{x},\mathbf{v})) + \nabla f_r \frac{\partial \mathbf{w}}{\partial \mathbf{v}}\Delta\mathbf{v} - f_t(\mathbf{x})]^2 \tag{6.2}$$

where ∇f_r is the gradient of the image f_r with respect to its coordinates. A partial derivative with respect to $\Delta\mathbf{v}$ is then obtained:

$$\frac{\partial d_{ssd}}{\partial \Delta\mathbf{v}} = 2\sum_{\mathbf{x}}[\nabla f_r \frac{\partial \mathbf{w}}{\partial \mathbf{v}}]^T[f_r(\mathbf{w}(\mathbf{x})) + \nabla f_r \frac{\partial \mathbf{w}}{\partial \mathbf{v}}\Delta\mathbf{v} - f_t(\mathbf{x})] \tag{6.3}$$

Assuming a locally parabolic shape and setting the gradient to zero gives a closed form solution for updating $\mathbf{v}$ of the form: $\Delta\mathbf{v} = \tilde{H}^{-1}G$, where:

$$G = 2\sum_{\mathbf{x}}\left(\nabla f_r \frac{\partial \mathbf{w}}{\partial \mathbf{v}}\right)^T (f_t(\mathbf{x}) - f_r(\mathbf{w}(\mathbf{x})))^2 \tag{6.4}$$

$$\tilde{H} = 2\sum_{\mathbf{x}}\left(\nabla f_r \frac{\partial \mathbf{w}}{\partial \mathbf{v}}\right)^T \left(\nabla f_r \frac{\partial \mathbf{w}}{\partial \mathbf{v}}\right) \tag{6.5}$$

Of course, a truly parabolic surface seldom occurs. Instead the warp parameter is iteratively computed and updated until the variation in parameters or function values becomes sufficiently small. The computational cost of each update is $O(N_{\mathbf{x}}N_{\mathbf{v}})$ for G and $O(N_{\mathbf{x}}N_{\mathbf{v}}^2)$ for H, where $N_{\mathbf{x}}$ is the number of pixels and $N_{\mathbf{v}}$ is the number of warp components.

The Hessian is denoted with a tilde because of an early hidden approximation that is not mentioned mentioned in the literature. The approximation is made

in the Taylor expansion, which neglects some of the second order information. A full second order expansion applied to the the entire d_{ssd} function yields the full Hessian:

$$H = 2\sum_{\mathbf{x}}[\nabla f_r \frac{\partial \mathbf{w}}{\partial \mathbf{x}}]^T[\nabla f_r \frac{\partial \mathbf{w}}{\partial \mathbf{x}}] + (f_r - f_t)[\frac{\partial \mathbf{w}}{\partial \mathbf{v}}]^T(\nabla \cdot \nabla f_r)[\frac{\partial \mathbf{w}}{\partial \mathbf{v}}] + \nabla f_r[\frac{\partial \mathbf{w}}{\partial \mathbf{v}}]^T[\frac{\partial \mathbf{w}}{\partial \mathbf{v}}]) \tag{6.6}$$

In a full Newton derivation (6.6) would replace (6.4). Apart from the second term in H being computationally expensive to compute $O(N_{\mathbf{x}}N_{\mathbf{v}}^2)$ it is often marginal compared to the first term, especially near the minimum and has little effect on convergence.

6.1.2 The Inverse-Compositional Method

Baker and Matthews presented a reformulation of the SSD distance function and update method called the inverse compositional method in [3]. The warp function was re-composed as a function of two warps $\mathbf{w}(\mathbf{x}, \mathbf{v})$ and $\mathbf{w}(\mathbf{x}, \Delta\mathbf{v})$ with the roles of f_r and f_t inverted:

$$d_{ssd}(\mathbf{v}, \Delta\mathbf{v}) = \sum_{\mathbf{x}} \left(f_t(\mathbf{w}(\mathbf{x}, \Delta\mathbf{v})) - f_r(\mathbf{w}(\mathbf{x}, \mathbf{v}))\right)^2 \tag{6.7}$$

Following the steps in Section 6.1.1 using this formulation yields the following approximation of the Hessian: $\tilde{H} = (\nabla f_t \frac{\partial \mathbf{w}}{\partial \mathbf{v}})^T(\nabla f_t \frac{\partial \mathbf{w}}{\partial \mathbf{v}})$. This depends solely on the template and is therefore constant with respect to $\mathbf{v}$. In other words the Hessian may be precomputed, decreasing the overall complexity of each iterative update to $\mathbf{v}$ from $O(N_{\mathbf{x}}N_{\mathbf{v}}^2)$to $O(N_{\mathbf{x}}N_{\mathbf{v}})$.

6.2 Mutual Information

For reference the MI function, its first and second derivative are restated from Section 3.2:

$$d_{mi} = \sum_{r,t} p_{rt}(r,t,\mathbf{v}) \log\left(\frac{p_{rt}(r,t,\mathbf{v})}{p_r(r,\mathbf{v})p_t(t,\mathbf{v})}\right) \tag{6.8a}$$

$$G_{mi} = \frac{\partial d_{MI}}{\partial \mathbf{v}} = \sum_{r,t} \frac{\partial p_{rt}}{\partial \mathbf{v}} \log\left(\frac{ep_{rt}(\mathbf{v})}{p_r(\mathbf{v})}\right) \tag{6.8b}$$

$$H_{mi} = \frac{\partial^2 d_{MI}}{\partial \mathbf{v}^2} = \sum_{r,t} \frac{\partial p_{rt}}{\partial \mathbf{v}}^T \frac{\partial p_{rt}}{\partial \mathbf{v}} \left(\frac{1}{p_{rt}} - \frac{1}{p_r}\right) + \frac{\partial^2 p_{rt}}{\partial \mathbf{v}^2} \log\left(\frac{ep_{rt}}{p_r}\right) \tag{6.8c}$$

In this Chapter, only the standard sampled version of Mutual Information was considered, because the primary focus is on speed, and standard sampling is the fastest method. Moreover, the methods described in this Chapter are easily adapted to the other MI methods, and the simplicity of standard sampling lends clarity to the following derivations.

For the standard sampled implementation, approximate derivatives of ψ were used to obtain $\frac{\partial p_{rt}}{\partial \mathbf{v}}$ and $\frac{\partial^2 p_{rt}}{\partial \mathbf{v}^2}$ by respectively treating ψ' as the convolution window $[-1\ +1]$ and ψ'' as a convolution window $[-1\ +2\ -1]$. In Chapter 3 the last term in (6.8c) was neglected because it is expensive to obtain and has little effect upon convergence once the solution is near the local minimum. This is the analogue of neglecting certain second order terms for SSD in (6.4).

6.2.1 Inverse-Compositional MILK

The inverse compositional derivation for MI can now be obtained in the same manner as for SSD, by splitting the warp into a function of two parameters:

$$d_{MI}(\mathbf{v}, \Delta\mathbf{v}) = \sum_{r,t} p_{rt}(\mathbf{v}, \Delta\mathbf{v}) \log\left(\frac{p_{rt}(\mathbf{v}, \Delta\mathbf{v})}{p_r(\mathbf{v})p_t}\right) \tag{6.9}$$

Hereafter to save space the function parameters are not shown, *i.e.* $p_t = p_t(\mathbf{0})$, $p_r = p_r(\mathbf{v})$ and $p_{rt} = p_{rt}(\mathbf{v}, \mathbf{0})$. Using the same approach as Section 6.1.2 for MI, the following gradient and Hessian functions are obtained.

$$G = \sum_{r,t} -\frac{p_{rt}}{p_t}\frac{\partial p_t}{\partial \Delta\mathbf{v}} + \log\left(\frac{e p_{rt}}{p_r p_t}\right)\frac{\partial p_{rt}}{\partial \Delta\mathbf{v}} \tag{6.10}$$

where the first term falls away because the p_{rt} factor may be separated to form $\sum_t \frac{p'_t}{p_t}\sum_r p_{rt} = \sum_t p'_t = 0$. In the second term, the p_r factor is also removed, using the reasoning of Section 3.2:

$$H = \sum_{r,t} \frac{p_{rt}}{p_t^2}\frac{\partial p_t}{\partial \Delta\mathbf{v}}^T\frac{\partial p_t}{\partial \Delta\mathbf{v}} - \frac{p_{rt}}{p_t}\frac{\partial^2 p_t}{\partial \Delta\mathbf{v}^2} - \frac{2}{p_t}\frac{\partial p_t}{\partial \Delta\mathbf{v}}\frac{\partial p_{rt}}{\partial \Delta\mathbf{v}} + \frac{1}{p_{rt}}\frac{\partial p_{rt}}{\partial \Delta\mathbf{v}}^T\frac{\partial p_{rt}}{\partial \Delta\mathbf{v}} + \log\left(\frac{e p_{rt}}{p_r p_t}\right)\frac{\partial^2 p_{rt}}{\partial \Delta\mathbf{v}^2} \tag{6.11}$$

In the first term of (6.11), only one term is dependent on r so the summation over r may be separated out to form: $\sum_t \frac{\partial p_t}{\partial \Delta\mathbf{v}}^T \frac{\partial p_t}{\partial \Delta\mathbf{v}} \frac{1}{p_t^2}\sum_r p_{rt} = \frac{1}{p_t}\sum_t \frac{\partial p_t}{\partial \Delta\mathbf{v}}^T \frac{\partial p_t}{\partial \Delta\mathbf{v}}$. Similarly the third term also has only one factor dependent on r, so it simplifies to $-\frac{2}{p_t}\sum_t \frac{\partial p_t}{\partial \Delta\mathbf{v}}^T \frac{\partial p_t}{\partial \mathbf{v}}$. Hence (6.11) simplifies to:

$$H = \sum_{r,t} -\frac{1}{p_t}\left(\frac{\partial p_t}{\partial \Delta\mathbf{v}}^T\frac{\partial p_t}{\partial \Delta\mathbf{v}} + \frac{\partial^2 p_t}{\partial \Delta\mathbf{v}^2}\right) + \frac{\partial p_{rt}}{\partial \Delta\mathbf{v}}^T\frac{\partial p_{rt}}{\partial \Delta\mathbf{v}} + \log\left(\frac{e p_{rt}}{p_r p_t(\Delta\mathbf{v})}\right)\frac{\partial^2 p_{rt}}{\partial \Delta\mathbf{v}^2} \tag{6.12}$$

In (6.12) there are three terms, the first of which is first order and independent of f_r. The second term is first order but dependent on f_r and the third is second order. The second order information can be neglected as was done for SSD, since again it is marginal at positions close to the minimum. The second term presents more of a problem, since it is also $O(N_{rt}N_{\mathbf{v}}^2)$ to compute, where N_{rt} is the number of joint histogram bins. However if it is assumed to be approximately constant relative to $\Delta\mathbf{v}$, H in (6.12) may be entirely pre-computed. This assumption still gives good results, as will be shown.

6.3 Experiments

The inverse compositional formulation for optimising MI was compared to the forward additive formulation using a number of tests for convergence error. The number of iterations and time to reach convergence was also recorded. In addition, comparisons were made with the full Newton (*i.e.* un-approximated Hessian) for both formulations.

The test setup described in Section 3.4 was used to measure the performance of the registration algorithms discussed in this Chapter, with some slight differences. For testing, the optimisation algorithm was initialised with 100 random angles at offsets ranging from 0.25 to 5 pixels. Levenberg-Marquardt was used to optimise an *affine* warp. The affine parameters were randomly chosen with a normal distribution and a standard deviation of 0.03. Bi-linear interpolation was used in each case.

Levenberg-Marquardt was chosen, because it uses the Hessian directly, unlike methods like Powell's method or the Conjugate gradients method, which build up a Hessian from a series of function evaluations. Hence, the latter two methods do not benefit from the MILK techniques discussed in this Chapter. However, recent work has shown that an inverse compositional formulation can be applied to the Goldfarb-Broyden-Fletcher-Shannon optimisation method [11]. An affine warp was used because it has sufficient parameters for Levenberg-Marquardt to have a clear speed advantage over methods that construct a Hessian over multiple iterations. For lower parameter warps this advantage would be less clear.

Two sets of analyses were performed. Firstly, performance was measured as the number of bins in the joint histogram was varied from 4^2 to 64^2 bins in multiples of four. Secondly, performance was measured as the template was varied in size from 9^2 to 17^2 pixels. In all 104,000 individual tests were performed.

Table 6.2: Mean costs of evaluating terms in (6.8a) and (6.9) in ms per measurement as a function of template size and no. of histogram bins. Pre-processed terms are bold.

Template Size	Forwards Additive				Inverse Compositional			
	d_{MI}	G	$\tilde{H}$	H	d_{MI}	G	$\tilde{\mathbf{H}}$	H
9^2	1.732	3.995	0.046	8.349	1.773	0.311	1.745	3.899
11^2	1.737	4.064	0.099	9.099	1.750	0.403	1.851	1.342
13^2	1.854	3.969	0.048	8.698	1.774	0.408	1.940	1.492
15^2	1.787	4.081	0.054	9.021	1.824	0.426	2.129	1.594
17^2	1.807	4.105	0.128	9.160	1.795	0.464	2.430	1.748
No. Bins								
	d_{MI}	G	$\tilde{H}$	H	d_{MI}	G	$\tilde{\mathbf{H}}$	H
4^2	1.886	3.894	0.057	8.035	1.838	0.374	1.774	0.934
8^2	1.885	4.005	0.039	8.759	1.847	0.425	1.863	1.542
16^2	1.882	4.160	0.222	10.655	1.859	0.575	2.218	3.003
32^2	1.905	4.762	0.591	16.757	1.919	0.946	3.117	7.404
64^2	2.157	6.513	1.320	38.153	2.142	2.254	6.299	22.010

The mean time to evaluate each term in (3.1) and (6.9) is given in Table 6.2. The cost is given as a function of both increasing template size and decreasing histogram bin-size. The processing overhead confounds the figures slightly, but it is clear that the cost increases with template size and decreases with the bin-size. In addition the cost per iteration is much lower for the inverse compositional approach, since only d and G need to be evaluated.

In Figure 6.1, the mean number of function evaluations is shown. In the Levenberg-Marquardt method G and $\tilde{H}$ (dark / blue) do not need to be evaluated every time d (shaded / green) is measured so these evaluations are separated. The number of evaluations follows a bell curve when plotted against increasing offset distance. A bell shape occurs because more function evaluations are required the further the initial position is from the maximum *when within the basin of convergence*. When outside this region the algorithm often terminates early upon getting stuck in a local maximum, with a low radius basin of convergence. The full-Newton methods generally requires less iterations due to their more accurate Hessians. Smaller template sizes generally implied fewer iterations: due to a higher frequency of failure (see Figure 6.2). The inverse compositional method also generally had fewer iterations, also due to more frequent failure and early termination.

The mean convergence error for five different template sizes is shown in Figure 6.2. As expected, the larger the template (*i.e.* the more data) the lower the mean error. The mean convergence error for varying numbers of histogram bins is shown in Figure 6.3. As the number of bins increases, the error decreases as the function is more discriminative, until 32^2 bins is reached. For 64^2 bins the error increases again, since the histogram is too sparsely populated. This results in the MI function becoming more unstable and convergence more difficult.

In all cases, performance in Figure 6.2 and 6.3 is similarly good for low offsets. Note that the mean error never reaches zero. This is due to the large number of degrees of freedom and the fact that the MI cost function surface is noisy. Likewise, performance is similarly bad for large offsets. At more intermediate offsets, however, there is a range of performance. It is at these distances which are near to the edge of the basin of convergence where the accuracy of G and $\tilde{H}$ can critically result in successful convergence or failure.

The full-Newton approach gave very similar results to the quasi-Newton methods. From this it can be concluded that the additional second order information is not worth computing. It only decreases the number of evaluations slightly (See Figure 6.1) and the loss in Hessian accuracy is not sufficient to make failure more likely.

The inverse compositional approach was generally less accurate than the forwards additive method with the greatest differences at intermediate offsets. On average, the inverse compositional approach had 3.6% more error ($\sigma = 8.3\%$). However the inverse compositional method was on average 17.0% cheaper to evaluate ($\sigma = 8.1\%$).

In addition to the tests above, a tracking algorithm was implemented using the strategic update approach of Matthews and Baker [53]. In the sequence, the tracking performance of forwards additive MI and inverse compositional MI were compared on a video sequence. The results are shown in Figure 6.4. As shown the inverse compositional approach performed just as well as forwards additive. SSD is also shown for comparison. SSD also tracked well, but a large occlusion by the hand pulled the tracker off target.

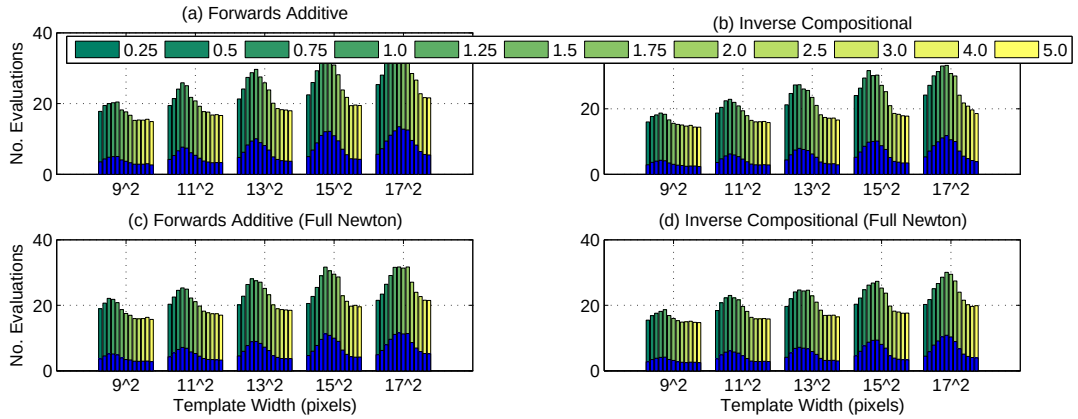


Figure 6.1: Mean no. evaluations for d_{mi} and d_{ssd} (shaded/green) in the forwards additive and inverse compositional formulations. The number of evaluations of G & $\tilde{H}$ are shown as well (dark/blue).

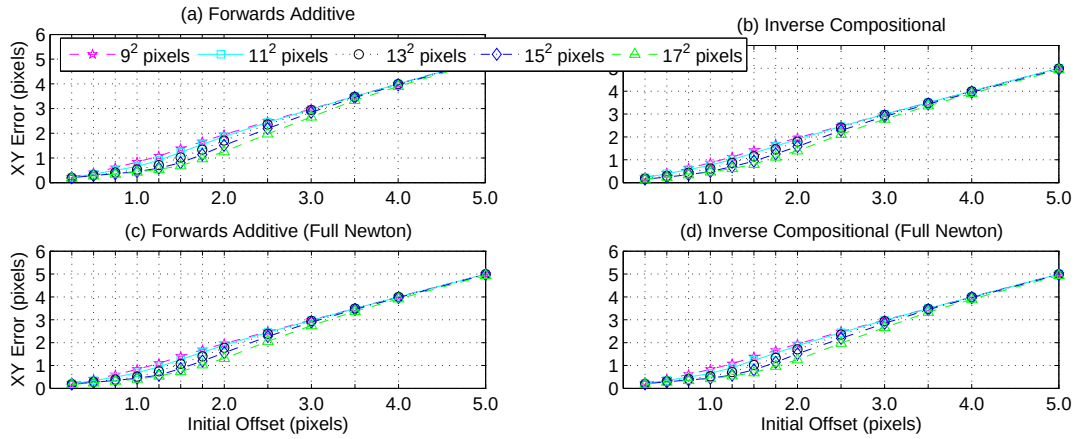


Figure 6.2: Mean convergence error when varying the template size for MI and SSD in forwards additive and inverse compositional formulations

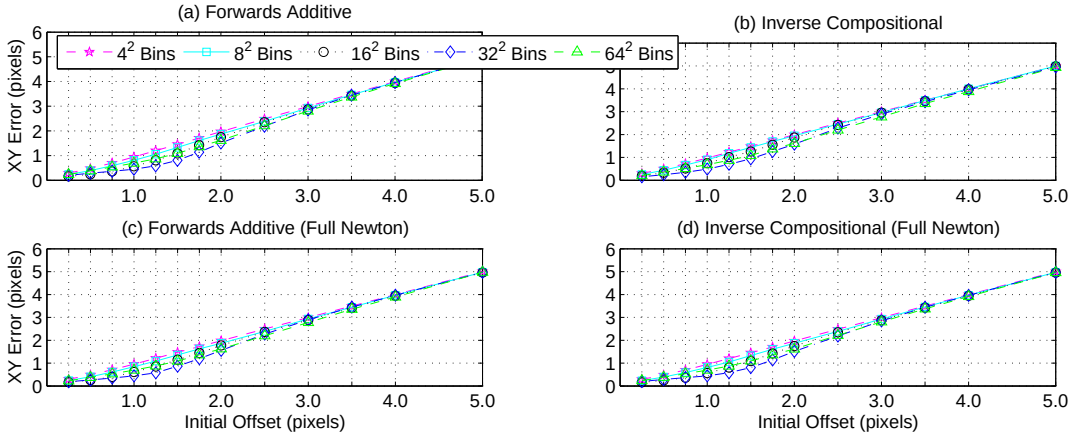


Figure 6.3: Mean convergence error when varying the histogram bin-size for MI and SSD in forwards additive and inverse compositional formulations

6.4 Conclusion

In this chapter, an inverse compositional formulation for Mutual Information was introduced. This reformulates the Mutual Information function so that the approximated second derivative is dependent primarily on the template image values and therefore constant. The cost of evaluation was reduced by 17% on average. On average the inverse compositional algorithm had a 3% larger mean convergence error. Considering the performance gains this is only a slight disadvantage since more processing cycles are available for improving robustness. With more optimised code, further performance gains are possible.

Using some of the techniques presented over the last four Chapters, in Chapter 7 Mutual Information is applied in a Lucas-Kanade framework for tracking objects over entire video sequences.

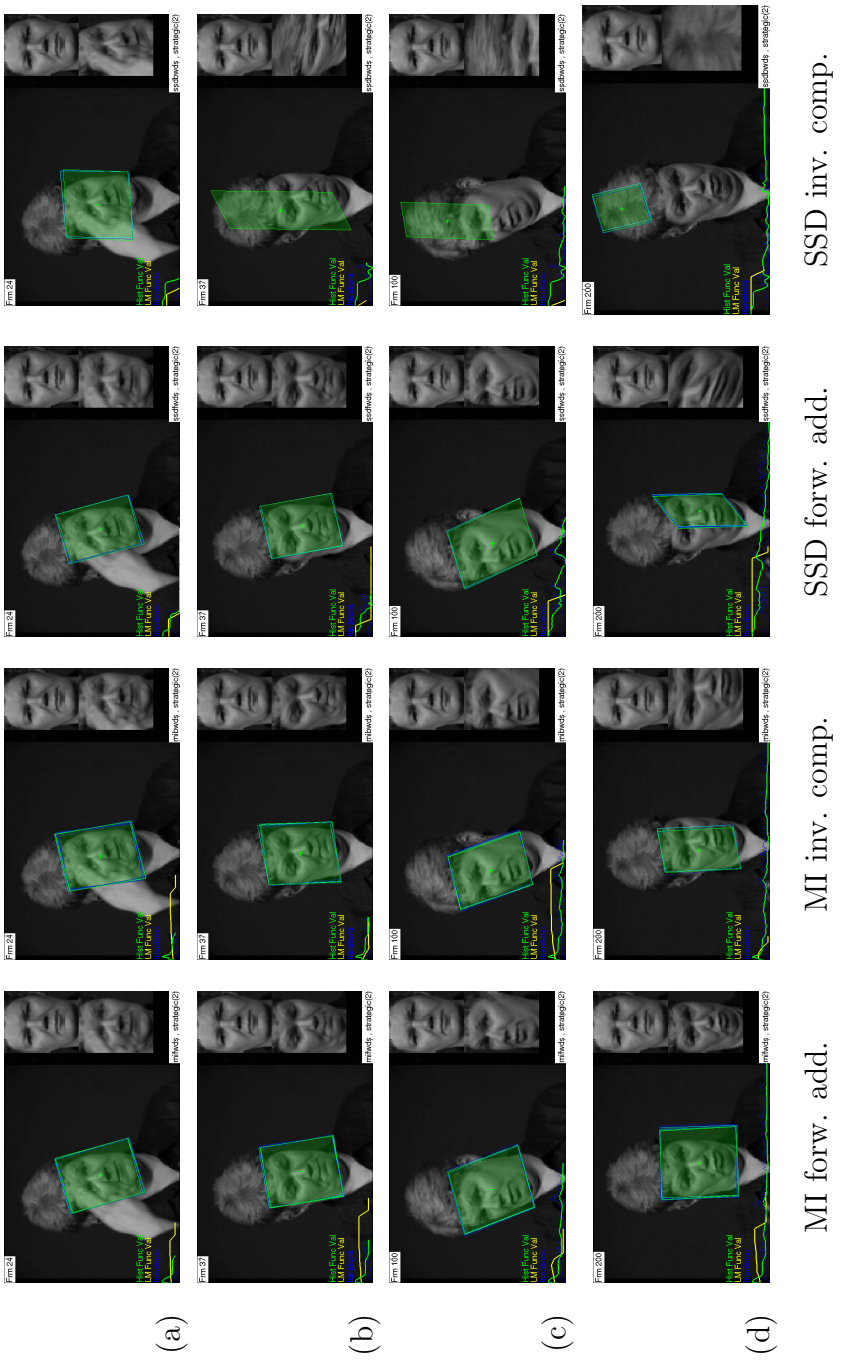


Figure 6.4: Tracking a face over a video sequence using using Forwards Additive and Inverse Compositional MI and SSD at (a) Frame 24 (b) Frame 37 (c) Frame 100 (d) Frame 200. The MI trackers perform similarly well despite the differences in implementation. SSD generally tracks well, but failure is induced by an early occlusion.

Chapter 7

Simultaneous Modelling and Tracking

In this chapter the Mutual Information function is used in a Lucas Kanade framework to track objects through a motion sequence. The aim of this work is to track non-rigid objects using appearance in real-time without a pre-learned model.

Lucas-Kanade (LK) type alignment [45] has often proven to be effective despite using only a very simple model (a single template) to track. It is natural to extend feature localisation, by image registration, to tracking a feature over a motion sequence. The last known location for the feature can be used to initialise the next registration as long as the motion is sufficiently slow, *i.e.* the temporal sampling rate is sufficiently high. This leads to a further problem, how and when to update the template. If the template is never updated, so-called *no-update* tracking, tracking will only work while the template closely represents the current appearance of the feature. This assumption is generally safe for several frames after the one from which the feature was extracted. Eventually however, the template does not represent the feature sufficiently well and a mismatch failure

ensues, and the tracker begins to track some other region that resembles the template more than the actual feature. Often recovery from mismatch is not possible. One alternative to this, is to update the template after every frame, *i.e.* keeping the representation as recent as possible. However, sub-pixel errors inherent to each match are also stored in each update. These errors gradually accumulate and the template drifts off the original feature.

Two recent approaches to overcome the problem of drift and mismatch, include those by Matthews *et al.* [53], and Kaneko and Hori [40]. Matthews *strategic update* approach is a simple but effective extension of the naive update algorithm, where the first template from the frame is retained and used to correct location errors made by the updated template. If the size of the correction is too large, the algorithm acts conservatively by not updating the template from the current frame. Kaneko on the other hand, makes a trade-off between accumulated drift error and mismatch error. In Kaneko's algorithm, the template is updated just before a mismatch error occurs, thereby delaying the onset of drift. Each of these errors is estimated from the boundary of the maximum error for possible templates.

The drawback to Matthews approach is that the appearance of the feature could change sufficiently that the correction is always larger than the threshold. This will result in the update being repeatedly blocked, and possible failure due to mismatch. Kaneko's approach attempts to minimise the number of updates, but eventually sub-pixel errors accumulate sufficiently for drift to occur. Both of these methods also rely on the first template being a good representation of the feature.

Consider the appearance of an object as it varies in pose as a manifold in a multi-dimensional appearance-space. All of the above methods rely on representing

the feature as one or two points on the manifold (templates). As the feature pose varies, the tracking algorithms attempt to shift the point on the manifold quickly enough for successful tracking. This work uses the philosophy that a successful tracking algorithm will try to model as large a region of the manifold as possible. This requires storing all the exemplars of appearance, rather than just one or two. Given sufficient exemplars from successful tracking, updates to the appearance-model could theoretically be reduced to zero, while the tracked feature would never be mismatched.

A suggested alternative to this approach, from the field of generalised predictive control [16], is to detect abrupt changes in appearance and use this as a cue to extract a new template. Both the new and old templates are used for a short-time thereafter, gradually ageing out the template that obtains comparably lower similarity measures each frame. This approach although requiring less memory, does not support multiple modalities and cannot take advantage of previously known appearances.

The remainder of the chapter is arranged as follows. After a background to tracking using one and two template models, in Section 7.1, the proposed Simultaneously modelling and Tracking (SMAT) approach is presented in Section 7.2. The extension of this method for use in modelling shape and hence multi-tier models is discussed in Section 7.3. Next, a description of the experimental setup is presented in Section 7.4, which introduces a new method for testing tracking algorithms.

Testing was performed on 11 challenging sequences with a mean length of 568 frames. The results of these tests are discussed in Section 7.5, before the Chapter is concluded in Section 7.6.

7.1 Tracking with one and two template models

For reference, the minimisation function used for registration is restated, with the addition of an index for frame number, m, for the reference image, f_r, and an index m' for a particular template, f_t:

$$\mathbf{v}_{opt} = \arg_{\mathbf{v}} \min d[f_r^{(m)}(\mathbf{w}(\mathbf{x}, \mathbf{v})), f_t^{(m',m)}(\mathbf{x})] \tag{7.1}$$

The index, m', used for f_t indicates that f_t is one template *chosen* from within a list of templates, for alignment in the current frame.

After alignment, a new exemplar may be extracted from each frame. Typically this is extracted from the region in f_r overlapped by f_t when warped by $\mathbf{v}_{opt}$. Each pixel in the new exemplar corresponds to the warped position of each pixel in f_t:

$$f_x^{(m)}(\mathbf{x}) = f_r^{(m)}(\mathbf{w}(\mathbf{x}, \mathbf{v}_{opt}^{(m)})) \quad | \quad \mathbf{x} \in \{ \text{ Defined pixels of } f_t\} \tag{7.2}$$

where $f_x^{(m)}$ is the exemplar extracted from frame m, and $\mathbf{v}_{opt}^{(m)}$ contains the parameters giving the image alignment of optimum match. The pixel positions in f_x will generally not line up with the pixels in f_r, so interpolation is required.

Perhaps the simplest model of appearance is the no-update method, which solely consists of the exemplar extracted from the first frame in the sequence:

$$f_t^{(m',m)}(\mathbf{x}) = f_x^{(1)}(\mathbf{x}) \qquad \forall m \in [1; N_m] \tag{7.3}$$

where N_m is the number of frames in the sequence. The *no-update* model, illustrated in Figure 7.1a, works only as long as the template closely resembles the feature being tracked. As mentioned, the resemblance is typically fleeting, and tracking fails due to a mismatch error, as shown in Figure 7.2.

One alternative is the *naive-update* method, where the template is updated after every frame:

$$f_t^{(m',m)}(\mathbf{x}) = f_x^{(m-1)}(\mathbf{x}) \qquad \forall m \in [1; N_m] \tag{7.4}$$

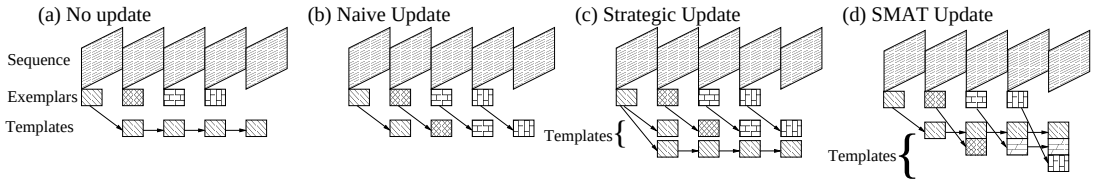


Figure 7.1: Building models using various update strategies. In each sub-figure, the upper row shows sequence frames, the middle row shows new exemplars extracted from each frame, and the bottom row shows the template used for tracking in the current frame. (a) The template is never updated. (b) The template is updated naively every frame. (c) Two templates are kept and one of them is selectively updated. (d) All the exemplars are collected into a single model.

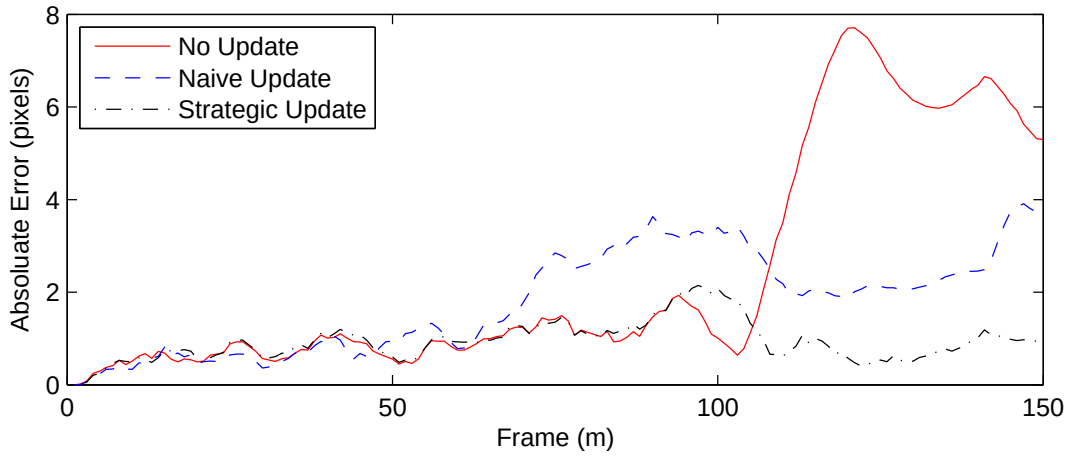


Figure 7.2: The patterns of accumulating error are shown for several update strategies. When no update is used, the error remains low until the model no longer represents the feature well and mismatch occurs. A spontaneous recovery occurs by chance towards the end of the sequence. For naive update, a steady increase in error (drift) occurs, due to accumulation of sub-pixel errors. Strategic update acts conservatively by blocking the update for overly large corrections, thereby avoiding drift or mismatch, but eventually mismatch failure will occur.

With naive-update, sub-pixel errors inherent to each match are stored during every update and gradually accumulate, resulting in the template drifting off the feature. The naive update method is illustrated in Figure 7.1b, and an example of drift is shown in Figure 7.2.

Matthew's *strategic-update* model combines the no-update and naive-update approaches in the following way. After obtaining an initial alignment with the updated template, $\mathbf{v}'_{opt}$, the registration operation is repeated with the exemplar from the first frame to correct for any errors, and to obtain $\mathbf{v}_{opt}$. If the correction of the second registration is too large, the algorithm acts conservatively by blocking an update from the current frame, m:

$$\mathbf{v}'_{opt(m)} = \arg \min_{\mathbf{v}|\mathbf{v}_{opt(m-1)}} d(f_t^{(m',m)}, f_r^{(m)}) \tag{7.5a}$$

$$\mathbf{v}_{opt(m)} = \arg \min_{\mathbf{v}|\mathbf{v}'_{opt(m)}} d(f_x^{(0)}, f_r^{(m)}) \tag{7.5b}$$

$$f_t^{(m',m+1)} = \begin{cases} f_t^{(m',m)} & |\mathbf{v}_{opt(m)} - \mathbf{v}'_{opt(m)}| > \tau_{\mathbf{v}} \\ f_x^{(m)} & \text{Otherwise} \end{cases} \tag{7.5c}$$

where $\tau_{\mathbf{v}}$ indicates the accepted threshold for the correction factor. The strategic update method is illustrated in Figure 7.1c, with an example of its performance in Figure 7.2.

Modified versions of the naive and strategic update methods, which use the principle of never using interpolated intensities, are also possible. This modification is referred to as the *zero-interpolation principle* (ZIP).

7.1.1 Zero Interpolation Principle

Because the intensities in $f_x^{(m)}$ are interpolated from $f_r^{(m)}$, there is a slight blurring and information loss. This is particularly apparent with warps of high shear or

low scale, where several $\mathbf{w}(\mathbf{x}, \mathbf{v}_{opt})$ points may occur within a single pixel-spacing in f_r. In these cases, the interpolated values of f_x do not approximate reality well and tracking failure becomes likely.

The zero interpolation principle aims to prevent the information loss associated with interpolation by extracting exemplars aligned with the pixel-lattice of f_r. These exemplars, denoted f_{zip}, are obtained from the rectangle closest to the matching region that is also aligned with the pixel-lattice of f_r, with associated parameters $\mathbf{v}_{zip}$. To prevent an accumulation of offsets between $\mathbf{v}_{opt}$ and $\mathbf{v}_{zip}$, the position of each exemplar relative to the first exemplar, $f_x^{(1)}$, is also stored and compensated for after registration in each frame. The set of ZIP exemplars has the benefit of being more general than normally extracted exemplars, because they are always slightly offset from the actual feature.

7.2 Simultaneous Modelling and Tracking

The inclusion of a second template in the strategic update model improves tracking results substantially. Simultaneous Modelling and Tracking (SMAT) extends this by storing the exemplars extracted from every frame, selecting templates from amongst the exemplars to locate the feature by solving (2.1), as illustrated in Figure 7.1d for four templates.

To limit the computational cost, the collection of exemplars is clustered on-the-fly, with each cluster, C_n, being represented by its median, μ_n. Only the cluster medians are ever used as templates, limiting m' in (2.1) to $m' \in [1; N_n]$, where N_n is the number of clusters. The median rather than the mean is used to avoid the pixel blurring inherent to the averaging of multiple intensity values. On-the-fly clustering is what differentiates this method from a particle filter. If particles

were used to select from amongst (many) stored templates, the large number of particles would require the warp space of the image to be randomly sampled as well. Numerous particles would be required to obtain statistics about which templates have a high probability of obtaining a match and which do not, making a implementation that operates in real-time unlikely. Also, registration via sampling of the warp space is often less precise than registration via optimisation. This negatively impacts upon the quality of the stored templates and hence tracking robustness.

A weighting, w_n, is also associated with each cluster, which represents the estimated *a priori* likelihood of the cluster resembling the current appearance of the feature being tracked. So the probability of the model matching the current feature appearance may be treated as a sum of likelihoods: $P = \sum_{n=1}^{N_n} w_n \frac{p(fg|d(f_x^{(m)},\mu_n))}{p(bg|d(f_x^{(m)},\mu_n))}$. The weights satisfy the constraint, $\sum_n w_n = 1$.

Two examples of such models are shown in Figure 7.3, with the exemplars represented as dots in a 2D analogue of appearance space, where the distance between two appearances is represented by d_{MI}. Using Figure 7.3, the construction of the model, P, is now described.

In each new frame, $f_r^{(m+1)}$, the tracked feature will change in appearance as conditions such as pose and lighting vary, with a corresponding shift in the feature's position in appearance space. In the examples of Figure 7.3a and b, the new positions are indicated by stars. Updates to single template models serve only to move the position of the model in appearance space, rather than properly describe the occupation of X. Ideally each new exemplar is added to the nearest cluster, otherwise clusters become overly inflated and are no longer specific to the feature. Moreover, some exemplars can be included that arise from an undetected tracking failure, which are not representative of the true feature. Either way mismatch

error can ensue, so membership to a given cluster must be determined.

Membership is determined in the following way. A collection of distances between μ_n and the matches with exemplars known to be *foreground* is maintained. In practice, this is obtained from the distance values, d, between the μ_n and each member of C_n. A normal distribution is fitted to these. Before each optimisation, a pre-search is normally used to locate the basin of convergence. The position giving the best score is used to initialise the optimisation algorithm. The scores of the remaining points are stored as examples of matches between μ_n and *background* exemplars. A normal distribution is fitted to the background distance scores. Pre-search scores are good at detecting ambiguous but incorrect matches, because many of these are on the edges of the basin of convergence. If the ratio between the foreground and background likelihoods is greater than one f_x is deemed to be a member of c_n:

$$\frac{p(fg|d(f_x^{(m)}, \mu_n), \eta_{n(fg)}(\mu, \sigma))}{p(bg|d(f_x^{(m)}, \mu_n), \eta_{n(bg)}(\mu, \sigma))} \tag{7.6}$$

This improves upon an earlier published incarnation of the model, which required a pre-selected bounding factor. The earlier thresholding method used a factor, τ', multiplied by the standard deviation of MI values between the median exemplar and the other members of the cluster. The factor sometimes needed to be tuned to specific sequences and gave unstable results, hence (7.6) is now used. The cluster boundaries produced by (7.6) are indicated by the ellipses in Figure 7.3.

Newly created clusters are generally less reliable than previously established ones, since they have fewer samples and may be the result of an earlier tracking failure. To model the effects of increasing relevance and reliability, the weights of each cluster are updated after after one of the clusters has achieved a successful match

as follows:

$$w_n \leftarrow \begin{cases} \frac{w_n+\alpha}{1+\alpha} & n = n_{match} \\ \frac{w_n}{1+\alpha} & n \neq n_{match} \end{cases} \tag{7.7}$$

where α is a learning parameter. This allows frequently successful clusters to dominate in the model, but allows obsolete clusters to be gradually removed. The learning parameter should be chosen with care. Too large an α-value allows potentially erroneous exemplars to quickly dominate within the model and marginalises older data that is more reliable. Too low an α-value prevents the model from reacting to new appearances and possible mismatch. In practice an α-value of 0.1 was found to work well.

To reduce computational expense further, a greedy approach is used when choosing between the clusters. Alignment, using (2.1), is attempted starting with the highest weighted cluster. If a match within the membership boundary, using (7.6), is obtained, as illustrated in Figure 7.3a, the new exemplar of appearance is added to the current cluster with appropriate updates to the weightings, median, η_{fg} and η_{bg}. Otherwise alignment is attempted using the next highest weighted cluster, and so on.

If none of the existing clusters achieves a successful match, as illustrated in Fig 7.3b, the most recently extracted exemplar, $f_x^{(m-1)}$, is used as a template: as this is the most likely to resemble the feature in its current form. A new cluster is then formed using this template, $f_x^{(m-1)}$, and the newly extracted region $f_x^{(m)}$, with an initial weight of zero. The number of clusters is limited to keep computational costs low. If this limit is exceeded, the cluster with the lowest weight is replaced. The algorithm is stated in Figure 7.2.

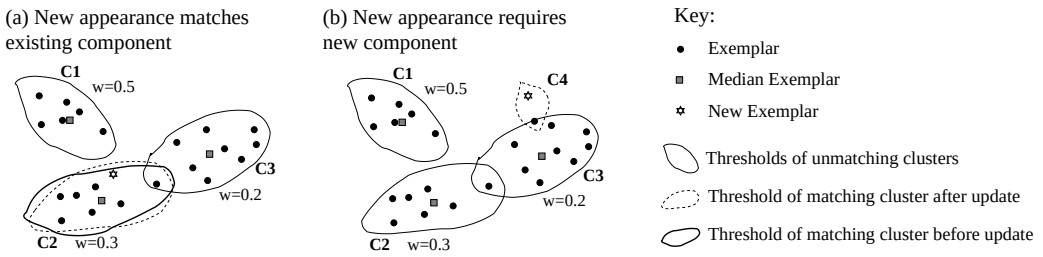


Figure 7.3: Diagram of Simultaneous Modelling and Tracking, showing 3 clusters of exemplars (dots) in appearance space. The tracked feature has a new appearance (star), which in (a) is within the bounds of Cluster 2 after minimisation, which is appropriately updated. In (b) no cluster achieves a successful match, so a new cluster is created, using the most recent template and the new exemplar obtained for the corresponding region in the current frame.

7.2.1 Zero Interpolation Principle & SMAT

Similarly to one or two template models, ZIP may be applied to SMAT. However this raises some issues with the clustering method, because for large warps the ZIP exemplar may be rotated away from the matching component. Hence the appearances will be highly dissimilar, and should not be clustered together. Two solutions to this problem were considered.

For warps that are too large, *e.g.* warps inducing a mean pixel shift of more than 0.1, a new cluster could automatically be generated. The disadvantage with this approach is that the limit is an arbitrary parameter to be chosen. Also, in sequences of agile objects, literally hundreds of clusters will be generated, most of which are deleted. The frequent radical changes to the model make drift more likely. A more sophisticated and processor intensive clustering method could solve these issues, but at the cost of real-time performance.

Alternatively, the new exemplar could be added to the matching component as

Require: $f_r^{(n)} \quad | \quad n \in [0; N_n], V, f_x^{(0)}$

$f_x^{(0)} \leftarrow f_r^{(0)}(\mathbf{w}(\mathbf{x}, \mathbf{v}))$

$N_n \leftarrow 1$, $C_1 \leftarrow \{f_x^{(0)}\}$, $w_1 \leftarrow 1$, $T_1 \leftarrow \overline{C_1} = f_x^{(0)}$

$\eta_{1,fg}(\mu, \sigma) = (0, \inf)$, $d_n^{(fg)} = \{d(f_x^{(0)}, f_x^{(0)})\}$

$\eta_{1,bg}(\mu, \sigma) = (0, 0)$, $d_n^{(fg)} = \{d(f_x^{(0)}, \mathbf{v}_{off})\}$

for $m = 1$ to N_m **do**

for $n = 1$ to N_n **do**

$\mathbf{v}_{opt} \leftarrow \arg_{\mathbf{v}} \min d[T_n(\mathbf{x}); f_r^{(m)}(\mathbf{w}(\mathbf{x}, \mathbf{v}))]$

if $\frac{\rho(d_{opt}|\eta(\sigma_n^{(fg)}, \mu_n^{(fg)}))}{\rho(d_{opt}|\eta(\sigma_n^{(bg)}, \mu_n^{(bg)}))} > 1$ **then**

$f_x^{(m)} \leftarrow f_r^{(m)}(\mathbf{w}(\mathbf{x}, \mathbf{v}_{opt}))$

$C_n \leftarrow C_n; f_x^{(m)}$

$w_n \leftarrow \begin{cases} \frac{w_n + \alpha}{1+\alpha} & n = n_{match} \\ \frac{w_n}{1+\alpha} & n \neq n_{match} \end{cases}$

Sort C_n in order of descending w_n

Break from **for** n loop.

else if $n == N_n$ **then**

$\mathbf{v}_{opt} \leftarrow \arg_{\mathbf{v}} \min d[f_x^{(m-1)}(\mathbf{x}); f_r^{(m)}(\mathbf{w}(\mathbf{x}, \mathbf{v}))]$

$f_x^{(m)} \leftarrow f_r^{(m)}(\mathbf{w}(\mathbf{x}, \mathbf{v}_{opt}))$

$n' = \max(N_n^{(max)}, n + 1)$

$C_{n'} \leftarrow \{x_m, f_x^{(m-1)}\}$

$w_{n'} \leftarrow 0$

$T_{n'} \leftarrow f_x^{(m)}$

Update $\sigma_{n'}^{(bg)}, \mu_{n'}^{(bg)}, d_{n'}^{(bg)}, \sigma_{n'}^{(fg)}, \mu_{n'}^{(fg)}, d_{n'}^{(bg)}$

end if

end for

end for

Figure 7.4: The SMAT algorithm

before. However, the clusters no longer represent nearby points in appearance space. Instead they represent a connection by causality, so the median loses its meaning as the cluster becomes over-sized. To surmount this issue, the founding exemplar of the cluster is used as a cluster-representative instead. This assumes that if $C_{k'}$ obtains a successful match in the current pose, successful matches will occur in all similar poses. For its simplicity this approach was taken when employing the ZIP.

7.3 N-tier hierarchical models

The SMAT method for clustering has so far only been considered for representing feature appearance as it varies through a sequence. In fact SMAT can be used to cluster any kind of data. All that is required is: a method to describe the data, and some measure to differentiate inliers and outliers.

For large non-rigid objects, using the single appearance-model described in Section 7.2 requires large numbers of clusters to represent all the variations in object appearance. Not only is this computationally expensive, but it is poorly representative, since insufficient exemplars are available to populate each cluster. Higher order warp parameterisations can reduce the number of clusters required, but these are still somewhat limited in their descriptiveness. An alternative is a bag-of-features approach: where the object is treated as multiple small features that are independently tracked and modelled. However, small features are more prone to failure, as they consist of fewer pixels and occlusions therefore are more likely. Another alternative is to use a hierarchical approach, where a larger region is tracked and used to pre-align smaller child features.

N-tier SMAT utilises both ideas and extends them by using the relative positions

of parents and children in a tree structure to build up a model of *shape*. An example of a 3 tier structure, used to model a face, is shown in Figure 7.5. Using the shape-model, a likelihood may be attached to the position of the child features, and hence to the probability of tracking failure. This allows tracking failures to be corrected and possible erroneous updates to the model to be blocked. N-tier SMAT thereby combines the advantages of tracking large features (robust to noise, occlusions and large motions) with those of tracking multiple small features (rich object description and accurate modelling of articulated shapes). The shape-model is also treated as a collection of weighted clusters: $p_{shp} = \sum_n w_n \eta(\mu_n, \sigma_n)$.

A recursive alignment is applied starting at the root-parent feature, in a manner illustrated in Figure 7.5 for a human face. First the largest "head" feature is aligned using the SMAT process discussed in Section 7.2. Second, the child features of the head (the "forehead" and "mouth region") are offset from their positions in the previous frame by the interframe shift of their parent. Third, a SMAT alignment is applied to each child as per Section 7.2, *except* the update is held in abeyance. Steps two and three are repeated for the "mouth region" children: "left mouth corner" and "right mouth corner", again with their respective SMAT updates held in abeyance.

In step six the positions of the "left mouth corner" and "right mouth corner" are concatenated into a vector describing the "mouth region" shape, which is fitted to each cluster of the shape-model. If the current shape fits an existing cluster, the shape-model is appropriately updated and the updates to appearance-models for the child features that were held in abeyance are applied. If the current shape does not fit the existing model, a new cluster is generated, but the appearance updates of the children are blocked. In addition, the current shape is "corrected" to fit the nearest cluster. The correction simply changes the initial position for

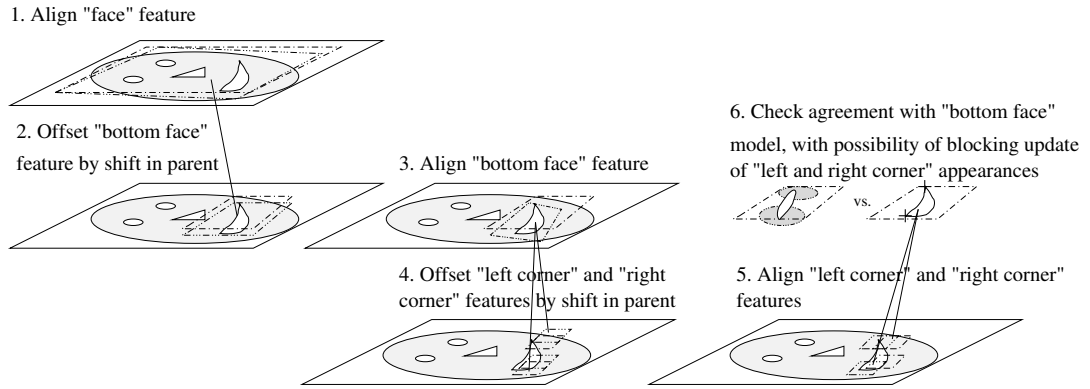


Figure 7.5: Illustration of the structure of a 3 tier shape-model and how the shape-model is used to recursively align and update the appearance-models at each tier. In steps 1 to 5, a hierarchical approach is used to match the features based on appearance, using shifts in the parent features to update the initial position of children. Step 6 is the first in series of steps going back up the model, where the updated positions based on appearance are compared to existing shape models. If the new shape is novel, a new shape cluster is spawned, but the appearance update is blocked and the shape corrected, because a failure may have occurred. Subsequently similar shapes will reinforce the new cluster, indicating that further corrections are not necessary.

optimisation in the next frame. If the corrected track is not an error, but due to an unseen shape, then the tracked position in the next frame should match the new shape-cluster. Step six is applied again at the level of the parent feature and updates are propagated upwards through the hierarchy.

The above shape update process is very similar to the SMAT appearance modelling process of Section 7.2, with some slight differences. Firstly, a PCA model is fitted to each cluster of shape exemplars, and Mahalanobis distance is use to measure the similarity of shape. Secondly, "background" and "foreground" shape scores are not available as they are for appearance, so a hard bound is required to establish cluster membership. A bound of 2σ, was chosen empirically, and worked well for all the sequences tested.

The shape-model consists of the positions of children *relative to their parent*, so a central feature (*i.e.* star model) or mean position is not required for a coordinate system base. This is an important advantage, since the use of a central feature makes tracking dependent on that feature's reliability. On the other hand, using an mean position allows outlying points in the shape to deform the entire shape and induce tracking failure in the remaining features.

As implemented, the hierarchical approach may be extended to any number of levels where multiple appearance-models are sandwiched together with shape-models, and is hence referred to as a *thick* shape-model.

A similar hierarchical alignment was also implemented, which does not explicitly model the shape of the object. Rather, the relative shift of parent objects is applied to the initial positions of child objects without any further modelling. This still implicitly models the dependence of child objects on their parents, which is similar to a multi-scale registration [32, 77]. This approach is referred to as a *thin* shape-model. Both methods yielded significant improvements over

independently tracked features. However, the use of a thick shape-model allows recovery from tracking failures in child features. Different warps may be used at different levels in the hierarchy as long as transformations between the two warps are available. Such transformations are not necessarily unique.

For the thick shape-model, the child feature positions are concatenated together to form a $N_f N_{\mathbf{v}}$ component vector, where N_f is the number of features and $N_{\mathbf{v}}$ is the number of vector components. As the use of PCA implies, a multi-modal normal distribution for shape is used, to give a Mahalanobis distance measure:

$$d_{mh}(\mathbf{s}_X) = \sqrt{(\mathbf{s}_X - C_\mu)^T C_\sigma^{-1} (\mathbf{s}_X - C_\mu)} \tag{7.8}$$

where C_μ is the mean of the cluster, and C_σ is the standard deviation of the cluster. In practice, the number of available shape exemplars is limited, so C_σ will often be non-invertible. Hence, Singular Value Decomposition is applied to obtain the eigenvectors C_{vec} and eigenvalues C_{val} from C_σ and this distance equation is used instead:

$$d_{mh}(\mathbf{s}_X) = \sqrt{C_{vec}^T (\mathbf{s}_X - C_\mu) C'_{val}} \tag{7.9}$$

where C'^{-1}_{val} is the pseudo-inverse of C_{val} with values close to zero being floored to zero for stability. The N-SMAT algorithm is outlined in Figure 7.6.

7.4 Testing methods

Testing tracking applications typically involves the researcher hand selecting the positions of features in each frame. Such an operation is time consuming, so testing using large numbers of long sequences is impractical. Hand selected features are not necessarily accurate, from the point of view of the registration algorithm, since a human's selection may be based largely on context unavailable to the

Align using SMAT, store the update without applying it
for All children **do**
  Apply change in parent to child
  Align children using SMAT
end for
Assemble shape vector
if Shape fits existing model cluster **then**
  Update matching cluster with new shape
  Apply the child's update
else
  Generate new shape component
end if

Figure 7.6: Outline of the N-SMAT algorithm

tracker. Conclusions are difficult to draw in this case. To overcome this problem the zig-zag-zen family of semi-automated tests is proposed.

7.4.1 Zig-zag-zen tests

In general, the only position that is known with absolute certainty, from the point of view of the tracker, is the position of the selected feature in the first frame. Any comparison against this position in the same frame is absolute as well. The zig test uses this property by tracking a frame through a sequence from frame 0 to frame N_m and then by reversing the sequence back to frame 0. The second instance of frame 0 is referred to as frame $2N_m$. The area of overlap between the $f_x^{(2N_m)}$ and $f_x^{(0)}$ gives an indication of how far the tracker has drifted from its original target. This is referred to as the *zig test*. Note that for trackers using

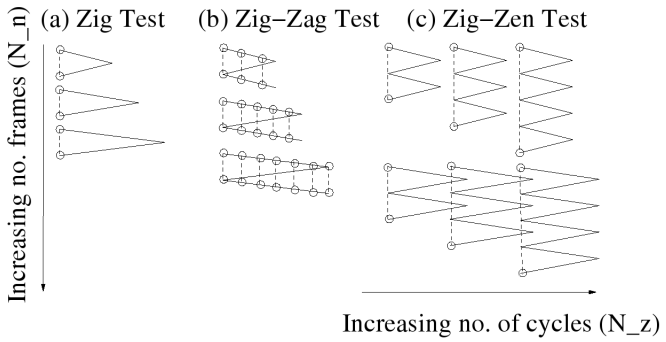


Figure 7.7: Zig-Zag-Zen tests consist of running sequences sequentially forwards and backwards and comparing the exemplars at intervals of $2N_m$ (indicated by joined circles). Three types of tests are shown: zig tests, where the position in the initial frame at the beginning and end of a cycle are compared, zig-zag tests, where a further forward cycle is made allowing comparisons in every frame, and zig-zag-zen tests where increasing numbers of cycles are used to analyse tracking accuracy.

models of dynamics, the dynamical model should have its signs inverted at frame N_m, due to the reversed time flow.

Generally, as sequence length increases, so does track error. However as shown in Figure 7.8, this is not always case, because sometimes the tracker can re-acquire its original target later on. In some cases failure occurs in the forward phase, and re-acquisition occurs in the reverse phase, giving a false "success". For this reason *some* user input is required to give a ground-truth at intervals in the sequence (in this work, $\frac{1}{10}N_m$ intervals were used). Tracked positions too far from the ground-truth are flagged as failures automatically. Additional performance figures may be obtained by varying the proportion of the sequence that is used.

Although the matched region in the frames after frame 0 might not be true representations of the original feature, they are representations of *some* feature.

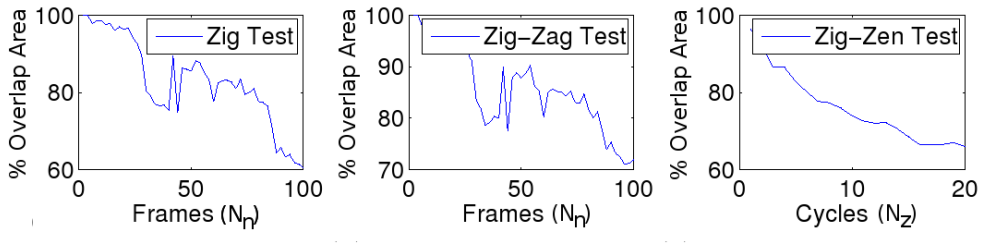


Figure 7.8: Examples of results obtained from Zig-Zag-Zen tests. In general as the number of frames and cycles increases, the area of overlap at the beginning and end of each test will decrease. There are some exceptions where re-acquisitions of the original feature occur on the backward cycle and a temporary improvement in performance is observed at around frame 50. For this reason, some manual ground-truthing is necessary to "sanity check" the results.

If the sequence is again cycled forwards to frame $3N_m$ (N_m), as shown in Figure 7.7b, all pairs of exemplars separated by $2N_m$ may be compared. This variation is referred to as the *zig-zag test*. For models more complex than the naive update model though, the appearance and shape-model dynamics can bias the results.

A final variation is to cycle through the sequence multiple times, which is referred to as the *zig-zen test* (a Zig-Zag N_z times). Similar to the zig test, the overlap between $f_x^{(0)}$ and $f_x^{(N_z N_m)}$ is measured. As shown in the example in Figure 7.8 this will generally also induce steadily increasing drift for increasing numbers of cycles.

Only zig tests were performed on the image sequences, with a varying N_m, in this work. Zig-zag tests were avoided, because complex models, where model dynamics would have an effect, were being examined. Zig-zag-zen tests were also avoided because they were time consuming to perform, and added little information to the results obtained from zig tests.

7.4.2 Error Metrics

The dissimilarity of exemplars $f_x^{(0)}$ and $f_x^{(2N_m)}$ may be measured in many ways, such as: finding the SSD of $f_x^{(0)}$ and $f_x^{(2N_m)}$; finding the difference between $\mathbf{v}_0$ and $\mathbf{v}_{2N_m}$, or measuring the overlap between $f_x^{(0)}$ and $f_x^{(2n)}$.

The first error metric biases towards SSD based trackers. Also, high appearance-match scores may be obtained despite a feature being mismatched. Differencing warp parameters gives some measure of error, but produces multiple parameters for assessment and is hence difficult to interpret. Combining the warp difference into a single value is problematic, due to the differences in scale between the various values.

Hence in this work, the ratio between the overlapping region and the area of the larger of the two patches $f_x^{(0)}$ and $f_x^{(2N_m)}$, is measured.

$$\frac{A(f_x^{(0)} \cap f_x^{(2N_m)})}{\max(A(f_x^{(0)}), A(f_x^{(2N_m)}))} \tag{7.10}$$

Using a max ratio avoids the issue of the smaller exemplar being completely encompassed by the larger one and then giving a falsely "perfect"result of 1.

7.4.3 Feature selection

The choice of feature is an important part of an effective tracking strategy, since badly chosen features will cause early failure. As a simple example consider trying to track a vertical edge using an image patch. The lack of a constraint in the y direction will lead to random vertical drift. The high error that results in such a case is due to the poor choice of feature rather than a low quality tracking algorithm.

Humans are good at selecting relevant features, however their choices are augmented by additional knowledge of the world which is unavailable to the average automated feature tracker. Hence, manually selected features are not necessarily good to track either.

Objects were pre-selected and the child features were chosen using the entropy based feature detector of Kadir and Brady [36], because the similarity measure used, MI, is also entropy based. This is analogous to choosing features based on their corner ratio for SSD based LK-trackers, as proposed by Shi and Tomasi [72].

7.4.4 Test data

Eleven long (568 frames on average) and challenging sequences of various subjects were obtained. These were split into two groups: eight sequence tracking single features, and three sequences for testing shape-models using 3-tiers of features. Objects tracked included humans in sporting events, human faces with changing expressions, gesturing human hands, pedestrians, yachts, and wild animals in Kruger National Park. Backgrounds were in some cases highly variable and cluttered, including ripples in a swimming pool and moving bushes. Likewise some sequences had lighting conditions that varied drastically. The first frame of each test sequence is shown in Figure 7.9.

7.5 Results and Discussion

The test framework of Section 7.4 was used to compare the proposed SMAT method to existing trackers. In particular the tests attempted to answer some specific questions. How much do models of increased complexity improve performance? Does the zero principle improve performance? Does the use of higher

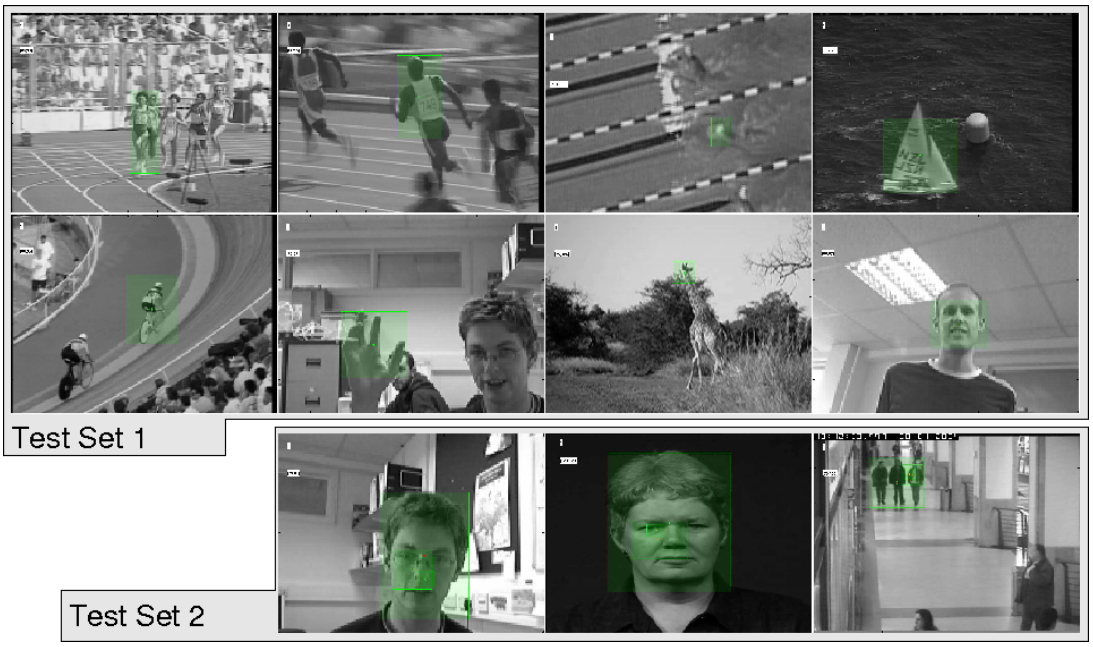


Figure 7.9: Test sequences used. Set 1 was used to test single feature tracks. Set 2 was used to track 3-tier of object models. The tracking performance on the smallest feature (lowest in the hierarchy) was examined, as this feature is the one most likely to fail due to occlusion or fast motion. The main frame length was 568 frames, and features were hand-selected from the top ten features selected by a Kadir-Brady saliency detector.

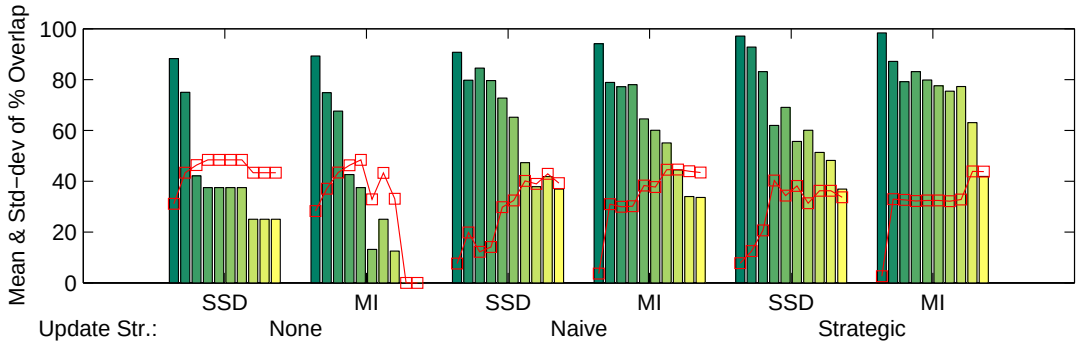


Figure 7.10: Comparison of SSD and MI metrics for one/two template methods: No-update, Naive-update and Strategic-update. Warp used: translation. Bar groups show mean while increasing sequence length. Lines show standard deviation.

order warps improve performance? Does the use of shape-models improve performance?

Ten zig tests were performed for each test sequence, ranging in length from 10% to 100% of the sequence length The mean percentage overlap between the rectangles at the beginning and end the sequences is shown in Figure 7.10 to 7.16. The standard deviation is also shown to express variability of the results. Bar groups show performance as sequence length is increased in 10% intervals from 10% to 100%, with the green bars indicating the mean percentage overlap and the red lines indicating the standard deviation. The standard deviation increases relative to the mean value as the sequence length increases. This occurs because complete failures are detected for increasing numbers of sequences, whose results are excluded to avoid biasing the results. The result is fewer tests being used to calculate the mean and a higher relative standard deviation.

In the first set of tests, test-set 1 was used to compare the one/two template update methods, namely: no-update, naive-update and strategic-update. These

methods were compared for the SSD and MI (standard sampled) similarity metrics, as shown in Figure 7.10 In all cases translation warps were used.

Although the results of Section 3.5 showed SSD comparing well to MI, in this set of tests, the variations in pose and noise within the image sequences strongly demonstrate the advantages of using MI compared to SSD. Superior results for MI are seen for all the models used. These are a direct result of MI not making any assumptions about the relationship between the intensity of the template and that of the reference image. MI generally has a smaller basin of convergence than SSD, so the no-update method causes earlier failure due to mismatch.

Comparing the different update methods, the sudden failure due to mismatch that occurs when using the no-update approach is clearly evident. The naive-update approach shows significant improvements, with more graceful degradation as the sequences lengthen. The use of an additional template for the strategic-update approach improves the results still further, especially for the longer sequences. This good performance is due to the descriptiveness granted by the additional template, which allows greater variation in pose without failure.

Tracking performance is subjectively illustrated for SSD and MI while tracking the head of an Olympic swimmer in Figure 7.11. In both cases, SMAT algorithms with translation warps were used. Tracking the feature (swimmers) head in this sequence is difficult for two reasons. Firstly, the head is often submerged in water completely changing its appearance, shown in frame 19. Secondly, the ripples in the water create numerous outlier pixel intensities in the template, visible in most frames. The result is that the SSD tracker fails due to mismatch frame 138, while MI manages to track the feature until frame 250.

In the second set of tests, test-set 1 was used to examine whether the use of higher order warps improves performance. The results are shown in Figure 7.12

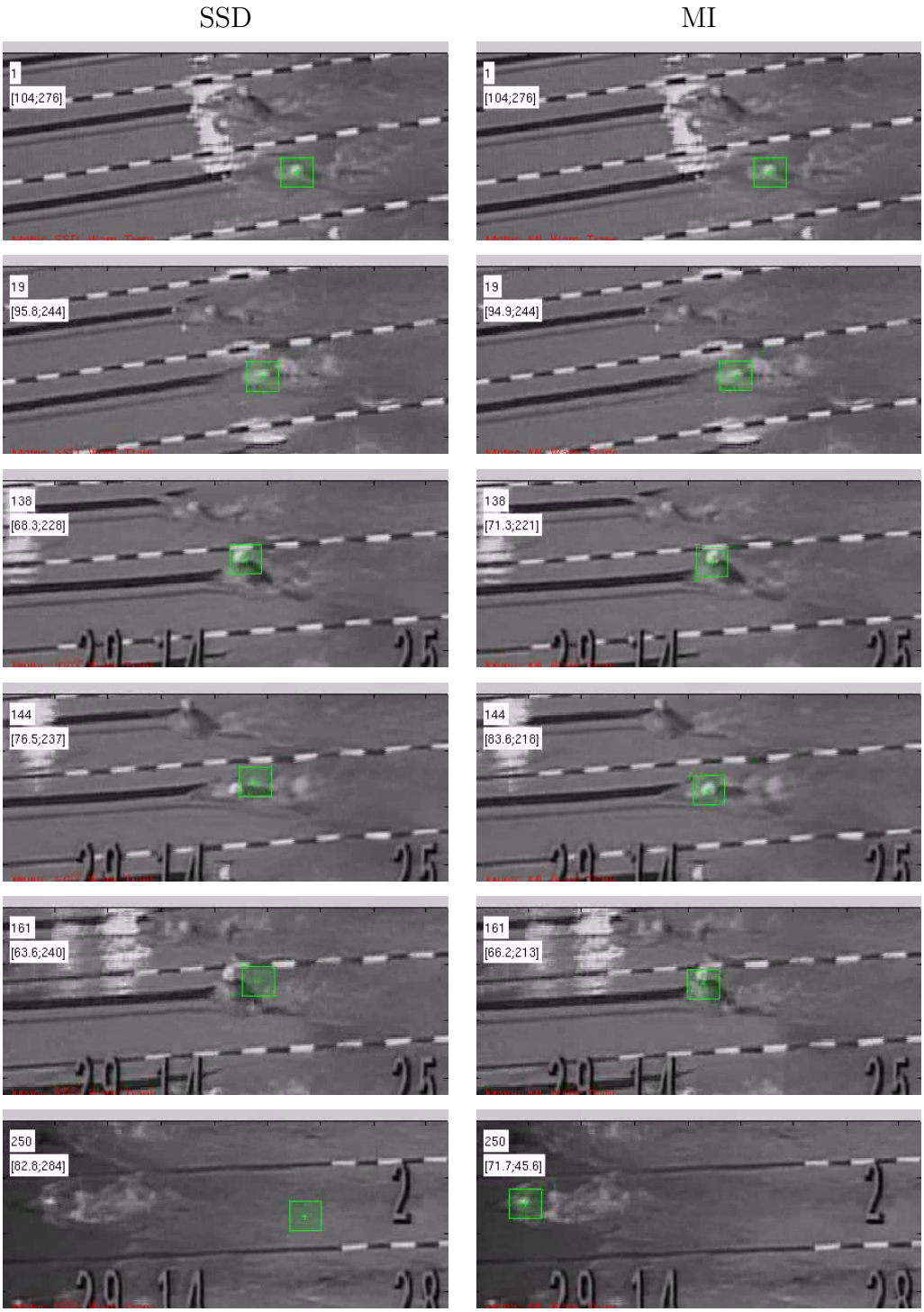


Figure 7.11: Subjective comparison of SSD and MI in "swim" sequence

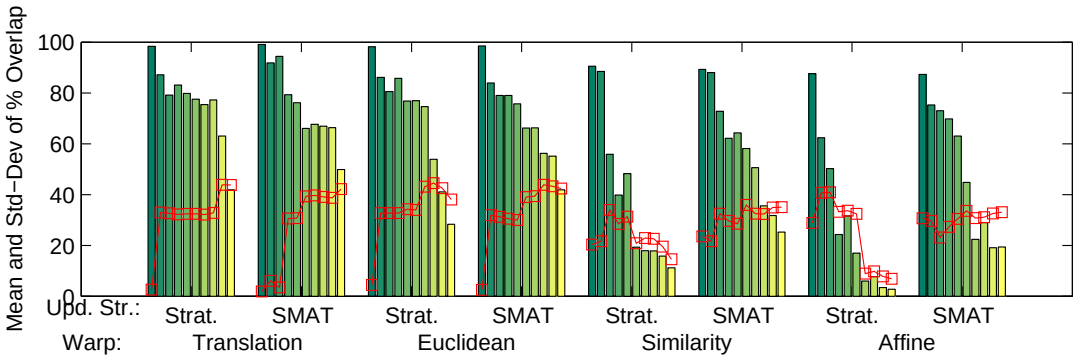


Figure 7.12: Comparison of performance using four warp types: Translation (2DoF), Euclidean (3DoF), Similarity (4DoF) and Affine (6DoF) for Strategic and SMAT update methods. Bar groups show mean while increasing sequence length. Lines show standard deviation. Metric: MI.

for the strategic-update approach and SMAT. In general performance degraded as the number of DoF increased. This is partially due to the high sensitivity of rotation, scale and affine parameters, which can lead to unstable optimisation. This is a particular issue if the template size is low, since there is insufficient data to for the number of parameters being optimised. Increasing the number of DoF gives more opportunities for drift to set in, which accounts for the increasingly graceful degradation as the number of DoF increases. Furthermore, the greater number of DoF allows the template more variation relative to its initial position, so a low area of overlap may be obtained, despite the centres of the two regions being in close proximity to each-other.

For the translation warps, the strategic-update method showed better performance at intermediate frames, while SMAT performed better at the beginning and end of sequences. Failure for the strategic-update method was also somewhat more dramatic when it occurred unlike the graceful degradation of SMAT. This was due to the strategic-update method tracking well using the initial template

and failing due to mismatch. SMAT has a more general description and degrades in performance more slowly.

In all other cases, SMAT significantly outperformed the strategic update approach, because when SMAT drifts slightly from over-sensitive parameterisation, recovery using a previous appearance is possible. This is more difficult for strategic-update, because two templates cannot adequately describe all the variations in appearance. When an incorrect update to the appearance-model occurs failure is absolute, because the correction cannot escape erroneous local minima.

Figure 7.13 subjectively compares the tracking performance of the Strategic update and SMAT algorithms while tracking an Olympic athlete. In both cases MI was used as a similarity metric. Translation warps were used to force the tracking algorithms to generalise well despite large pose and scale changes, or fail. The results show, the strategic update algorithm fails due to mismatch at around frame 122, with complete failure by frame 311. Notice how similar the region being tracked by strategic update is to the first template, because the model does not change sufficiently to represent the new appearance. SMAT on the other hand tracks the runner throughout the sequence.

An example of tracking a person through a lobby, using SMAT with a similarity warp, is given in Figure 7.14. This particular sequence is difficult because the feature (pedestrian) being tracked moves from a sunlit region to a shadowed region, with a large change in appearance. However the structure of the feature remains fairly similar throughout the sequence, allowing successful tracking when using SMAT combined with MI. No other combination of tracking method and similarity metric managed to track past frame 50.

In the next set of tests, test-set 1 was used to examine whether using the lattice

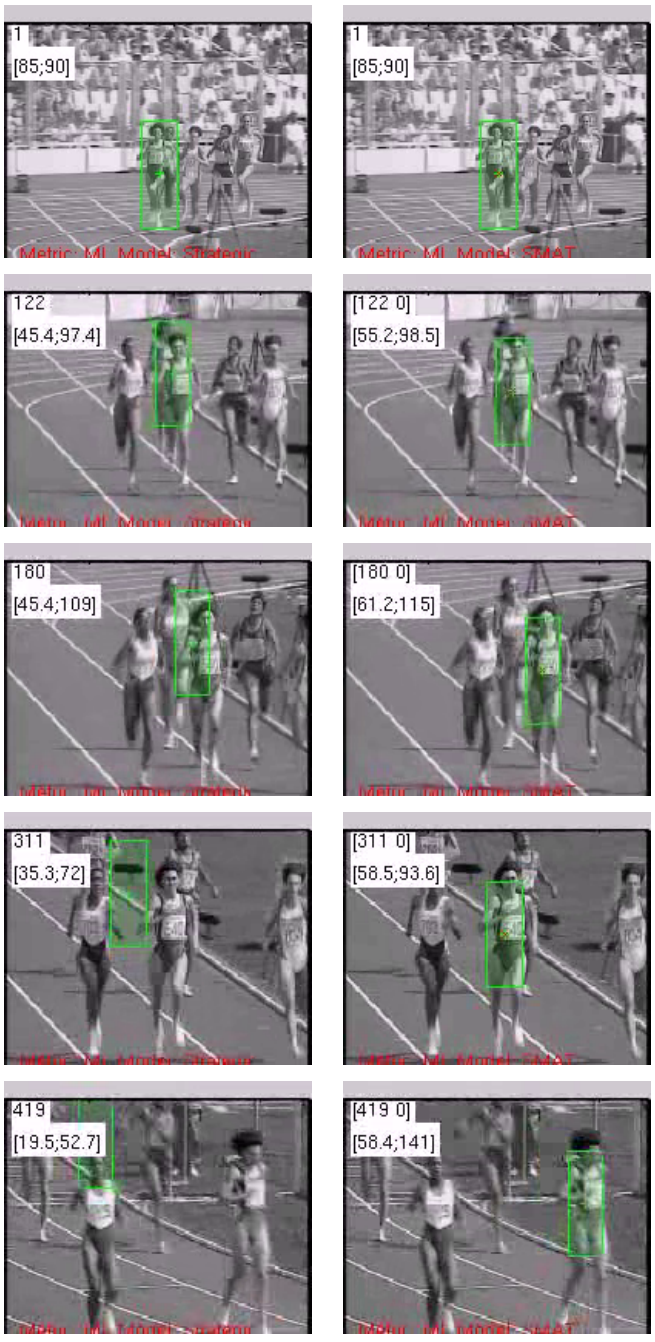


Figure 7.13: Subjective comparison of Strategic Update and SMAT in "run" sequence. The strategic algorithm fails by frame 311 because the scale changes cannot be represented using only two exemplars. SMAT tracks to the end of the sequence.

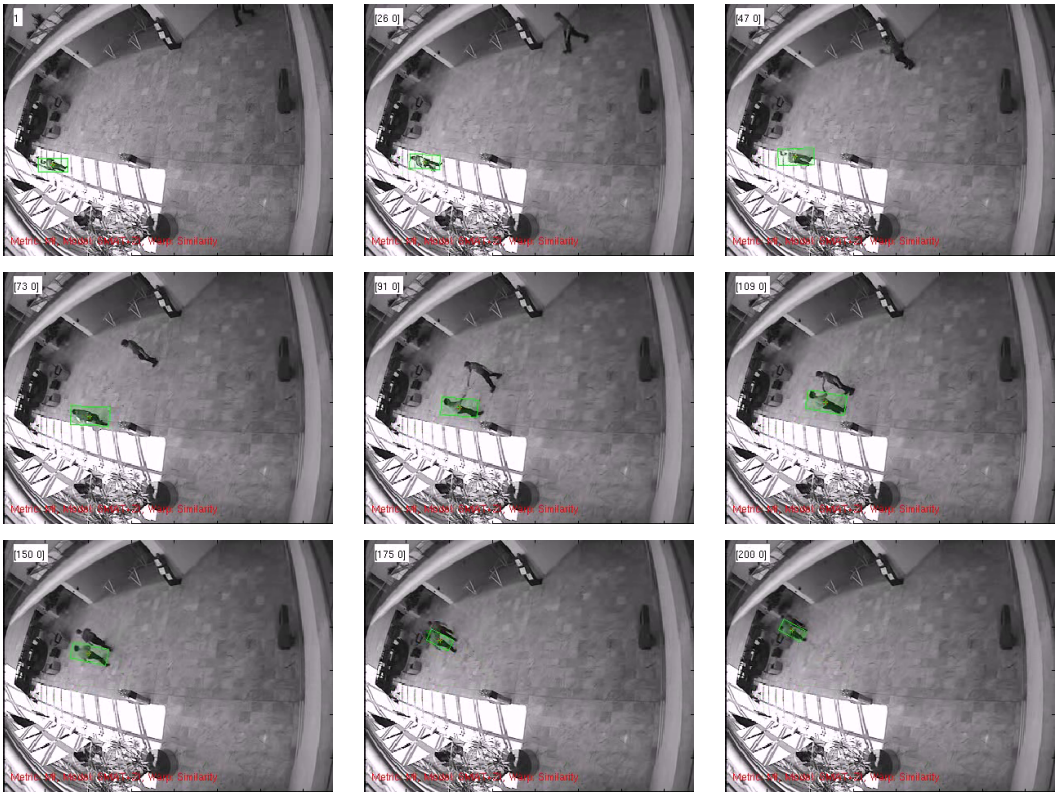


Figure 7.14: Example of tracking using MI and similarity warps in "lobby" sequence. The pedestrian moves from an area of sunlight to and area of shadow, making tracking difficult. The use of SMAT combined with MI allowed successful tracking to the end of the sequence.

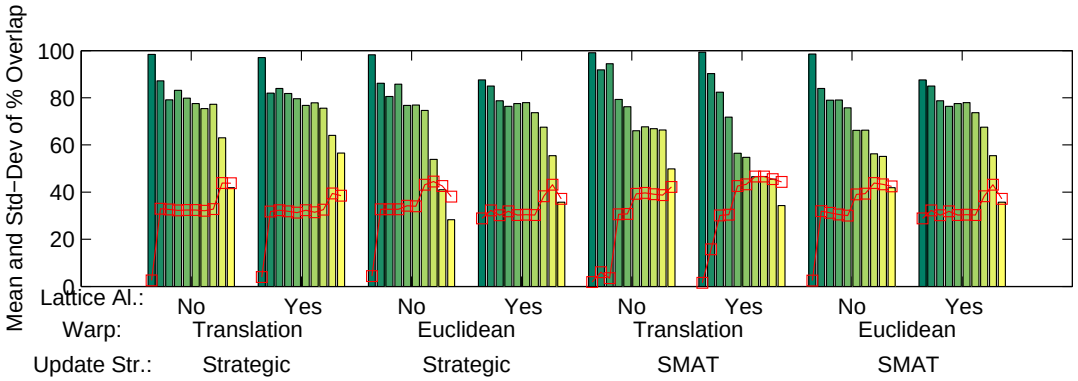


Figure 7.15: Comparison of between using warped templates and lattice aligned templates for strategic-update & SMAT methods, and Translation & Euclidean warps. Bar groups show mean while increasing sequence length. Lines show standard deviation. Metric: MI.

aligned templates improves performance or not. The strategic-update and SMAT methods were used for this tests. Translation and Euclidean warps were also compared. The complete set of results is shown in Figure 7.15.

For the strategic-update method the inclusion of the ZIP improved results slightly. For translation, the improvement probably occurred because the use of ZIP meant that data used to match the reference and the template was not diluted by multiple interpolations. This not only reduces drift, but slightly widens the appearance-model in the case of the strategic-update method.

For Euclidean warps, the use of ZIP widens the manifold of the appearance-model still further, as the inclusion of an extra DoF allows the two templates to be rotated relative to each-other. The large error that occurs early on, is due to the extra DoF and because rotation is a sensitive parameter.

Non-ZIP SMAT, generally outperformed strategic-update for longer sequences and had a more graceful degradation in performance as sequences increased in length. However, strategic-update performed well at intermediate sequence

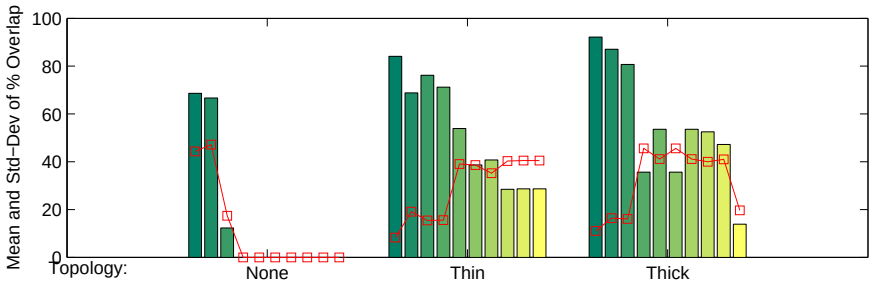


Figure 7.16: Comparison of performance for no/thin/thick shape-models. Bar groups show mean while increasing sequence length. Lines show standard deviation. Tracking method: SMAT. Metric: MI. Warp: Similarity + Translation + Translation

lengths due its conservative nature, with a sudden degradation in performance.

The use of ZIP degraded the performance of SMAT for translation warps. The reason is the large number of clusters that are created in the ZIP version of the SMAT algorithm. As only the first few clusters are used, and new clusters are frequently created, the clusters never become properly populated and do not adequately describe local regions in the appearance manifold.

For Euclidean warps, the performance of SMAT improved slightly when using ZIP. The extra degree of freedom imbues better representation of the feature being tracked, which offsets the negative effects of large numbers of improperly populated clusters. The initial error occurs for much the same reason as it did for the strategic-update method.

It can be concluded that the use of ZIP will generally only improve the performance of one/two template methods, since they do not fully represent features. In the case of large appearance-models such as SMAT, the extra descriptiveness offered by ZIP does not greatly augment that of the existing appearance-model.

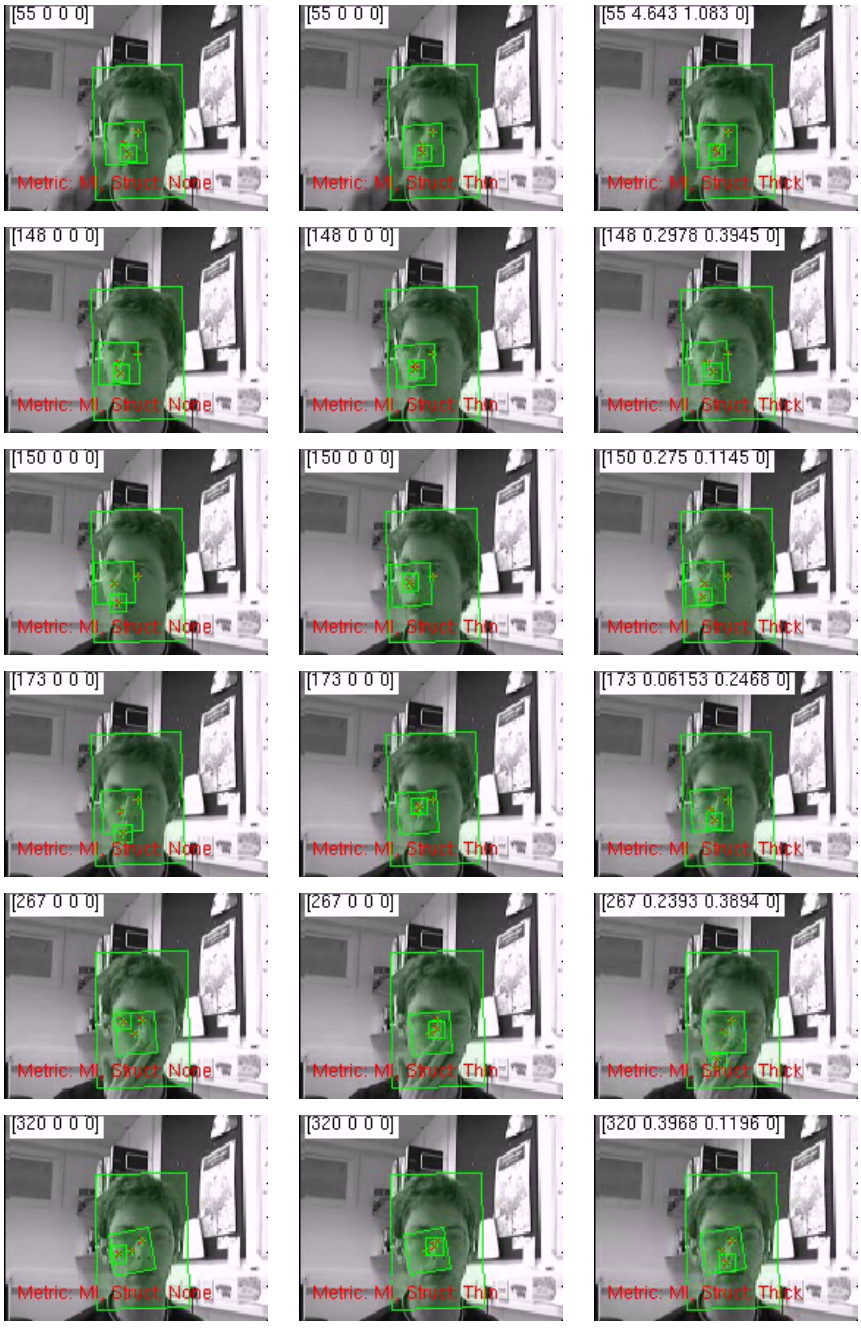
Test-set 2 was used to examine the effects of modelling shape-models in multiple

tiers. In each test three features, in three tiers, were used. A similarity warp was used for the top level tier, while translation warps were used for the second and third tiers to avoid over parameterisation. The results in Figure 7.16 show the performance for the lowest tier *i.e.* the feature most likely to fail.

As shown, even using a thin shape-model (i.e. treating the problem as a multi-scale registration) improves performance significantly over independent tracking. Explicitly modelling the shape improves performance further. However, thick shape-models should be used with care, since they can drag features off their correct position. This appears to occur in the "Thick" results of Figure 7.16 at 40% of the maximum sequence length. However the use of a thick shape-model allows re-acquisition to occur, hence the intermittent improvements seen for intermediate length sequences.

Figure 7.17 shows a subjective comparison between tracking with no shape-model, a thin shape-model and a thick shape-model using the "Nick's face" sequence. A hierarchy of three features was used for tracking: the face at the highest level using similarity warps, the left cheek using similarity warps, and the left nostril at the lowest level using translation warps. The smallest feature, namely the nostril, is the most likely to have tracking failure, due to the high likelihood of occlusion and the rapid movement relative to the template size.

As shown, in Figure 7.17, an early failure (frame 148) occurs for the independently tracked nostril, due to pose variation and rapid movement. Recovery never occurs. For the thin shape-model, failure occurs some frames later (by frame 173), because the optimising from high to low scale grants robustness to rapid movements. The thin shape-model never recovers from the mismatch error in frame 173. The thick shape-model, also fails slightly in frame 150, but a recovery by frame 173 occurs due to a correction by the shape model. A similar recovery occurs in frame 320,



No shape-model Thin shape-model Thick shape-model

Figure 7.17: Subjective comparison of the no/thin/thick shape-models for "Nick's face" sequence. Unlike the other methods, the thick shape-model recovers from a mismatch error in frame 150 and an occlusion in frame 267.

after an occlusion by the hand causes temporary failure in frame 267.

7.6 Conclusion

The Simultaneous Modelling and Tracking (SMAT) algorithm has been proposed, which builds up a model of the appearance of a feature using a single exemplar from the initial frame. This done on-the-fly while the feature is being tracked, allowing real-time performance. In outline, SMAT stores the exemplars of the feature extracted from each frame and clusters them. Using a greedy process, clusters are selected to track the feature in the current frame, based on their weight.

The SMAT method was extended to track multiple features of a single object and model the relative positions between these features to build up a shape-model. The same clustering process used for the appearance- and shape-models. The extended method uses a hierarchical approach to track the features, by placing the features into a tree-like structure, which consists of shape-models sandwiched between appearance-models. This method is referred to as N-tier SMAT.

A new set of semi-automated testing methods has been proposed called the zig-zag-zen tests. The test methods work by cycling forwards and backwards through a sequence. The position of tracker frame 0 is compared to its position at the end of the backwards cycle. The sequence length is progressively increased to examine how performance degrades. Ten ground-truth positions were used to ensure that a false successes from re-acquisition in the backwards cycle did not occur. The use of lattice-aligned exemplars was also proposed to avoid the onset of drift due to interpolation blur.

The proposed zig test was used to test the effects of the following on the per-

formance of a tracking application: similarity metric, coordinate warp function used, template update method used, and shape-model used. This test method allowed exhaustive testing on 11 challenging sequences with a mean length of 568 frames.

The use of the SMAT method yielded improvements over one/two template methods, because the method fully describes the manifold of a feature in appearance space, whereas one/two template methods only describe one point. Hence, the strategic-update method outperformed naive-update, which outperformed no-update.

The use of non-lattice aligned templates was also tested. For one/two template methods this improved the results, partially because of the lack of interpolation blur. The increase in descriptiveness granted to the appearance-model, by the use of exemplars slightly offset from the actual feature, probably had more of an effect. The improvement is particularly obvious for high DoF warps. For SMAT, ZIP gave little improvement (or degradation at times), due to the generation of multiple under-populated clusters.

N-tier SMAT yielded some performance improvements compared to independently tracking features, and over using a hierarchical approach to alignment. This is because a shape-model allows recovery from tracking failures.

Chapter 8

Conclusion

The aim of this work was to track non-rigid objects in real-time without *a priori* data. Tracking had to be robust to noise, occlusions and variation in lighting conditions. These objectives were partially met by developing a method to model the appearance and shape of a feature during tracking it, termed Simultaneous Modelling and Tracking (SMAT).

Additional robustness was imparted to the tracking algorithm by using Mutual Information (MI) as a similarity metric when comparing images. Two particular difficulties became apparent when using MI to register small image patches. Firstly, many different methods for the estimation of MI had been published and the best method for the purposes of tracking needed to be selected. Secondly, many of these methods had no published analytic derivative, precluding the use of a fast Newton-type optimisation method like Levenberg-Marquardt (LM).

To study the plethora of available MI methods, the four most widely used MI families were placed into a single mathematical framework in Chapter 3, namely: standard sampling, post-Parzen windowing, in-Parzen windowing and generalised partial volume estimation. It was shown that these families differ only in the

sampling function, *i.e.* how the joint-histogram is populated. The mathematical framework allowed the construction of analytic derivatives for each family, which was novel in every case except in-Parzen windowing and first order partial volume estimation. Importantly, the computation of the analytic derivatives shared redundancies with the function computation, yielding significant speed advantages when using LM optimisation, *e.g.* computing the MI, its Jacobian and Hessian for a 6DoF affine warp, only cost three times one function evaluation rather than the 36 times that would normally be expected.

Exhaustive testing was performed, to compare the various MI methods, Normalised Correlation (NC) and Sum of Squared Differences (SSD) measures to each-other. NC had a wide basin of convergence, allowing it to perform well for noise free, unoccluded images. However PVE, the best of the MI methods, managed to perform as well as NC at substantially lower computational cost. The disadvantage of PVE is its large bias (a shift in the function minimum away from its true position). For this reason, PVE is a recommended measure where speed is paramount and a large basin of convergence is required. When the images being registered are noise free and unoccluded, and accuracy of the final solution is important, NC or SSD are recommended.

Further investigation revealed that the inferior performance of standard sampling, post-Parzen windowing and in-Parzen windowing methods compared to PVE occurred because of artefacts in the function surface. In Chapter 4, it was shown that two types of artefacts exist: hiss (numerous discontinuities in the function surface) and glitches (patterns of local minima). Hiss arises from the finite bin-size of the joint-histogram and the discrete shifts in membership as sample points shift in the reference image dependent on the warp parameter. Glitches arise from harmonics between the pixel-lattice of the reference and template images.

The effect of hiss was minimal so long as the Newton steps were large enough. However, as the algorithm approached convergence the effects of hiss overwhelmed the macroscopic trends in the function surface. This resulted in non-zero convergence errors; even for initial positions within the basin-of-convergence. The effect of glitches is to generate widely spaced local minima and to to shift the position of the minimum away from its "true" position, *i.e.* bias. Measures to limit the effects of artefacts were examined and tested. The issue of artefacts has been examined before, but never in a complete way that considered positional (as opposed to amplitude) bias and never comparing all the MI families.

Notably, testing found a large bias associated with the impressive convergence capabilities of PVE, hence it is not recommended where final registration accuracy is important. Of all the methods to reduce artefacts, using super-resolution (increasing the sampling resolution) was found to give the lowest bias, while reducing the effects of both hiss and glitches in a controllable way. Hence, a trade-off between computation and accuracy was possible. Standard sampling with a controlled sampling rate is the recommended method for registration.

The idea of super-resolution was taken to its logical conclusion in Chapter 5, where the NP-windowing technique was extended to consider a pair of 2D images. NP-windowing integrates the effects of interpolation over an entire pixel to populate the histogram in a manner that is equivalent to sampling at infinite resolution. This was achieved by using the structure inherent in the ordering of image data, which is discarded by most sampling methods. Numerical integration was required for each neighbourhood of samples. However since standard rendering algorithms were used, NP-windowing was only an order of magnitude slower than standard sampling: making it practical for use in registration applications, rather than merely a theoretical curiosity. NP-windowing collapses to first order PVE in the simplest case of Nearest Neighbour interpolation.

Testing revealed an improved rate of convergence and lower bias for NP-windowed MI than for any other MI method. The bias was limited solely by accuracy of the interpolation method. Hence, NP-windowing should be the method of choice for applications such as medical image registration where accuracy is critical.

NP-windowed MI was further extended using a resolution aware approach. This did not yield expected improvements in bias or convergence, due to the effects of harmonics in the template and reference pixel-lattices. Substantial simplifications to the existing NP-windowing theory were also introduced using Green's theorem to perform a perimeter integral rather than an area integral.

To increase the speed of optimisation, MI was placed into a Lucas-Kanade framework, termed MILK, in Chapter 6. The MI function was reformulated into an inverse compositional formulation, where the roles of the template and reference image are inverted and the effects of warps and warp variations are separated. This was difficult for MI, because the the intensities of the reference and template are not separable in the joint-histogram. However with the assumption that the co-intensities remain similar a constant Hessian was obtained for a given template-reference image pair.

In testing with a registration algorithm, speed improvements of 17% were obtained. This less than expected improvement, occurred because the less accurate Hessian required more iterations to reach convergence. Moreover, the initial computations are relatively expensive, especially if the number of iterations is not large. Testing using naive updates on sequences of data showed that comparable registration and tracking performance could be obtained for the inverse compositional formulation.

The image registration algorithms that had been developed were finally applied to tracking features through image sequences, in Chapter 7. This raised a new

problem: how and when to update the template used for tracking. The SMAT algorithm was proposed, which built a multi-modal appearance model by storing examples of the feature being tracked and clustering these on-the-fly. The result was increased robustness over the current state of the art.

The proposed clustering method was used to model the position of multiple features relative to a parent object *i.e.* a model of shape. Multiple appearance and shape models were then stacked together to allow hierarchical registration of features, improving robustness and giving a richer description of articulated objects. In addition, a novel testing method for the semi-automated testing of tracking algorithms on multiple test sets, was proposed. This allowed testing using eleven sequences with a mean length of 568 frames.

Testing vindicated the choice of MI as a similarity measure, where it outperformed SSD in noisy sequences with changes in lighting conditions and outlying pixels. SMAT also outperformed other real-time tracking algorithms which used only one or two templates as appearance models, because the SMAT appearance model fully described possible variations in appearance. The use of shape models had the marked benefit of allowing recovery from tracking failures, in addition to increased robustness to occlusion and rapid movement.

8.1 Future work

In the future, the NP windowing method will be extended to other applications such as 3D medical imaging, where the accuracy of the final solution (bias) is important. This method will use a combination of MI alignment methods to optimise basin-of-convergence size, speed and final registration accuracy. In addition, extensions to unify generalised PVE and NP-windowing should be considered, by

interpolating probabilistically from neighbouring pixels. Such an approach will allow a more computationally efficient hierarchical optimisation to be performed. The improved NP-windowing implementation, using Green's theorem, will be applied to saliency detection methods.

Much research is still to be done in the use of multi-dimensional joint histograms, either due to the use of colour or the encoding of neighbourhood statistics. High-dimensional joints have not been practical before, due to the lack of samples. NP Windowing solves this issue and easily generalises to multi-dimensional statistics, where standard geometry can be used to detect simplex intersections and calculate volumes. The use of such multi-dimensional joints will impact on diverse range of applications including image restoration, resolution enhancement, segmentation and labelling.

Recently, detection methods have increasingly been used for tracking, but pre-training is still required. In future work, registration methods like SMAT should be used to bootstrap detection methods, with the algorithm reverting to registration when detection approaches fail. This should lead to more efficient algorithms where the trade-off between speed and tracking accuracy is explicitly made.

It is the hope of the author that this thesis places Mutual Information in the same position as Sum of Square Differences, as a mature and well understood similarity measure. MI and SSD can then be selected for a particular application based on their merits, rather than MI being avoided, due to a perceived complexity, computational expense and impracticality.

Appendix A

Jacobians and Hessians

A.1 Normalised Correlation

Normalised correlation relies on computing the products of pixel values. As a result it is more robust to lighting changes since outliers have less of an influence.

$$D_{nc}(\mathbf{v}) = \frac{\sum_{\mathbf{x}} f_t(\mathbf{x}) f_r(\mathbf{x_w})}{(\sum_{\mathbf{x}} f_t(\mathbf{x})^2)^{\frac{1}{2}} (\sum_{\mathbf{x}} f_r(\mathbf{x_w})^2)^{\frac{1}{2}}} \tag{A.1}$$

The derivative of the normalised correlation function is obtained by applying a combination of the chain rule and quotient rule. Note that because of the derivative operation is distributive with respect to addition, it may be applied within the sum. Since the function consists of several sums multiplied together too many steps are required in this derivation to show here.

$$\frac{\partial D_{nc}}{\partial \mathbf{v}} = \frac{\sum_{\mathbf{x}} f_r^2 \sum_{\mathbf{x}} f_t \nabla f_r \frac{\partial \mathbf{w}}{\partial \mathbf{v}} - \sum_{\mathbf{x}} f_t f_r \sum_{\mathbf{x}} f_r \nabla f_r \frac{\partial \mathbf{w}}{\partial \mathbf{v}}}{(\sum_{\mathbf{x}} f_t^2)^{\frac{1}{2}} (\sum_{\mathbf{x}} f_r^2)^{\frac{3}{2}}} \tag{A.2}$$

Note that $\nabla f_r = \frac{\mathrm{d}}{\mathrm{d}\mathbf{w}} f_r$.

The Hessian for the normalised correlation function is even more complicated to obtain and Mathematica was used for its derivation:

$$\begin{aligned}\frac{\partial^2 D_{nc}}{\partial \mathbf{v}^2} &= \left(\sum f_t^2\right)^{-\frac{1}{2}} \left(\sum f_r^2\right)^{-\frac{5}{2}} \Big\{ 3 \sum f_t f_r \sum f_r (\nabla f_r \frac{\partial \mathbf{w}}{\partial \mathbf{v}})^T \sum f_r (\nabla f_r \frac{\partial \mathbf{w}}{\partial \mathbf{v}}) \\ &+ \left(\sum f_r^2\right)^2 \sum \left(f_t \nabla^2 f_r (\frac{\partial \mathbf{w}}{\partial \mathbf{v}})^T (\frac{\partial \mathbf{w}}{\partial \mathbf{v}}) + f_t \nabla f_r \frac{\partial^2 \mathbf{w}}{\partial \mathbf{v}^2} \right) \\ &- \sum f_r^2 \Big[\sum f_r (\nabla f_r \frac{\partial \mathbf{w}}{\partial \mathbf{v}})^T \sum f_t (\nabla f_r \frac{\partial \mathbf{w}}{\partial \mathbf{v}}) + \sum f_t (\nabla f_r \frac{\partial \mathbf{w}}{\partial \mathbf{v}})^T \sum f_r (\nabla f_r \frac{\partial \mathbf{w}}{\partial \mathbf{v}}) \\ &+ \sum f_t f_r \sum \left((\nabla f_r)^T (\nabla f_r) (\frac{\partial \mathbf{w}}{\partial \mathbf{v}})^T (\frac{\partial \mathbf{w}}{\partial \mathbf{v}}) + f_r \nabla^2 f_r (\frac{\partial \mathbf{w}}{\partial \mathbf{v}})^T (\frac{\partial \mathbf{w}}{\partial \mathbf{v}}) + f_r \nabla f_r \frac{\partial^2 \mathbf{w}}{\partial \mathbf{v}^2} \right) \Big] \Big\}\end{aligned}$$

The warp derivatives used in Hessians are sometimes complicated to visualise, so some warp derivatives have been provided for reference in Appendix A.2.

A.2 Various Warp Hessians

There are four types of coordinate warps $\mathbf{w}(\mathbf{x}, \mathbf{v})$ that are used in this work: translation, Euclidean, similarity and affine warps. These have the following

Hessians:

$$\nabla f \frac{\partial^2 \mathbf{w}_{tx}}{\partial \mathbf{v}^2} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

$$\nabla f \frac{\partial^2 \mathbf{w}_{eu}}{\partial \mathbf{v}^2} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & (f_x\, f_y) R''(x\, y)^T \end{pmatrix}$$

$$\nabla f \frac{\partial^2 \mathbf{w}_{si}}{\partial \mathbf{v}^2} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & (f_x\, f_y) R'' v_4 (x\, y)^T & (f_x\, f_y) R'(x\, y)^T \\ 0 & 0 & (f_x\, f_y) R'(x\, y)^T & 0 \end{pmatrix}$$

$$\nabla f \frac{\partial^2 \mathbf{w}_{af}}{\partial \mathbf{v}^2} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

In all the above equations:

$$R = \begin{pmatrix} \cos v_3 & \sin v_3 \\ -\sin v_3 & \cos v_3 \end{pmatrix}$$

$$R' = \begin{pmatrix} \sin v_3 & -\cos v_3 \\ \cos v_3 & \sin v_3 \end{pmatrix}$$

$$R'' = \begin{pmatrix} -\cos v_3 & -\sin v_3 \\ \sin v_3 & -\cos v_3 \end{pmatrix}$$

Bibliography

[1] I. Amidror. Scattered data interpolation methods for electronic systems: A survey. *Journal of Electronic Imaging*, 11(2):157–176, April 2002. 11

[2] A. Argarwal and B. Triggs. Tracking articulated motion using a mixture of autoregressive models. In *Proc. 8th European Conf. On Computer Vision*, volume 33, pages 54–65, Prague, May 2004. 2

[3] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, March 2004. 10, 14, 114, 117

[4] Simon Baker and Iain Matthews. Lucas-kanade 20 years on: A unifying framework: Part 1. Technical Report CMU-RI-TR-02-16, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, July 2002. 4

[5] A. Bartoli and A. Zisserman. Direct estimation of non-rigid registrations. In *Proc. 15th British Machine Vision Conf.*, Kingston University, London, UK, September 2004. 13

[6] M.J. Black and A.D. Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view based representation. *Int'l Journal of Computer Vision*, 26(1):63–84, 1998. 2

[7] C. Bliek, P. Spellucci, L.N. Vicente, A. Neumaier, L. Granvilliers, E. Monfroy, F. Benhamou, E.Huens, P. van Hentenryck, D. Sam-Haroud, and B. Faltings. Algorithms for solving nonlinear constrained and optimization problems: The state of the art. Technical report, ILOG and TU Darmstad and U. of Coimbra and U. of Vienna and U. of Nantes and Catholic U. of Louvain and Swiss Federal I. of Technology, June 2001. 12

[8] A.F. Bobick and J.W. Davis. The recognition of human movement using temporal templates. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 23(3):257–267, March 2001. 1

[9] G. Bouchard and B. Triggs. Hierarchical part-based model for visual object categorization. Submitted to CVPR'05, October 2004. 9

[10] Richard P. Brent. An algorithm with guaranteed convergence for finding a zero of a function. *Comput. J.*, 14(4):422–425, 1971. 12, 115

[11] R. Brooks and T. Arbel. Generalizing inverse compositional image alignment. In *Int'l Conf. on Pattern Recognition*, volume II, pages 1200–1203, Hong Kong, China, August 2006. 120

[12] M. Brown, R. Szeliski, and S. Winder. Multi-image matching using multi-scale orientated patches. In *Proc. IEEE Int'l Conf. on Computer Vision and Pattern Recognition*, San Diego, CA, USA, June 2005. 9

[13] M.C. Burl, M. Weber, and P. Perona. A probabilistic approach to object recognition using local photometry and global geometry. In *Proc. of 5th European Conference on Computer Vision*, University of Freiburg, Germany, June 1998. 9

[14] Hua-Mei Chen and P.K. Varshney. Mutual information-based CT-MR brain

image registration using generalised partial volume joint histogram estimation. *IEEE. Trans. Medical Imaging*, 22(9):1111–1119, September 2003. 17, 27, 45, 58, 61, 62, 87, 104

[15] O. Chum and J. Matas. Matching with prosac - progressive sample consensus. In *Proc. Computer Vision and Pattern Recognition*, San Diego, CA, USA, June 2005. 9

[16] D. W. Clark, C. Mohtadi, and P. S. Tuffs. Generalized predictive control - part i. the basic algorithm; part ii. extensions and interpretations. *Automatica*, 23(2):137–160, 1987. 129

[17] A. Collignon, F. Maes, D. Delaere, D. Vandermeulen, P. Suetens, and G. Marchal. *Information Processing in Medical Imaging*, chapter Automated Multi-modality image registration based on information theory, pages 263–374. Kluwer Academic, 1995. 9, 15, 17

[18] R.T. Collins, Y. Liu, and M. Leordeanu. Online selection of discriminative tracking features. *IEEE Trans. PAttern Analysis and Machine Intelligence*, 27(10):1631–1643, October 2005. 9

[19] T.F. Cootes, G.J. Edwards, and C.J. Taylor. Active appearance models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(6):681–685, June 2001. 1

[20] T. Cover and J. Thomas. *Elements of Information Theory*. Wiley & Sons, New York, 1991. 16

[21] W. R. Crum, D. L. G. Hill, and D. J. Hawkes. Information theoretic similarity measures in non-rigid registration. In *Information Processing in Medical Imaging*, pages 378–387, 2003. 18, 31

[22] E. D'Agostino, D. Vandermeulen F. Maes, and P. Seutens. A viscous fluid model for multimodal non-rigid image registration using mutual information. *Medical Image Analysis*, 7:565–575, 2003. 13

[23] A.J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *Proc. 9th IEEE Int'l Conf. on Computer Vision*, pages 1403–1410, Nice, France, October 2003. 1

[24] G. Dorko and C. Schmid. Object class recognition using discriminative local features. Submitted to IEEE Trans. on Pattern Analysis and Machine Intelligence, January 2005. 9

[25] C.H. Esteban and F. Schmitt. Silhouette and stereo fusion for 3D object modeling. *Computer Vision and Image Understanding*, 96:367–392, August 2004. 9

[26] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 264–271, June 2003. 9

[27] E. Hadjidemetriou, M.D. Grossberg, and S.K. Nayar. Multiresolution histograms and their use for recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(7):831–847, July 2004. 25, 87

[28] G.D. Hager and P.N. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(10):1025–1039, October 1998. 10, 14, 114

[29] C. Harris and M. Stephens. A combined corner and edge detector. In *Proc. of 4th ALVEY vision conference*, pages 147–151, September 1988. 19

[30] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004. 13

[31] Michael Isard and Andrew Blake. Condensation – conditional density propagation for visual tracking. *Int. J. Computer Vision*, 29(1):5–28, 1998. 2, 8

[32] M. Jenkinson and S. Smith. A global optimisation method for robust affine registration of brain images. *Medical Image Analysis*, 5:143–156, 2001. 17, 142

[33] A.D. Jepson, D.J.Fleet, and T.F. El-Maraghi. Robust online appearance models for visual tracking. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(10):1296–1311, October 2003. 9

[34] Jim Xiuquan Ji, Hao Pan, and Zhi-Pei Liang. Furthern analysis of interpolation effects in mutual information-based image registration. *IEEE. Trans. Medical Imaging*, 22(9):1131–1140, September 2003. 3, 19, 48, 62, 87

[35] Andrew E. Johnson and Martial Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):433–449, 1999. 2

[36] T. Kadir and M. Brady. Saliency, scale and image description. *Int'l Journal of Computer Vision*, 45(2):83–105, November 2001. 20, 88, 102, 148

[37] T. Kadir and M. Brady. Estimating statistics in arbitrary regions of interest. In W.F. Clocksin, A.W. Fitzgibbon, and P.H.S. Torr, editors, *Proc. British Machine Vision Conf.*, volume 2, pages 589–598, Oxford, 2005, September 2005. 4, 48, 88, 101, 103

[38] T. Kadir and M. Brady. Non-parametric estimation of probability distributions from sampled signals. Technical report, Robotics Research Lab, Oxford, UK, July 2005. 88

[39] T. Kadir, A. Zisserman, and M. Brady. An affine invariant salient region detector. In *Proc. 8th European Conf. On Computer Vision*, volume 1, pages 228–241, Prague, May 2004. 20

[40] T. Kaneko and O. Hori. Template update criterion for template matching of image sequences. In *Proc. Int'l Conf. on Pattern Recognition*, volume 2, pages 1–5, 2002. 9, 14, 128

[41] T. Kirishima, K. Sato, and K. Chihara. Real-time gesture recognition by learning and selective control of visual interest points. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(3):351–364, March 2005. 1

[42] M.E. Leventon and W.E.L. Grimson. Multi-modal volume registration using joint intensity distributions. In *Proc. Of Conf. On Medical Image Computing and Computer Assisted Intervention*, 1998. 9

[43] Anat Levin and Yair Weiss. Learning to combine bottom-up and top-down segmentation. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision – ECCV 2006*, volume 3954 of *LNCS*, pages 581–594. Springer, 2006. 15

[44] D.G. Lowe. Distinctive image features from scale-invariant keypoints. *Int'l J. of Computer Vision*, 60(2):91–110, 2004. 2, 20

[45] B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. 7th Int'l Joint Conf. on Artificial Intelligence*, pages 674–679, Vancouver, Canada, August 1981. 10, 14, 113, 116, 127

[46] F. Maes. *Segmentation and registration of multimodal medical images: From theory, implementation and validation to a useful tool in clinical practice.* PhD thesis, Dept. Elect. Eng. (ESAT/PSI), KU Leuven, Leuven, Belgium, 1998. 18

[47] F. Maes, A. Collignon, D. Vandermeulen, G. Marchal, and P. Suetens. Multimodality image registration by maximisation of mutual information. *IEEE. Trans. On Medical Imaging*, 16(2):187–198, April 1997. 3, 17, 25, 45, 51, 61

[48] F. Maes, D. Vandermeulen, and P. Seutens. Comparative evaluation of multiresolution optimization strategies for multimodalitiy image registration by maximization of mutual information. *Medical Image Analysis*, 3(4):272–286, April 1999. 17, 18, 19, 75

[49] F. Maes, D. Vandermeulen, and P. Suetens. Medical image registration using mutual information. *Proc. of the IEEE*, 2003. 87

[50] D.W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11(2):431–441, June 1963. 12, 115

[51] J. Matas and S. Obdrzalek. Learning parameters of a recognition system based on local affine frames. In *Cognitive Vision Workshop*, Zurich, Switzerland, September 2002. 9, 20

[52] I. Matthews, T. Ishikawa, and S. Baker. The template update problem. In *Proc. 14th British Machine Vision Conference*, University of East Anglia, Norwich, UK, September 2003. 9

[53] I. Matthews, T. Ishikawa, and S. Baker. The template update problem.

IEEE Trans. on Pattern Analysis and Machine Intelligence, 26(6):810–815, June 2004. 10, 123, 128

[54] C. Meyer, J.L. Boes, B. Kim, R.L. Bland, P.H. Wahl, K.R. Zasadny, P.V. Kison, K. Koral, and K.A. Frey. Demonstration of accuracy and clinical versatility of mutual information for automatic multimodality image fusion using affine and thin plate spline warped geometric deformations. *Medical Image Analysis*, 1(3):195–206, April 1997. 17

[55] A. Micilotta, E.J. Ong, and R. Bowden. Real-time upper body detection and 3d pose estimation in monoscopic images. In *Proc. European Conference on Computer Vision*, Graz, Austria, May 2006. 7

[56] R. Moddemeijer. On estimation of entropy and mutual information of continuous distributions. *Signal Processing*, 16:233–248, 1989. 19, 48, 63, 87, 97

[57] H.Y. Nguyen and A.W.M. Smeulders. Fast occluded object tracking by a robust appearance filter. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(8):1099–1104, August 2004. 9, 10

[58] A. Nuchter, H. Surmann, and J Hertzberg. Planning robot motion for 3d digitisation of indoor environments. In *Proc. 11th Int'l Conf. on Advanced Robotics*, 2003. 1

[59] S. Obdrzalek and J. Matas. Object recognition using local affine frames on distinguished regions. In *Proc. 13th British Machine Vision Conference*, Cardiff, UK, September 2002. 20

[60] A.S. Ogale, C. Fermuller, and Y. Aloimonos. Motion segmentation using occlusions. *IEEE Trans. PAttern Analysis and Machine Intelligence*, 27(6):988–992, June 2005. 9

[61] K. Okuma, A. Taleghani, N. de Freitas, J. Little, and D. Lowe. A boosted particle filter: multitarget detection and tracking. In *Proc. 8th European Conf. On Computer Vision*, volume 1, pages 28–39, Prague, May 2004. 7, 9

[62] Athanasios Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill Inc., 3rd edition, 1991. 90, 95

[63] E. Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, September 1962. 17, 45

[64] J. Paterson and A. Fitzgibbon. 3D head tracker using non-linear optimization. In *Proc. 14th British Machine Vision Conference*, University of East Anglia, Norwich, UK, September 2003. 1

[65] J. Pilet, V. Lepetit, and P. Fua. Real-time nonrigid surface detection. In *Int'l Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 822–828, San Diego, CA, USA, 2005. 13

[66] J.P.W. Pluim, J.B.A. Maintz, and M.A. Viergever. Interpolation artefacts in mutual information-based image registration. *Computer Vision And Image Understanding*, 77:211–232, 2000. 3, 18, 19, 44, 51, 56, 61, 75, 88, 106

[67] J.P.W. Pluim, J.B.A. Maintz, and M.A. Viergever. Mutual-information-based registration of medical images: A survey. *IEEE Trans. Medical Imaging*, 22(8):986–1003, August 2003. 17

[68] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical Recipes in C*. Cambridge University Press, 2nd edition, 1992. 12, 104, 115

[69] A. Rajwade, A. Banerjee, and A. Rangarajan. New method of probability density estimation with application to mutual information based image registration. In *Computer Vision and Pattern Recognition*, volume 2, pages 1769–1776, New York, USA, June 2006. 89

[70] D. Russakoff, C. Tomasi, T. Rohlfing, and C. Maurer. Image similarity using mutual information of regions. In *Proc. 8th European Conference on Computer Vision*, volume 3, pages 596–607, Prague, May 2004. 3

[71] C.E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 623–656, July, October 1948. 16

[72] Jianbo Shi and Carlo Tomasi. Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition*, Seattle, June 1994. 148

[73] H.-Y. Shum and R. Szeliski. Systems and experiment paper: Construction of panoramic image mosaics with global and local alignment. *International Journal of Computer Vision*, 36(2):63–84, 2000. 114

[74] P. Smith, T Drummond, and R Cipolla. Layered motion segmentation and depth ordering by tracking edges. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(4):479–494, April 2004. 1

[75] R.A. Smith, A.W. Fitzgibbon, and A. Zisserman. Improving augmented reality using image and scene constraints. In *Proc. 10th British Machine Vision Conference*, Nottingham, UK, September 1999. 2

[76] S.M. Smith and J.M. Brady. Susan - a new approach to low level image processing. *International Journal of Computer Vision*, 23:45–78, 1997. 20

[77] C. Studholme, D. Hill, and D. Hawkes. Automated 3d registration of trun-

cated mr and ct images of the head. In *Proc. British Machine Vision Conference*, pages 27–36, September 1995. 9, 15, 142

[78] C. Studholme, D.LG. Hill, and D.J. Hawkes. Automated 3-d registration of mr and ct images of the head. *Medical Image Analysis*, 1(2):163–175, June 1996. 17

[79] D. Terzopoulos and R. Szeliski. Tracking with kalman snakes. In A. Blake and A. Yuille, editors, *Active Vision*. MIT, 1992. 2

[80] A. Thayananthan, B. Stenger, P. Torr, and R. Cipola. Shape context and chamfer matching in cluttered scenes. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 127–133, Madison, USA, June 2003. 1, 2

[81] P. Thevenaz and M. Unser. Optimization of mutual information for multi-resolution image registration. *IEEE Trans. On Image Processing*, 9(12):2083–2099, December 2000. 17, 18, 21, 29, 45, 61, 62, 87, 104

[82] C. Tomasi and T. Kanade. Detection and tracking of point features. Technical Report CMU-CS-91-132, CMU, April 1991. 19

[83] P.H. Torr, A.W. Fitzgibbon, and A. Zimmerman. The problem of degeneracy in structure and motion recovery from uncalibrated image sequences. *Int'l. J. Computer Vision*, 31(1):27–44, 1999. 1

[84] L. Torresani and A. Hertzmann. Automatic non-rigid 3d modeling from video. In *Proc. 8th European Conf. On Computer Vision*, volume 2, pages 299–312, Prague, May 2004. 13

[85] Lorenzo Torresani, Aaron Hertzmann, and Christoph Bregler. Learning non-rigid 3d shape from 2d motion. In Sebastian Thrun, Lawrence Saul, and

Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004. 1

[86] K. Toyama and A. Blake. Probabilistic tracking in a metric space. In *Proc. 8th IEEE Int'l Conf. on Computer Vision*, volume 2, pages 50–57, Vancouver, July 2001. 1, 9

[87] B. Triggs. Empirical filter estimation for subpixel interpolation and matching. In *Proc. Int. Conf. on Computer Vision*, pages 550–557, 2001. 93

[88] J. Tsao. Interpolation artifacts in multimodality image registration based on maximisation of mutual information. *IEEE. Trans. Medical Imaging*, 22(7):854–864, July 2003. 19, 61, 75

[89] Yaron Ukrainitz and Michal Irani. Aligning sequences and actions by maximizing space-time correlations. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision – ECCV 2006*, volume 3953 of *LNCS*, pages 538–550. Springer, 2006. 15

[90] M. Unser, A. Aldroubi, and M. Eden. B-spline signal processing: Part i–theory. *IEEE. Trans. Signal Processing*, 41(2):821–833, February 1993. 24, 28

[91] P. Viola and M.J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, May 2004. 7

[92] P. Viola and W.M. Wells. Alignment by maximization of mutual information. In *Proc. Int'l Conf. on Computer Vision*, pages 16–23, Boston, MA, USA, June 1995. 9, 15

[93] P. Viola and W.M. Wells. Alignment by maximization of mutual information. *Int'l Journal of Computer Vision*, 24(2):137–154, 1997. 16

[94] P.A. Viola. *Alignment by Maximisation of Mutual Information*. PhD thesis, Massachussets Institute of Technology, 1995. 45

[95] G. Welch and E. Foxlin. Motion tracking: no silver bullet, but a respectable arsenal. *IEEE Computer Graphics and Applications*, 22(6):24–38, November 2002. 1

[96] W.M. III Wells, P. Viola, H. Atsumi, S. Nakajima, and R. Kikinis. Multi-modal volume registration by maximization of mutual information. *Medical Image Analysis*, 1(1):35–51, March 1996. 17

[97] J. West, J. M. Fitzpatrick, and M. Y. Wang et.al. Comparison and evaluation of retrospective intermodality brain image registration techniques. *J. Comput. Assisted Tomography*, 21(4):554–566, 1997. 16

[98] Y. Wu, T. Yu, and G. Hua. Tracking appearances with occlusions. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 789–795, June 2003. 9

[99] J. Xiao, S. Baker, I. Matthews, and T. Kanade. Real-time combined 2D+3D active appearance models. In *Proc. Conf. on Computer Vision and Pattern Recognition*, June 2004. 14

[100] Alper Yilmaz, Xin Li, and Mubarak Shah. Contour-based object tracking with occlusion handling in video acquired using mobile cameras. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(11):1531–1536, November 2004. 2, 9

[101] X.S. Zhou, D. Comaniciu, and A. Gupta. An information fusion framework for robust shape tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(1):115–129, January 2005. 2