Priors for New View Synthesis

D.Phil Thesis

Robotics Research Group
Department of Engineering Science
University of Oxford

Supervisors:
Doctor Andrew W. Fitzgibbon
Doctor Ian D. Reid
Professor Philip H. S. Torr

Oliver J. Woodford
New College

August 16, 2009

Abstract

New view synthesis (NVS) is the problem of generating a novel image of a scene, given a set of calibrated input images of the scene, *i.e.* their viewpoints, and also that of the output image, are known. The problem is generally ill-posed—a large number of scenes can generate a given set of images, therefore there may be many equally likely (given the input data) output views. Some of these views will look less natural to a human observer than others, so prior knowledge of natural scenes is required to ensure that the result is visually plausible. The aim of this thesis is to compare and improve upon the various *Markov random field* and *conditional random field* prior models, and their associated *maximum a posteriori* optimization frameworks, that are currently the state of the art for NVS and stereo (itself a means to NVS).

A hierarchical example-based image prior is introduced which, when combined with a multi-resolution framework, accelerates inference by an order of magnitude, whilst also improving the quality of rendering.

A parametric image prior is tested using a number of novel discrete optimization algorithms. This general prior is found to be less well suited to the NVS problem than sequence-specific priors, generating two forms of undesirable artifact, which are discussed.

A novel pairwise clique image prior is developed, allowing inference using powerful optimizers. The prior is shown to perform better than a range of other pairwise image priors, distinguishing as it does between natural and artificial texture discontinuities.

A dense stereo algorithm with geometrical occlusion model is converted to the task of NVS. In doing so, a number of challenges are novelly addressed; in particular, the new pairwise image prior is employed to align depth discontinuities with genuine texture edges in the output image. The resulting joint prior over smoothness and texture is shown to produce cutting edge rendering performance.

Finally, a powerful new inference framework for stereo that allows the tractable optimization of second order smoothness priors is introduced. The second order priors are shown to improve reconstruction over first order priors in a number of situations.

This thesis is submitted to the Department of Engineering Science, University of Oxford, in fulfilment of the requirements for the degree of Doctor of Philosophy. This thesis is entirely my own work and, except where otherwise stated, describes my own research.

Oliver Woodford
New College

Acknowledgements

I owe a huge debt of thanks to the many people who have patiently supported me on this long journey.

To my excellent supervisors, Andrew Fitzgibbon, Ian Reid and Phil Torr, for their guidance and inspiration, and for letting me find my own way.

To members of the Visual Geometry Group, and the wider Robotics Research Group, past and present who have helped me both in my research and provided me with such an enjoyable and stimulating place to work. There are too many to name, but Andrew Zisserman, Mark Everingham, Nick Apostoloff, David Claus, Jamie Paterson, Josef Sivic, Aeron Buchanan and Ali Shahrokni deserve special mention for their generous assistance, as do Mukta Prasad and Maria-Elena Nilsback for accompanying me throughout my journey. Thanks also to the many visiting academics who have listened about and given feedback on my research.

To Pushmeet Kohli and everyone else in the Oxford Brookes Computer Vision group for their help and camaraderie.

To Vladimir Kolmogorov and Carsten Rother for being so generous with their software.

To Graham Jones and Lyndon Hill of Sharp Labs Europe, whose sponsorship through a CASE studentship made my research possible, and to the Engineering and Physical Sciences Research Council for their funding.

I am grateful to all those people who have inspired my interest in Science and Engineering, especially Tom Drummond, who introduced me to Computer Vision through an exciting Masters project.

Finally to my family and all my friends for their unwavering support. I especially thank Rachael for her companionship, and my parents, for making me the man I am today.

Contents

Nomenclature

Notation

Notation	Meaning
a, A, μ	*Italic Serif* typeface indicates a scalar variable, or a function which returns a scalar variable. N is the number of input images, excluding the reference image in stereo. $p(\cdot)$ returns a probability.
$\mathbf{a}$, $\mathbf{A}$, $\boldsymbol{\mu}$	**Bold Serif** typeface indicates vector (columnwise unless otherwise stated). A_i references the *i*th element of $\mathbf{A}$. $\mathbf{x}$, $\mathbf{p}$, $\mathbf{q}$, $\mathbf{r}$, and $\mathbf{s}$ represent 2-d locations in an image, pixel centres being at integer locations.
$\mathring{\mathbf{A}}$	A homogenized vector, such that $\mathring{\mathbf{A}} = \begin{bmatrix} \mathbf{A} \\ 1 \end{bmatrix}$.
$\mathtt{A}$	`Typewriter` typeface indicates a matrix. $\mathtt{\Sigma}$ is a covariance matrix. $\mathtt{I}$ is the identity matrix. $\mathtt{P}$ is a 3×4 projection matrix.
$\mathcal{A}$	*Calligraphic* typeface indicates a data structure or set. $\mathcal{N}$ is a set 2-d locations of pixels in a neighbourhood or clique. $\mathcal{X}$ is the set of all pixels in an image.
$\mathbb{A}$	$\mathbb{Blackboard}$ typeface also indicates a data structure or set. $\mathbb{N}$ is a set of neighbourhoods. $\mathbb{T}$ is a set of image patch exemplars.
A	Sans Serif typeface indicates a 2-d image-sized array with a scalar value per image pixel, *e.g.* a dense label or depth map. D is a disparity map, for which $D(\mathbf{x})$ is the value at $\mathbf{x}$.
$\boldsymbol{\mathsf{A}}$	**Bold Sans Serif** typeface indicates a 3-d image-sized array with a vector of values per pixel, *e.g.* an RGB image. $\boldsymbol{\mathsf{I}}^*$ is the noiseless version of $\boldsymbol{\mathsf{I}}$, the latter being a real image. $I(\mathbf{x})$ and $I(\mathcal{N})$ represent the image values for the location $\mathbf{x}$ and the neighbourhood $\mathcal{N}$ respectively; $\overrightarrow{I(\mathcal{N})}$ is a vectorized, *i.e.* columnated, version of $I(\mathcal{N})$. The colour at non-integer locations of images is bilinearly interpolated.
$\pi_i(\mathbf{x}, d)$	The projection function, which defines the projection into the *i*th input image, $\boldsymbol{\mathsf{I}}_i$, of a pixel at location $\mathbf{x}$ and disparity d in the output or reference image.
ζ	The partition function: the area of an unnormalized probability distribution.
ψ, ϕ	Clique functionals of an MRF, the former in a probability, the latter in an energy, *i.e.* $\phi(\cdot) = -\log \psi(\cdot)$.
$\lvert \cdot \rvert$	The magnitude or vector length operator. For a scalar, this returns the magnitude. For any other variable, this returns the number of elements in that variable.
$\lVert \cdot \rVert_\alpha$	The Lα operator: $\lVert \mathbf{X} \rVert_\alpha = \sqrt[\alpha]{\sum_i X_i^\alpha}$. If α is absent it is the L2 operator.
$[\,\cdot\,]$	The Iverson bracket: $[statement] = 1$ if $statement$ is true, 0 otherwise.
$\mathcal{O}(\,\cdot\,)$	Order of magnitude or computational complexity.

Acronyms

Acronym	Meaning
n-d	n-dimensional.
BP	Belief propagation.
CRF	Conditional random field.
FoE	Field of experts.
ICM	Iterated conditional modes.
ILS	Iterated local search.
MC	Monte Carlo—a sampling approach.
MRF	Markov random field.
NVS	New view synthesis.
PCA	Principal components analysis.
PDE	Partial differential equation.
PoE	Product of experts.
RGB	Red, green, blue—a 3-d colour space.
RMS	Root mean squared.
SA	Simulated annealing.
SSD	Sum of squared differences.
TRW	Tree-reweighted message passing.
YCbCr	A 3-d colour space with one channel of intensity and two channels of chrominance, often used in image and video compression.

Chapter 1

Introduction

The aim of this thesis is to present research investigating the efficacy of “priors”, both old and new, in improving solutions to the problems of new view synthesis (NVS) and stereo, the latter being an intermediate step in some NVS approaches. This chapter gives a gentle introduction to the NVS problem, what priors are, and why they are needed in this problem. Finally, a detailed outline of the rest of this thesis is given.

1.1 New view synthesis

New view synthesis is the generation of a novel view (image) of a scene, given a set of other images of that scene, *i.e.* images with viewpoints different from that of the view one wishes to reconstruct. In addition to a set of input images, one needs to know the viewpoints of these input images, plus some other details of camera calibration discussed in appendix A, relative to the output image’s viewpoint. The problem is summarized in figure 1.1.

A calibrated camera allows one to determine the exact path in space of the light ray that was (or would be, for the new view) measured at a given pixel of the image. For example, the ray that would be measured by the pixel denoted $\mathbf{x}$ in figure 1.1 is marked by the red line passing through that pixel. If all the rays required to construct the new view are collinear with rays that were measured in the input images then the output image can be constructed directly from the input data. However, this situation rarely arises without careful premeditation of the placement of input and output views.

In this thesis I assume the general case, where output rays are not collinear with input rays. In this case, each output ray generates a line when projected onto each input image, as shown in figure 1.1.

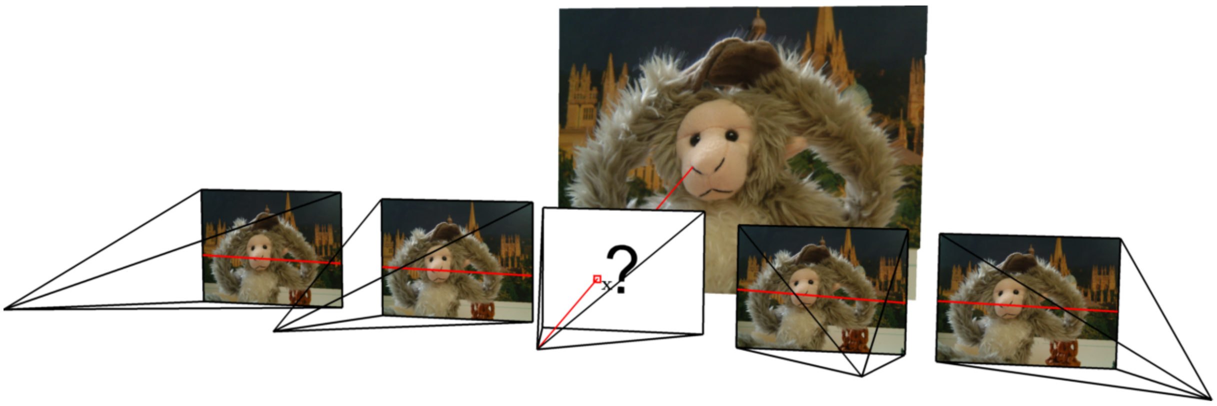


Figure 1.1: NVS problem. A visual representation of the NVS problem—several images of a scene are given, whose viewpoints are known, and the viewpoint of a new, unknown image is also given. The aim is to reconstruct the image of the scene from that new viewpoint using the images provided. The ray measured by an output pixel, $\mathbf{x}$, is shown in red, as are the resulting epipolar lines on the input images.

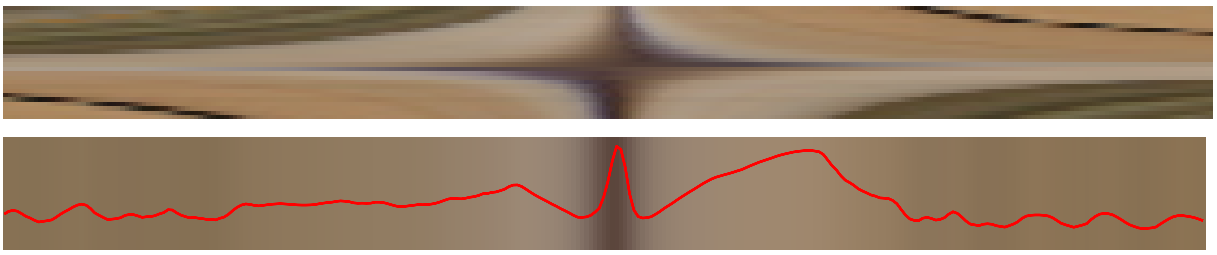


Figure 1.2: Photoconsistency. This figure demonstrates how the colour of the pixel at $\mathbf{x}$ in figure 1.1 is chosen using *photoconsistency*. *Top row*: Epipolar lines are sampled, at a range of depths along the epipolar ray, across all (here, 26) input images, and stacked up row-wise, *i.e.* depth is along the horizontal axis and input image along the vertical axis. *Bottom row*: The mean colour at each depth is computed, as is a measure of how consistent the input samples at a given depth are with the mean colour at that depth, indicated by the red line (higher is more consistent). The most likely colour is that which gives the highest photoconsistency.

These lines are known as *epipolar* lines. The problem is therefore constrained to finding the colour of each output pixel, based on the colour along the epipolar line in each of the input images.

1.1.1 The inverse problem

In order to infer the colour of an output pixel from a set of image samples along epipolar lines, one must first infer something about the scene that created the input data, $\mathcal{D}$ (the input images). This scene model, $\mathcal{S}$, can then be used to generate the output image. Such a model might include the geometry and reflectance properties of objects, the colour and position of light sources, even the refractive nature of translucent solids and fluids. Whatever the form of scene model used, the task of inferring the model parameters from input data is known as an inverse problem, written as

$$f(\mathcal{S}) = \mathcal{D} \tag{1.1}$$

where $f(\cdot)$ is the function or process that generates the input data from the scene model, and which needs to be inverted. This function includes a model of the data measurement process, which can be assumed to be noisy, *i.e.* imperfect. The noise model can be used to compute the likelihood of the measured data, given a scene model, which is known as the *data likelihood* and written as $p(\mathcal{D}|\mathcal{S})$.

The *maximum likelihood* (ML) scene, $\mathcal{S}_{\mathrm{ML}}$, is that which maximizes the likelihood of the data, *i.e.*

$$\mathcal{S}_{\mathrm{ML}} = \underset{\mathcal{S}}{\operatorname{argmax}}\, p(\mathcal{D}|\mathcal{S}). \tag{1.2}$$

Data likelihood is usually based on the *photoconsistency* measure [SD97], which assumes that surfaces are Lambertian, *i.e.* the same colour when viewed from any angle. Figure 1.2 shows how the most photoconsistent (therefore the ML) colour is computed, for pixel $\mathbf{x}$ from figure 1.1. First, the input images are sampled at a range of depths along the epipolar ray. Then the most photoconsistent colour at each depth is computed, along with the photoconsistency of that colour; the method for doing this depends on the noise model, but the mean colour is used here. The most photoconsistent colour is then selected from all depths. Note that in figure 1.2(*bottom*), two very different colours vie for the label of most photoconsistent.

The ML solution is typically used when the problem is well-posed. Hadamard [Had02] states that a problem is well-posed if

1. a solution exits.
2. the solution is unique.
3. the solution depends continuously on the data.

However, as with most inverse problems, NVS is typically *ill*-posed—for a given set of input images there can be many scene models which would produce that set [PTK85], and it can be seen from figure 1.2 that a small change to the input images could change the most likely colour completely. It therefore violates the second and third of Hadamard's requirements.

1.1.2 Regularization using priors

A common approach to solving ill-posed problems is to regularize them by adding a term to the problem which reduces ambiguity and improves robustness. Such a term can also encourage the solution to be more likely *a priori*, which is to say more likely given what is known about the expected form of solutions in general, based on prior knowledge. This kind of term or model is called a "prior"—developing and studying the efficacy of a variety of such models in a variety of optimization frameworks is the main aim of this thesis.

Prior knowledge can be incorporated into the problem using Bayes' rule, which defines the *posterior* probability of, in this case, the scene as

$$p(\mathcal{S}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathcal{S})p(\mathcal{S})}{p(\mathcal{D})}, \tag{1.3}$$

based on the data likelihood, and the a priori probabilities of the scene, $p(\mathcal{S})$, and the input data, $p(\mathcal{D})$. The standard approach is to find the most likely scene under this distribution, which is called the *maximum a posteriori* (MAP) solution, $\mathcal{S}_{\text{MAP}}$, thus

$$\mathcal{S}_{\text{MAP}} = \underset{\mathcal{S}}{\operatorname{argmax}}\, p(\mathcal{S}|\mathcal{D}). \tag{1.4}$$

An example of the improvement in image quality the MAP solution provides over the ML solution is given in figure 1.3. Note that, while more plausible as a natural image, the MAP solution is not guaranteed to be a better reconstruction of the true scene.

1.2 Motivation

I will motivate two things: first, the need for priors in regularizing the NVS problem; second, the need for NVS algorithms as a whole. NVS, as an application, cannot fulfil its aim if the images generated are not believable, and as figure 1.3 shows, a prior can make a huge improvement to the visual plausibility of an image—the high-frequency artifacts of the ML image immediately mark it out to the human observer as being artificial, while a cursory glance over the MAP image could easily leave the viewer thinking it was a natural image.

What, then, are the applications for visually plausible NVS? Possible applications include:

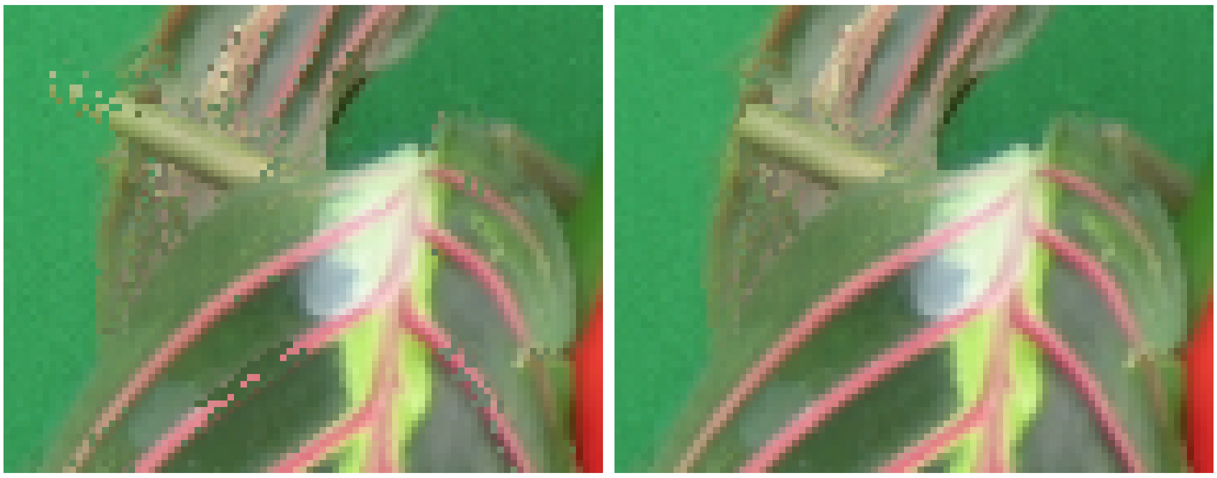


Figure 1.3: ML *vs*. MAP solutions. Equivalent images for a given sequence rendered from (a) the ML scene model, and (b) the MAP scene model using the Sampled prior described in chapter 6. *The image is very close to, but not quite the optimal solution.

- Generating photorealistic videos of paths through buildings and outdoor scenes from a few photos, for use in computer games, film post-production or architectural applications.
- For use in multi-camera setups filming one-off events, to move the viewpoint to any desired position, *e.g.* in sports, especially for aiding refereeing decisions; film production, especially for determining camera trajectory around a costly shot, such as an explosion, in post-production.
- Generating a second, stereoscopic view of a scene for each frame in a video, to view old, 2-d film-footage on a 3-d display.

1.3 Challenges

The challenges to regularizing the NVS problem are two fold—developing a prior that 1) accurately models the relevant statistics of natural scenes or images, but that also 2) yields a tractable problem for which the MAP solution, or a good approximation thereof[1], can be found.

The prior model must be able to overcome the challenges presented by the data, namely the fact that some scene points may be occluded in some of the input images, reducing the photoconsistency of the

[1]In this thesis "MAP solution" will generally refer to that solution with the highest MAP probability found, which might not be a global optimum of this probability.

true colour, and the multi-modal nature of photoconsistency over depth and colour, which can occur for two main reasons:

- A fine textural detail is surrounded by a region of homogenous colour, such that both the fine detail and the homogenous colour generate modes (as per the example pixel, $\mathbf{x}$, shown in figures 1.1 & 1.2).
- Scene points at two different depths along the same output ray are visible in most or all of the input images, *e.g.* the edge of a foreground object in front of a background object.

These problems can generate both small and large scale errors in the ML image (as seen in figure 1.3(a)), which the prior must be able to overcome.

Priors for NVS usually work on small neighbourhoods, called *cliques*, of pixels. The larger these cliques are, the less tractable the optimization problem, or at least finding a good solution to the problem, becomes. However, the smaller these cliques are, the less information there is with which to distinguish between natural texture and artifacts. There is therefore a fine balance to be made between ability of a prior to discriminate, and the tractability of the resulting optimization problem.

1.4 Contributions

I will now briefly outline the contributions made in this thesis; a more detailed account of the contributions appears in §9.2. The contributions can be split into three main categories, summarized below.

1.4.1 A comparison of priors and optimizers

Each chapter applies one or more different forms of prior to the NVS problem. In addition, one or more optimizers are used to maximize the resulting posterior probability. Each chapter gives a detailed analysis of the relative performance of the priors and/or optimizers used in that chapter. The final chapter then compares the best algorithm from each chapter. This informs the reader as to which priors and associated optimizers are most suited to the task of NVS.

1.4.2 New prior models

In addition to employing image priors from the literature, sometimes applied to the NVS problem for the first time, new prior models are developed specifically for the NVS problem. In chapter 6 a texture prior that operates on two-pixel cliques is developed, which is able to distinguish between real and artificial texture discontinuities. In chapter 7 this prior is combined with a prior on the smoothness of the scene, creating a joint prior on smoothness and texture. This latter prior is shown to produce cutting edge results in reconstruction performance.

1.4.3 New optimization techniques

Several new optimization techniques are developed to minimize energies of both existing and new prior models. In particular

- Chapter 4 introduces a hierarchical, exemplar library tree structure which, when intertwined with a multi-resolution framework, speeds up optimization by a factor of around 30, whilst also improving the output solution.
- Chapter 5 introduces an extension of Iterated Conditional Modes (ICM), Greedy ICM, and a normalized cooling schedule (described in appendix B) to improve the efficiency of Simulated Annealing (SA). Iterated Local Search (ILS) is also tested, with two novel perturbation schemes, and a binary optimizer called "optimal splice" is developed to combine the resulting local minima in a fusion move framework.
- Chapter 6 introduces a fast, deterministic method for computing modes of photoconsistency over colour.
- Chapter 7 develops an asymmetrical, geometrical occlusion model which contains only pairwise cliques, and a fusion move framework to optimize the resulting problem. Quadratic Pseudo-Boolean Optimization (QPBO) is used as the binary optimizer, and two novel extensions to this are made which improve optimization performance. Triple cliques are introduced in order to simultaneously optimize over depth and colour, and I show that these can be decomposed into a set of pairwise cliques regardless of the form of clique functional.
- Chapter 8 creates an optimization framework for stereo that allows any ad hoc depth map proposals to be combined in a principled way using both first and second order smoothness priors, with a geometrical visibility model. Three example methods for generating depth map proposals are given.

1.5 Thesis outline

Chapter 2 describes the NVS problem in detail, defining the inputs and discussing the various scene models available. Regularization is explained in a Bayesian framework, and the form and learning of prior models discussed. The data likelihood term and its simplifying assumptions are described.

In chapter 3 I review the literature relevant to the work presented in this thesis, focussing first on optimization techniques for inference, followed by the forms of statistical models currently used for geometry and then image priors. I pay particular attention to those techniques and models which will be employed and extended in this work.

Chapter 4 investigates improvements to the efficiency and efficacy of the example-based image prior framework of Fitzgibbon *et al.* [FWZ05], which uses 5×5 pixel cliques. Three different ICM-based optimization strategies are tested in a single resolution framework. A hierarchical, exemplar library tree structure, which exploits the highly dependent nature of neighbouring pixels and also image patches at different scales, is then created and embedded in a multi-resolution framework in order to accelerate and improve the rendering. A detailed analysis of the improvements is given.

The Field of Experts (FoE) prior of Roth & Black [RB05], which again uses 5×5 pixel cliques, is tested on the NVS problem in chapter 5. A discrete optimization framework is used, and several different optimizers, which extend ICM, SA and ILS, are developed and tested. Experiments highlight two properties of the prior undesirable for NVS, and these are discussed.

Chapter 6 marks a shift in the thesis to priors with smaller cliques, which are therefore easier to optimize well. A fast, deterministic method for computing modes of photoconsistency over colour is developed, to reduce the number of labels per pixel. Three different priors from the literature—the sparse derivative prior of [TRF03], the global example-based prior of [FWZ05], and the local example-based prior of [CB04]—are compared with a new example-based prior with discriminative local libraries on the NVS problem.

In chapter 7 I regularize the implicit geometry of the scene, using priors from stereo[2] to encourage surface smoothness. An objective function with geometrical visibility reasoning and modulation of the smoothness prior using a pairwise texture prior, to align depth and texture discontinuities, is introduced. An optimization framework able to minimize the resulting, complex energy is developed, solving several key challenges. Experiments test the various design choices made.

[2]Stereo is the problem of inferring a dense depth map of a scene, given two or more input images.

Chapter 8 moves fully into the stereo domain, developing a global optimization framework for depth regularization using planarity preserving second order priors. The framework is based around a fusion move approach which combines ad hoc depth map proposals in a sequence of binary optimizations. Three example methods for generating depth map proposals are given. The framework is used to compare the performance of both first and second order smoothness priors, with both truncated linear and truncated quadratic priors. The resulting depth maps are then used to generate new views.

The thesis concludes in chapter 9 with a comparison of the best performing priors from chapters 4–7, a summary of the contributions made in the thesis, and a discussion of possible future avenues of research.

1.6 Publications

The work presented in chapter 4 is based on work published at BMVC'05 [WF05]. Chapter 5 describes work first published at BMVC'06 [WRTF06]. Chapter 6 presents work published at CVPR'07 [WRF07]. The work presented in chapter 7 was first published at BMVC'07 [WRTF07], but additionally incorporates improvements introduced at CVPR'08 [WTRF08]. The work presented in chapter 8 extends work which won the Best Paper Prize at CVPR'08 [WTRF08]. In addition, I co-authored a paper at ACCV'07 [SWR07] which extends the work of chapter 6 to video sequences, for temporally consistent rendering.

Chapter 2

Background

This chapter describes the new view synthesis (NVS) and stereo problems, outlines the rendering approaches used in the literature, and describes the rendering framework used throughout this thesis. The framework used is merely a means for demonstrating the efficacy of priors, and as such is kept as simple as possible. This chapter also states the assumptions and restrictions made in this simplification.

2.1 Problem statement

The problem of new view synthesis, expressed most simply, is that of determining the colour of each pixel in an output image, I_0^*, where colour is a vector in the chosen colour-space, such that it reconstructs the true view of a real-world scene.

2.1.1 Inputs

The inputs given are a set of images, $\{\mathsf{I}_i\}_{i=1}^N$, of the scene, and a set of associated projection functions, $\{\pi_i\}_{i=1}^N$. A 2-d vector, $\mathbf{x}$, denotes a pixel location in an image, I, the colour of which is written as $I(\mathbf{x})$; pixel colours at non-integer locations (*i.e.* locations not directly over the centre of a pixel) are linearly interpolated from the image throughout this thesis; locations outside the image boundaries are given a value of ∞. The projection function $\pi_i(\mathbf{x}, d)$ computes the 2-d projection into I_i of the 3-d point defined by the pixel location $\mathbf{x}$ in I_0^*, and its associated disparity, d. Appendix A gives an overview of the projective geometry used in this computation.

2.1.2 NVS approaches

Each pixel in the input and output images is generated by the ray that passes through the centre of that pixel and through the centre of projection of that image. If a ray generated by a pixel in the output image is collinear with a ray generated by an input image pixel, then, ignoring for the time being the effect of occlusions, those two pixels are viewing the same ray of light and should therefore be the same colour. Clearly if each output pixel ray is collinear with an input pixel ray then the problem is well defined, and the output image can be constructed by rearranging the input pixels.

This is the approach of light field rendering [LH96], an approach that uses a grid of uniformly sampled images to construct the 4-d plenoptic function of a scene, which then allows any view of the scene, outside its convex hull, to be reconstructed. This method effectively captures all the rich illumination of a scene and the reflectance properties of surfaces, but with the disadvantage that many carefully placed input images are required to achieve this.

If not all output pixel rays are collinear with input rays (or if input images are very noisy), then in order to reconstruct the full plenoptic function, or at least that part of it which is required, one must infer physical properties of the scene in order to create a scene model capable of generating the plenoptic function. Chai *et al.* [CTCS00] demonstrate that one can trade off sampling frequency against amount of geometrical information in generating the same plenoptic function (assuming a Lambertian scene).

This observation characterizes the spectrum of NVS methods, first described by Shum & Kang [SK00], which places NVS methods on a continuum, with no-geometry approaches such as the light field method at one end, running through to geometry focused approaches, which either construct a 3-d model of the scene and texture map it using the input images, or carve out a voxel[1] based model, *e.g.* [KS99].

What is important to note is that Chai's evaluation assumes that the geometrical information provided is accurate, but in the NVS problem as stated here, all geometrical information must be inferred from the input images. As Poggio *et al.* [PTK85] state, the problem of ascertaining geometry from a set of images is ill-posed—there are many scenes that could generate the same set of images. Successful approaches therefore assume as little geometry as possible.

In this work the sequences of images are relatively dense, such that there is a narrow baseline between neighbouring images. For such sequences there is not quite enough information to reconstruct the plenoptic function without geometry, but moving to a full 3-d model is not suitable either. For example,

[1]A voxel is a 3-d pixel, essentially a small cube in space with an associated colour.

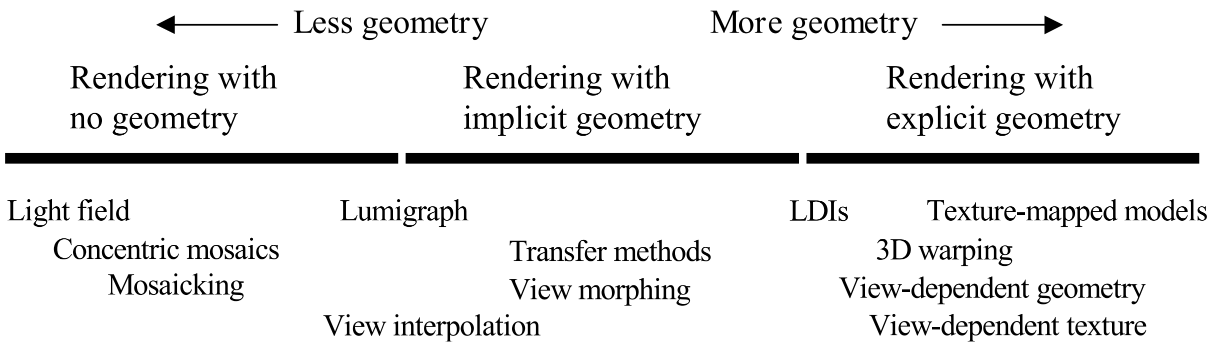


Figure 2.1: NVS continuum. A continuum of rendering frameworks for NVS, as viewed by Shum & Kang [SK00]. Figure reproduced from [SK00].

fine details such as the fur on the monkey in figure 1.1 would be heavily aliased when reconstructed in such a model using, for example, Kutulakos & Seitz's space carving approach [KS99].

Rather, I focus on a class of methods between the two extremes, known as implicit geometry methods. These methods forgo a full 3-d model, but infer some geometry in the image (*i.e.* 2-d) domain in order to transfer colour from the input to output images, and as such suffer less from aliasing issues.

Forward transfer methods employ stereo techniques to evaluate a dense depth map for one or more input images, and use this to project input pixels into output images. However, as the input pixels do not generally coincide with output pixel centres, the question of how to interpolate output pixel colours arises. Examples of forward transfer methods are given in [Sch96, ZK07].

Backward transfer methods evaluate a dense depth map for the output image and use this to sample the input images at the locations of output pixels, hence avoiding the interpolation problem of forward transfer methods. Examples of backward transfer methods are given in [IHA02, FWZ05].

This work will focus on the backward transfer method, which involves generating a disparity map, D, in addition to the output image, I_0^*, so that our scene model is $\mathcal{S} = \{\mathsf{I}_0^*, \mathsf{D}\}$. The disparity of an output pixel at image location $\mathbf{x}$ is given by $D(\mathbf{x})$. Some investigation into forward transfer methods is also included, focussing predominantly on the stereo problem of determining the depth of input pixels; the scene model is the same, but in this case one has a noisy version of the output image (I_0^*) known as the *reference* image, I_0.

2.2 Bayesian framework

Ill-posed problems such as NVS require strong regularization, based on prior knowledge of what scene models should be like—the prior.[2] The topic of this thesis is an investigation into the forms of models the prior can take. For this reason, the NVS problem is posed as one of finding the MAP solution to the following posterior probability:

$$p(\mathcal{S}|\mathbf{I}_1, .., \mathbf{I}_N) = \frac{p(\mathbf{I}_1, .., \mathbf{I}_N|\mathcal{S})p(\mathcal{S})}{p(\mathbf{I}_1, .., \mathbf{I}_N)}, \tag{2.1}$$

To simplify the problem, it is usually assumed that the input projection functions, $\{\pi_i\}_{i=1}^N$, are noiseless, hence they do not appear in the above formulation. A more advanced approach is to combine the estimation of $\mathcal{S}$ and $\{\pi_i\}_{i=1}^N$ into a single framework.

The term $p(\mathbf{I}_1, .., \mathbf{I}_N)$ depends only on input variables and is therefore constant. Our problem thus becomes one of maximizing the quasi-probability

$$\mathcal{S}_{\text{MAP}} = \underset{\mathcal{S}}{\operatorname{argmax}}\, q(\mathcal{S}|\mathbf{I}_1, .., \mathbf{I}_N) \tag{2.2}$$

$$q(\mathcal{S}|\mathbf{I}_1, .., \mathbf{I}_N) = p(\mathbf{I}_1, .., \mathbf{I}_N|\mathcal{S})p(\mathcal{S}). \tag{2.3}$$

2.3 Learning the prior

The prior model, $p(\mathcal{S})$, defines a probability distribution over all possible scenes. This section discusses exactly where this distribution comes from.

There are three general approaches for generating the distribution, all of which involve first selecting a form of model, Ψ, to use, with a set of free parameters, Θ, such that the probability distribution can be written as

$$p(\mathcal{S}) = \frac{1}{\zeta(\Theta)}\Psi(\mathcal{S}, \Theta), \tag{2.4}$$

where $\zeta(\Theta)$ is a normalization factor, sometimes known as the partition function, which is defined by

$$\zeta(\Theta) = \int \Psi(\mathcal{S}, \Theta)\, \mathrm{d}\mathcal{S}. \tag{2.5}$$

[2] Defining regularization to be the prior term in a posterior probability is merely one interpretation (a Bayesian one) of the regularization process, but has powerful repercussions.

For many forms of model, Ψ, computing $\zeta(\Theta)$ (ζ for short) is intractable[3]. Fortunately, when solving equation 2.2, ζ is just a multiplicative constant, so will not affect the value of $\mathcal{S}_{\text{MAP}}$. As such, models with no free parameters, *i.e.* non-parametric models, are often selected, or the parameters are either set by the user or chosen to optimize some other, tractable score; such models will be described as "ad hoc".

The second approach is to maximize the likelihood of a corpus of training data, $\{\mathcal{T}_i\}_{i=1}^{T}$,

$$\Theta = \underset{\Theta}{\operatorname{argmax}} \frac{1}{\zeta(\Theta)^T} \prod_{i=1}^{T} \Psi(\mathcal{T}_i, \Theta), \tag{2.6}$$

either by evaluating the likelihood over a discrete set of parameters and choosing the best, or using a gradient-ascent-based approach. While the former method requires that ζ is tractable, the latter method can be achieved by using various approximations to the gradient, *e.g.* [Hin02, Hyv05], that avoid computing ζ or its derivative. Models generated using this approach shall be referred to as ML models.

The final approach is to marginalize out the unknown model parameters as follows

$$p(\mathcal{S}) = \int \frac{1}{\zeta(\Theta)} \Psi(\mathcal{S}, \Theta) \cdot \frac{1}{\zeta(\Theta)^T} \prod_{i=1}^{T} \Psi(\mathcal{T}_i, \Theta) \, \mathrm{d}\Theta, \tag{2.7}$$

producing what shall be referred to as "fully Bayesian" models, as they follow the Bayesian principle of marginalizing out latent variables. This model adds another layer of generally intractable integration to the model learning, and as such is avoided by current priors for NVS, and indeed for the priors introduced in this thesis. As such, the approach is merely discussed here for completeness.

It should be noted that the choice of parameterized model, Ψ, is itself rather ad hoc, more so the fewer learned (or marginalized) parameters the model has, so there is clearly a sliding scale from ad hoc to either ML or fully Bayesian models.

2.3.1 Graphical models

Often probabilistic models, such as priors, over a set of variables, $\mathbf{X} = X_1, .., X_n$, model statistical dependencies only over certain subsets of variables, known as *cliques*. These dependencies can be helpfully expressed in the form of a graphical model [Bis06, Chapter 8], in which each node represents a variable and each edge represents a dependency, indicating that the variables joined by the edge are in a clique

[3]Tractability includes computing a reasonable estimate using Monte Carlo integration [Mac03, Chapter 29] in a reasonably short time frame.

together. Three kinds of graphical model are shown in figure 2.2. The first graph, figure 2.2(a), shows a directed, acyclic model (no loops exist), also called a Bayesian network, in which edges indicate unidirectional dependencies from parent to child variables, such that the complete model can be decomposed, using the product rule of probability, as

$$p(X_1, .., X_n) = \prod_{i=1}^{n} p(X_i|\text{parents}(X_i)). \tag{2.8}$$

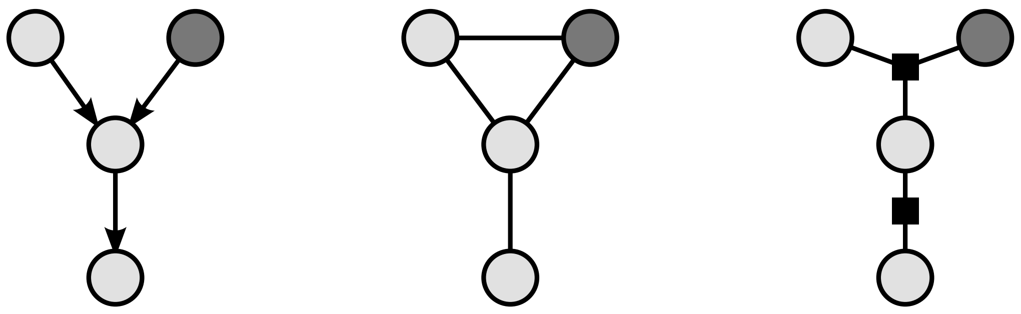
Importantly, this form of decomposition does not require a global normalization, therefore each of the conditional probabilities can be trained independently. Indeed, the most likely value of each variable can be computed even if the conditional probabilities are not normalized (*i.e.* only the *shape* of the distribution is correct), making training much simpler.

A more general model is the undirected graph, shown in figure 2.2(b), which allows bidirectional dependencies, a result of which is that the product rule cannot be applied to it. Instead, one turns to the *Hammersley-Clifford theorem* [Bes74], which states that the probability can be decomposed into a product of functionals over the cliques defined by the graph. Each edge in an undirected graph represents a clique containing only the two end nodes, *except* when a larger set of nodes are all directly connected to one another, in which case these nodes form a larger clique. If $\mathbf{X}_i$ represents the subset of variables in the *i*th of C cliques then the complete distribution can be written as:

$$p(X_1, .., X_n) = \frac{1}{\zeta} \prod_{i=1}^{C} \psi_i(\mathbf{X}_i). \tag{2.9}$$

Even if each of the clique functionals, $\psi_i(\cdot)$, are normalized distributions (not a necessary feature, though it is necessary that they return non-negative values only), their product will not generally integrate to a constant, so the overall distribution must be normalized by computing the partition function, ζ. An upshot of the above decomposition is that cycles in the graph are now permissable, and for this reason an undirected graphical model is also called a Markov random field (MRF). The downside is that the parameters of all clique functionals must be trained together, as these parameters affect the value of ζ, which itself does not generally have a closed-form solution[4], even if the integrals of each of the clique functionals do, making the prior model learning process much less tractable. A further, important result

[4]A product of zero mean Gaussians is a useful exception, the normalization term being the sum of the determinants of the covariance matrices.



(a) Directed, acyclic graphical model / Bayesian network.

(b) Undirected, cyclical graphical model / Markov random field.

(c) Factor graph representation of (b).

Figure 2.2: Three kinds of graphical models. The dark grey node denotes an observed random variable; all other random variables are unknown. Figure reproduced from [Rot07].

is that $\psi_i(\mathbf{X}_i)$ cannot be assumed to have the same distributional *shape* as $p(\mathbf{X}_i)$, a fact that can be explained by reasoning that $p(\mathbf{X}_i)$ does not take into account any dependencies with other, overlapping cliques.

The ambiguity in how to represent pairwise functionals between fully connected nodes has led to the development of the factor graph [KFL01], shown in figure 2.2(c). This is a bipartite graph, with circular nodes, representing variables, connected only to clique functionals, represented by square nodes, indicating the cliques each variable is in.

The complexity of the prior model can be characterized by the size of the largest clique in the model—a prior with only unary (single variable) cliques is a zeroth order prior, a prior with pairwise cliques is a first order prior, and so on (*i.e.* order equals largest clique size minus one), with the term "higher order prior" referring to second order priors or higher. Figure 2.3 shows the posterior MRF models typical of low-level vision problems such as NVS; the cliques of the prior model connect the unknown variables (light grey) together, while the data likelihood terms connect the unknown variables to observed variables (dark grey).

2.3.2 Meta-priors

I use the term "meta-prior" to describe prior knowledge regarding the conditional relationship between input data and output variables that are explicitly not noiseless reconstructions of the input data. This is in contrast to traditional priors, which model the joint probability of the output data itself, and data

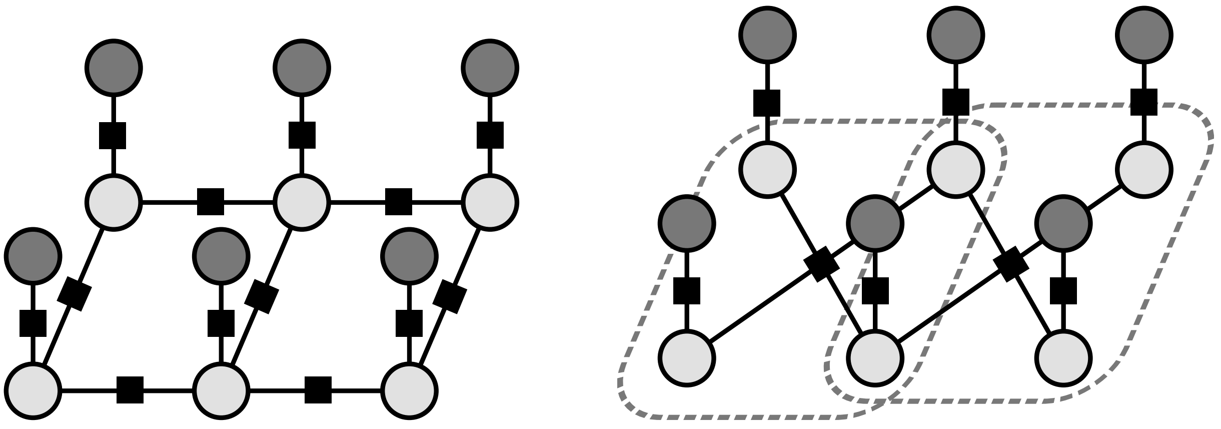

(a) A pairwise MRF. (b) A higher order MRF.

Figure 2.3: Pairwise and higher order MRFs. Factor graphs of MRF models typical of low-level vision problems such as NVS, showing (a) a model with only pairwise cliques, and (b) a model with higher order (4 variable) cliques, with the cliques circled with dashed lines for clarity. Figure adapted from [Rot07].

likelihood terms, which model the statistical relationship between input data and their hidden, noiseless values (though knowledge of the noise model might also be called a meta-prior).

One example of a meta-prior is the conditional random field (CRF) [LMP01]. Given a set of observed input data, $\mathbf{V}$, let the aim be to infer some property of the data other than its noiseless value ($\mathbf{V}^*$), for example a class label, denoted $\mathbf{X}$. The traditional approach is to learn a prior on the joint probability of noiseless input data, and output data, $p(\mathbf{V}^*, \mathbf{X})$, then maximize the posterior probability $\int p(\mathbf{V}|\mathbf{V}^*)p(\mathbf{V}^*, \mathbf{X})\mathrm{d}\mathbf{V}^*$, which might be approximated by $\max_{\mathbf{V}^*} p(\mathbf{V}|\mathbf{V}^*)p(\mathbf{V}^*, \mathbf{X})$. The CRF approach is to directly learn a model for the conditional relationship between input and output variables, $p(\mathbf{X}|\mathbf{V})$, which greatly simplifies both learning and inference.

A useful aspect of NVS, which it has in common with other vision applications such as image inpainting and texture synthesis, is that the input images are themselves examples of what the output should look like. One can therefore learn the prior model from the input data itself (an approach used by many of the priors discussed in this thesis), in which case what is considered to be a prior, $p(\mathbf{X})$, is actually conditioned on the input data, $p(\mathbf{X}|\mathbf{V})$, making it a meta-prior—the prior knowledge here is simply that the input and output should share certain statistics which make them visually similar. Often such prior models could equally have been learnt from other data (becoming normal priors), but their ability to regularize the problem at hand may well be reduced. For the remainder of this thesis the notion of a meta-prior shall simply be subsumed into the general class of priors.

2.4 Energy minimization

When finding the MAP value of equation 2.9, one can ignore the partition function, ζ, as this is a multiplicative constant which will not change the value of $\mathbf{X}$ that maximizes $p(\mathbf{X})$. Furthermore, if the clique functionals of equation 2.9 involve exponentials, such as Gaussians, or the inference algorithm to be employed can only optimize over sums of functions, then it can be preferable to minimize the negative log of the unnormalized probability, a value referred to as the energy:

$$E(X_1, .., X_n) = -\sum_{i=1}^{C} \log\left(\psi_i(\mathbf{X}_i)\right). \tag{2.10}$$

Doing so casts our problem—finding the MAP solution to the probability given in equation 2.3—into the familiar MRF (or CRF) energy minimization framework now standard in computer vision [KZ02, SP07]:

$$\mathcal{S}_{\text{MAP}} = \operatorname*{argmin}_{\mathcal{S}} E(\mathcal{S}|\mathbf{I}_1, .., \mathbf{I}_N) \tag{2.11}$$

$$E(\mathcal{S}|\mathbf{I}_1, .., \mathbf{I}_N) = \underbrace{E_{\text{photo}}(\mathbf{I}_1, .., \mathbf{I}_N|\mathcal{S})}_{\text{data likelihood}} + \underbrace{E_{\text{prior}}(\mathcal{S})}_{\text{prior}} \tag{2.12}$$

2.4.1 Simplifying assumptions

The data likelihood term measures the likelihood of our input data in light of our scene model and a noise model. For the purpose of this work—an investigation into priors for NVS, rather than NVS itself—the data likelihood model is to be kept as simple as possible, allowing attention to be fully focussed on the prior model. I therefore assume that a point in the scene, sampled by a camera from any angle (assuming the point is still visible) will always, in the absence of sampling noise, give the same colour. Note that this not only implies a Lambertian reflectance assumption, but also that there is no colour space transformation (*e.g.* due to different exposure settings or white balance) between input images.

In terms of a noise model, it is assumed, as stated previously, that the given projective functions are noiseless, and it is also assumed that sampling noise is independent and identically distributed (*i.i.d.*) over the input image pixels. The data likelihood term can therefore be evaluated independently over each of the input pixels, and summed, thus:

$$E_{\text{photo}}(\mathbf{I}_1, .., \mathbf{I}_N|\mathbf{I}_0^*, \mathsf{D}) = \sum_{i=1}^{N} \sum_{\mathbf{x}\in\mathcal{X}_i} f\left(I_i(\mathbf{x}) - I_0^*\left(\pi_i^{-1}\left(\mathbf{x}, D_i(\mathbf{x})\right)\right)\right), \tag{2.13}$$

where $f(\cdot)$ is the data cost function defined by our noise model, $\mathcal{X}_i$ is the set of pixels in I_i, such that $\mathbf{x}$ is a pixel in I_i, and $D_i(\mathbf{x})$ is the disparity of pixel $\mathbf{x}$ in I_i.

Computing $D_i(\mathbf{x})$ from D is non-trivial for several reasons:

- Since D is defined in continuous space the correspondences defined by D generally do not lie on exact pixel locations in I_i, so that D_i must be resampled, which causes errors, especially around discontinuity boundaries.
- Several pixels in D may project onto the location $\mathbf{x}$ in I_i, requiring some form of occlusion reasoning to evaluate $D_i(\mathbf{x})$.
- $D_i(\mathbf{x})$ will not have a value if no pixels in D project onto the location $\mathbf{x}$ in I_i.

For this reason, a further, standard assumption made is that E_{photo} can be approximated by summing over pixels in I_0^*, rather than pixels in $\mathsf{I}_1, .., \mathsf{I}_N$, thus:

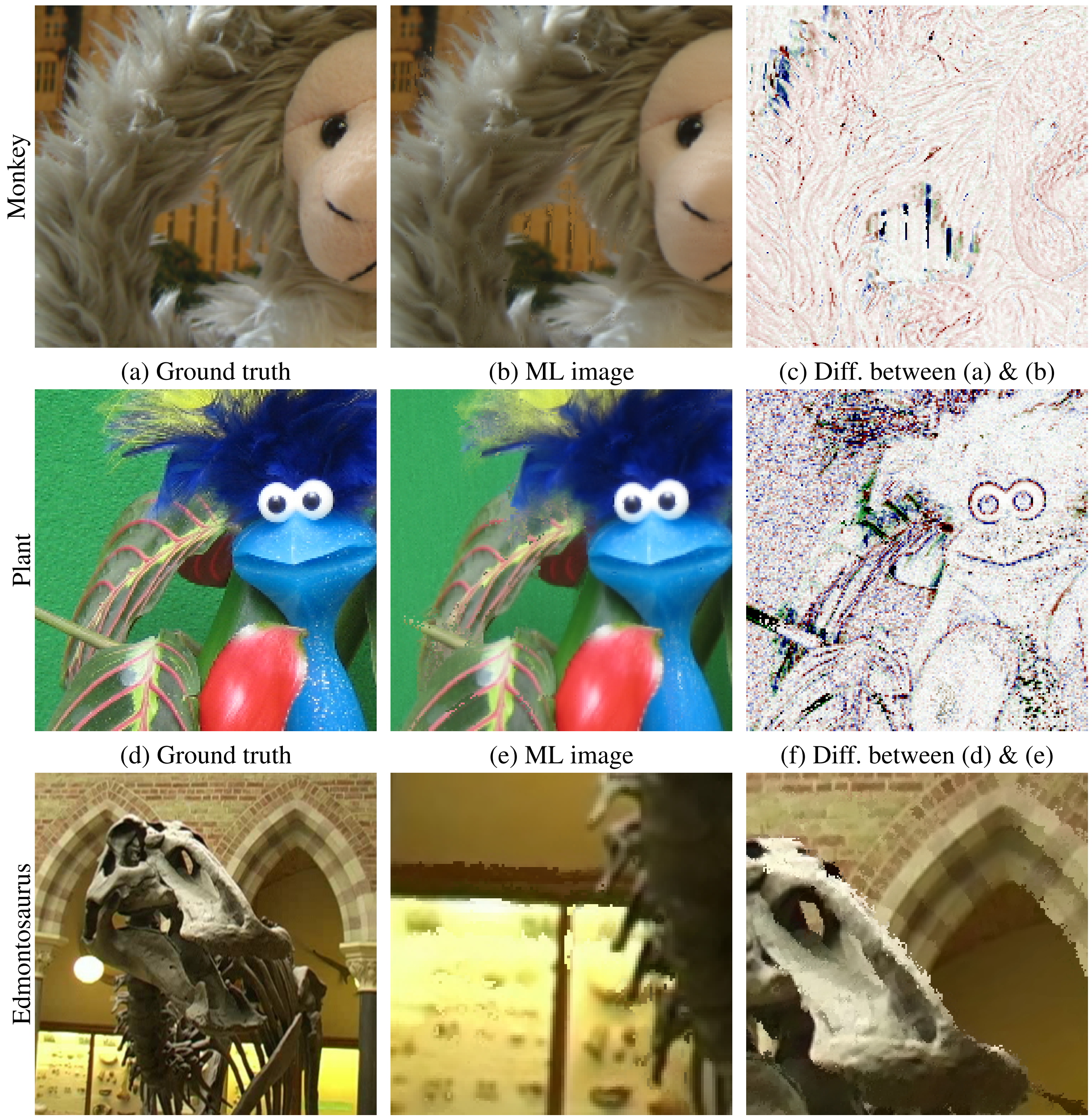
$$E_{\text{photo}}(\mathsf{I}_1, .., \mathsf{I}_N | \mathsf{I}_0^*, \mathsf{D}) = \sum_{\mathbf{x} \in \mathcal{X}_0} \sum_{i=1}^{N} f\left(I_i\left(\pi_i\left(\mathbf{x}, D(\mathbf{x})\right)\right) - I_0^*(\mathbf{x})\right). \tag{2.14}$$

As Gargallo & Sturm point out [GS05], this approach miscounts the contribution of each input pixel to the overall probability, leading to errors in wide baseline situations. However, I shall employ it for simplicity, and stick to narrow baseline sequences.

2.5 Test sequences

For consistent comparison of results throughout this work, I will use a fixed set of test sequences across all chapters, each of which have particular characteristics that make them challenging for NVS algorithms. These sequences, shown in figure 2.4 along with the ML output images (assuming *i.i.d.* Gaussian noise, as in chapters 4 & 5), are as follows:

- **Monkey** – A sequence of 27 still photos, taken from [FWZ05], calibrated using Boujou [2d303]. The output image reconstructs a 200×200 pixel region of an input image from four neighbouring views ($N = 4$). Challenges are occlusion of the background around the arm, fine detail of the fur in the silhouette of the arm, and low contrast stochastic texture on the face.



(a) Ground truth (b) ML image (c) Diff. between (a) & (b)

(d) Ground truth (e) ML image (f) Diff. between (d) & (e)

(g) Example image (h) Dino1 ML image (i) Dino2 ML image

Figure 2.4: Test sequences. This figure shows the various image sequences that will be used to test the NVS algorithms presented in this work. The top two rows show the *Monkey* [FWZ05] and *Plant* [WF05] sequences respectively, containing (a & d) the ground truth image to be reconstructed in a leave-one-out test, (b & e) the maximum likelihood reconstruction, and (c & f) the difference between ground truth and the ML image. The bottom row shows the *Edmontosaurus* [WRTF06] sequence, containing (g) an image from the input sequence (a 63 frame video), and (h & i) maximum likelihood reconstructions of sections of output frames from a novel "steadicam" (where the camera trajectory has been smoothed) version of the original video sequence, which will be called *Dino1* and *Dino2* respectively.

- **Plant** – A sequence of 12 still photos, taken from [WF05], calibrated using Boujou. The output image reconstructs a 180×180 pixel region of an input image from four neighbouring views ($N = 4$). Challenges are fine features surrounded by regions of homogenous colour (*e.g.* 'ribs' on leaves; stalk), fine details of the feathers, occlusions to the left of the toy head, high frequency specularities on the toy body, and the low contrast stochastic texture of the baize background.

- **Dino1** – A 63 frame video sequence, taken from [WRTF06], calibrated using Boujou. The output image is a 120×120 pixel image from a novel viewpoint, generated as part of a "steadicam" version of the original video, whereby the camera trajectory has been smoothed. The image is of a similar resolution and distance from the scene as the input views, and is generated using six neighbouring views ($N = 6$). Challenges are fine features surrounded by regions of homogenous colour (*e.g.* cabinet door frame, objects in cabinet), and the complex occlusion boundary of the dinosaur neck vertebrae.

- **Dino2** – Similar to Dino1, except that the output image is a different, 200×200 pixel region of a different frame of the novel, steadicam video trajectory. Challenges are noisy input data (input images are highly compressed; some input frames are motion blurred/out of focus; camera calibration is possibly poor for some frames), and fine features surrounded by regions of homogenous colour (*e.g.* brickwork).

In addition, all the ML images suffer from the single pixel artifacts associated with the random possibility of an incorrect colour being more likely, given the data.

In chapter 8 some Middlebury stereo sequences [SS08] are also used. All the sequences used are made up of RGB images with 256 grey levels per channel, the difference in value between consecutive grey levels being 1.

2.5.1 Quantitative error measurement

Quality of reconstruction can be determined either in terms of visual plausibility, *i.e.* how real an image looks to a human observer, or in terms of difference from ground truth. Because the former measure is more subjective, it will be evaluated qualitatively. The latter will be evaluated quantitatively, using two measures: r.m.s. error and proportion of gross errors. Ground truth is only available for the Monkey and

Plant test images, so the measures will be applied to only these two sequences.

2.5.1.1 R.m.s. error

R.m.s. error is the "root mean squared" error. If **I** is the output image and **J** is the ground truth, then the r.m.s. error is computed as

$$\epsilon_{\text{rms}} = \sqrt{\frac{1}{|\mathcal{X}|}\sum_{\mathbf{x}\in\mathcal{X}} \|I(\mathbf{x}) - J(\mathbf{x})\|^2}, \tag{2.15}$$

where $\mathcal{X}$ is the set of all pixels in **I**.

2.5.1.2 Gross errors

Gross errors are the number of pixels that are grossly wrong, given as a percentage of the total number of pixels. The measure is computed as

$$\epsilon_{\text{ge}} = \frac{100}{|\mathcal{X}|}\sum_{\mathbf{x}\in\mathcal{X}} \left[\|I(\mathbf{x}) - J(\mathbf{x})\|^2 > 1000\right], \tag{2.16}$$

where $[\cdot]$ is the Iverson bracket.[5]

2.5.1.3 Difference images

Difference images presented in this thesis, *e.g.* in figure 2.4(c & f), are not renderings of **I** − **J**, but rather of a transformation of this image which plots zero difference as white, positive intensity[6] differences as blue, negative ones as red, and differences in chrominance as green. Darker colour indicates a larger difference. This allows one to distinguish between different types of artifact, *e.g.* when a region is predominantly lighter in one image than another, or when a feature is smaller in one image, rather than shifted slightly.

[5]The Iverson bracket: $[statement] = 1$ if $statement$ is true, 0 otherwise.

[6]The intensity of a colour image is taken to be the Y channel of the image in YCbCr colour space.

Chapter 3

Literature review

In this chapter I review the priors used in Computer Vision literature which might be useful for NVS, and also the types of inference algorithms which can be used to optimize the resulting problems. The first section gives an overview of inference techniques available, the second reviews the prior models commonly used to regularize scene geometry, and the final section looks at the various priors used to model the likelihood of natural images.

3.1 Inference algorithms

As discussed in the previous section, there are generally two inference problems in Bayesian frameworks—that of learning the prior, and that of inferring the optimal (in case of this thesis, MAP) output solution, given the prior and input data. Knowing what inference tools are available, and how effective they are, is key to understanding how well different prior models can be learned and applied. This section gives a brief overview of the inference algorithms commonly used in Computer Vision applications.

3.1.1 Message passing algorithms

Message passing algorithms are the class of algorithms based on Pearl's "Belief Propagation" (BP) algorithm [Pea88], which compute the marginal, in the case of sum-product BP, or min-marginal in the case of max-product BP, probabilities of variables by iteratively sending messages along the edges of a graphical model until convergence. Each message is a distribution, *e.g.* a histogram, parameterized function or mixture of Gaussians [SIFW03], over values of the target variable. The original BP sent messages between pairs of variables, permitting only pairwise interactions; in developing factor graphs, Kschischang *et al.* [KFL01] generalized the method to higher order cliques by sending messages from variables to factors and vice-versa. There are no constraints on the form of clique functionals. The MAP solution is indicated by the maximum min-marginal probability of each variable, obtained using max-product BP; when minimizing energy instead of maximizing probability, the max-product algorithm becomes the min-sum algorithm, the minimum final energy value of each variable indicating the MAP solution.

3.1.1.1 Optimality

BP has been proven [KFL01, Pea88] to give the optimal MAP solution for any factor graph without loops, reached after a maximum of two sequential[1] message-passing iterations. There are no optimality guarantees for BP applied to loopy graphs, or even guarantees that the messages will converge—oscillations can occur. However, it has been found that, in practical vision applications, messages do tend to converge, and do so on a reasonable solution. Additionally it has been proven that convergent solutions on graphs with a single loop are optimal also [Wei00].

The “Junction Tree” algorithm [LS88] allows graphs with loops to be solved optimally, by transforming the loopy graph into a tree. However, this can greatly increase clique size, *e.g.* a regular grid connected graph would be converted to a single clique. “Generalized Belief Propagation” [YFW00] sends messages not only between connected nodes but also between regions of nodes, improving the solution converged to.

3.1.1.2 Efficiency

A single iteration of BP has complexity $\mathcal{O}(MN^k)$, where M is the number of nodes, N the number of labels and k the clique size. Several approaches have been used to increase the efficiency of the algorithm, including the use of a multi-resolution framework to speed mixing and therefore convergence [FH06], and, for certain forms of clique functional, the use of the distance transform [FH06, LRHB06] to speed up message computation. For problems with clique functionals that do not involve products of different input variables, Potetz [Pot07] shows how constraint nodes can be used to reduce complexity to $\mathcal{O}(kMN^2)$.

[1]Messages can either be sent synchronously between nodes, or sequentially, starting from a designated root node and moving along the graph in order.

3.1.1.3 Tree-reweighted message passing

Tree-reweighted message passing (TRW) [WJW03] is another algorithm for MAP estimation on loopy graphs. Based on BP, rather than simply apply the algorithm to the loopy graph, it decomposes the problem into tree-structured subproblems that sum (in the case of energy minimization) to the original problem, which are then solved optimally using BP. In order to achieve consensus between subproblems, the energy distributions for a given variable are averaged between subproblems, then the subproblems re-solved and so on. Wainwright *et al.* [WJW03] update all variables simultaneously, leading to a non-convergent algorithm, but prove that if the subproblems agree then the optimal solution has been found. Kolmogorov proposed a modification whereby the variables are updated sequentially and the subproblems re-solved in between, calling this TRW-S, which he proved converged to give a local minimum energy, additionally providing a lower bound on the energy, giving some indication of the solution's optimality.[2] Recently Komodakis *et al.* [KPT07] showed how the projected subgradient method can be used to find the *optimal* lower bound on energy using this decomposition approach, finding better minima as a result. Their algorithm has the additional benefit that it only requires the MAP solution of each subproblem for each update, so that other optimizers and forms of subproblem can be employed for reasons of efficiency or to increase the tightness of the bound.

3.1.2 Graph cuts

It has been shown [GPS89, KZ04] that the max flow/min cut algorithm of Ford & Fulkerson [FR56] can be used to find the optimal MAP solution to certain binary[3] MRF (and CRF) problems, in a framework now commonly known as "graph cuts"—specifically, Kolmogorov & Zabih [KZ04] proved that only energies that can be written as

$$E(\mathbf{X}) = \sum_i \phi_i(X_i) + \sum_{i<j} \phi_{ij}(X_i, X_j), \qquad X_i \in \{0,1\}\ \forall\, i \tag{3.1}$$

and for which all pairwise terms satisfy the submodularity constraint,

$$\phi_{ij}(0,0) + \phi_{ij}(1,1) \leq \phi_{ij}(0,1) + \phi_{ij}(1,0), \tag{3.2}$$

[2]When the lower bound is equal to the energy of the final solution then that solution is a global optimum.

[3]A binary problem, sometimes also referred to as a boolean problem, is one in which each variable has only two possible discrete states.

can be minimized (optimally) using graph cuts.

3.1.2.1 Extensions to non-submodular energies

In the case that one or more pairwise terms are non-submodular, the minimization problem becomes NP-hard and cannot be solved by graph cuts. However, a graph-cuts-based construction called "Roof Duality" [HHS84], introduced to the Computer Vision community [KR06b, RKLS07] under the name "Quadratic Pseudo-Boolean Optimization" (QPBO), is able to label some (including all or none) of the variables of a non-submodular problem, with certain properties:

[P1] *All nodes labelled by QPBO are part of a global optimum labelling.*[4]

[P2] *QPBO returns the same result for equivalent problems that are parameterized differently.*

[P3] *Let $\mathbf{X}'$ be the output of QPBO, with unlabelled nodes set to zero. Then $E(\mathbf{X}') \leq E(\mathbf{0})$.*

Two extensions to QPBO which improve the output labelling have been proposed. The "probe" method [BHT06] (QPBOP [RKLS07]) of Boros *et al.* can label additional nodes optimally, but requires a number of graph solves which is at worst exponential in the number of unlabelled pixels. The "improve" method (QPBOI) [RKLS07] of Rother *et al.* can label all remaining nodes sub-optimally, in a time linear in the number of unlabelled nodes, which generally produces a lower energy than simply fixing unlabelled variables to a default value.

3.1.2.2 Extensions to higher order problems

Since graph cuts can only optimize energies which can be expressed by equation 3.1, any higher order cliques must be decomposed into a set of pairwise terms. This has been shown to be possible, first for triple cliques which project[5] onto submodular pairwise terms [KZ04], and consequently for cliques of all sizes [FD05], again only where the cliques project onto submodular pairwise terms. Unfortunately, not only does the set of pairwise edges and vertices generated generally grow exponentially with clique size, but the submodularity constraint becomes more limiting the larger the clique [FD05]. However, it has recently been shown that not only do higher order Potts models [KKT07] and truncated linear

[4]A global optimum labelling is one which has the lowest energy of all possible labellings. Note there may be more than one of these.

[5]The projection referred to here reduces higher order cliques to pairwise cliques by fixing all but two variables to each of their possible values in turn.

kernels [KLT08] fulfil the submodularity constraint, but that they also require only two extra graph vertices per clique, regardless of clique size.

3.1.2.3 Extensions to multi-label problems

Multi-label problems, where each variable has a discrete set of possible values represented by labels, are much more common than binary problems in Computer Vision. Graph cut algorithms have been developed to tackle certain types of multi-label problems. The approach of Ishikawa [Ish03] transforms a pairwise multi-label problem into a pairwise binary problem; the resulting graph is only submodular if each pairwise energy is convex w.r.t. each variable, when the other variable is fixed to any value. Schlesinger & Flach [SF06] generalize this transformation, which they call "K to 2", to support a wider class of pairwise energies. Boykov *et al.* [BVZ01] solve multi-label problems through a sequence of binary optimizations between the current solution and a set of input labellings, or moves; the submodularity constraint requires that the moves are either each uniform ("α-expansion") or the same as the current solution but for the variables that have two particular labels, whose labels are swapped ("$\alpha\beta$-swap"), and that the pairwise terms are *metric* and *semi-metric* [BVZ01] respectively, constraints consequently generalized in [KZ04]. While the binary optimizations are optimal, the final solution is a local minimum w.r.t. the moves, but, since the moves are large scale, the local minima found tend to be good solutions in practice. Veksler [Vek07] combines the exact and approximate approaches to create another approximate algorithm called "$\alpha\beta$ range moves" which optimizes truncated convex priors even more effectively.

With the introduction of QPBO, the constraints on both exact and approximate multi-label approaches due to the submodularity constraint are removed. Raj & Zabih [RSZ06] first used QPBO with the α-expansion approach, successfully—for their MRI reconstruction problem they quote a maximum of 2% unlabelled variables per optimization. Using QPBO removes the constraint on pairwise energies, consequently allowing arbitrary moves generally[6], a fact noted by Lempitsky *et al.* [LRB07], who call the resulting sequence of binary optimizations "fusion moves", generalizing α-expansion and $\alpha\beta$-swap in the process. Property P3 of QPBO, known as the "autarky" property [RKLS07, page 2], ensures that α-expansion using QPBO is convergent, regardless of the proportion of variables that are labelled. Recently Kohli *et al.* [KSR$^+$08] used QPBO and QPBOP to solve non-submodular multi-label problems in

[6]Arbitrary (*i.e.* non-uniform) moves have always been permitted using α-expansion, simply by reparameterizing the α (move) labels for each variable individually, as shown to good effect in [WS06], but the constraints on pairwise terms have meant that such moves could rarely be used.

a single optimization using the K to 2 transformation [SF06].

3.1.2.4 Efficiency and efficacy

Due to the popularity of the graph cuts algorithm, much work has gone into making it and its extensions fast. Boykov & Kolmogorov [BK04] developed an implementation of the max flow/min cut algorithm that performs well on graphs from standard Computer Vision applications, with a compute time for image denoising that is empirically linear in the number of nodes. Kohli *et al.* [KT05] developed a framework that enables fast re-solving of graphs that have been modified slightly, with several applications including the acceleration of α-expansion [AKT08], something also tackled by the development of the graph-cuts-based "Fast PD" algorithm [KTP07], as well as the "LogCut" algorithm [LRB07], which uses QPBO to optimize over ranges of labels rather than single labels, effectively reducing computation time exponentially with the number of labels.

The efficacy of α-expansion graph cuts has been compared to those of several other optimizers on some typical Computer Vision problems [KR06a, SZS$^+$06, TF03]. Szeliski *et al.* [SZS$^+$06] showed that on regular grid graphs, α-expansion performs similarly to TRW-S, both of which reach a lower energy than BP, while Kolmogorov & Rother [KR06a] show that on highly connected graphs, such as those used in stereo with occlusions, α-expansion performs significantly better than other algorithms.

3.1.3 Local approaches

I classify local optimization approaches as those which require a concept of current state, and develop a solution iteratively from this point. Unlike the above methods (which will be referred to as global optimization methods, despite their general lack of optimality), such approaches do not generally place any limit on clique size, either theoretically or practically (computation time is linear in clique size, for linear clique functionals), but they do tend to find poor, local minima.

3.1.3.1 Gradient descent

If the output variables are continuous and the clique functionals of the model are partially differentiable, then the gradient of the energy w.r.t. each variable can be computed (given a current solution), and a gradient descent (or ascent, in the case of maximizing probability) approach used to find an equilibrium solution to the problem. Such approaches will deterministically find a solution whose gradient is zero, which will generally occur at a local minimum.

The level sets method [Set98] reparameterizes the problem, embedding the state space into a higher dimensional space (*e.g.* a parameterized 2-d surface is embedded in a 3-d space, the surface being represented by an isocontour in this higher-dimensional space), and uses gradient descent in this hyperspace to find the solution. The benefit of this approach is that it allows topological changes of the current state which could not otherwise be parameterized.

3.1.3.2 Coordinate descent

Coordinate descent approaches are those which sequentially minimize subsets of variables, conditioned on the current state of the rest of the variables. The minimization can be local or global but, as it cannot increase the energy, the cycle over subsets of variables can be repeated until convergence. The most common of these approaches is "Iterated Conditional Modes" (ICM) [Bes86], which sequentially globally minimizes the energy for each variable individually, conditioned on the rest. If subsets of variables are independent of each other (*i.e.* do not have a clique in common), then these variables can be updated simultaneously using ICM. The minimum number of such subsets depends on the size and connectivity of the cliques, but, in the particular case that the variables form a bipartite graph, one can update half the variables conditioned on the other half, followed by the reverse.

Factor graphs, as shown in figure 2.2(c), are bipartite graphs, but it should be noted that the variables themselves do not form a bipartite graph. However, it is sometimes the case that the functionals of each clique in such a graph contain one or more latent, variable parameters of the model which are dependent solely on the output variables and on which the output variables solely depend, so that the factor graph can be decomposed in the fashion shown in figure 3.1. The bipartite graph formed in this decomposition, with latent parameters in one set, and output variables in another, can be optimized using ICM with two steps per cycle. This special case is similar to Expectation Maximization (EM) [DLR77], in which the E-step evaluates the expected value[7] of the parameters, given the current state of the variables, then the M-step maximizes the likelihood of the variables given the current parameters.

[7]The expected value is generally a real value, so is only useful if the parameters are continuous variables. In the case of discrete parameters, the ML value can be used instead, making this 'EM-style' ICM.

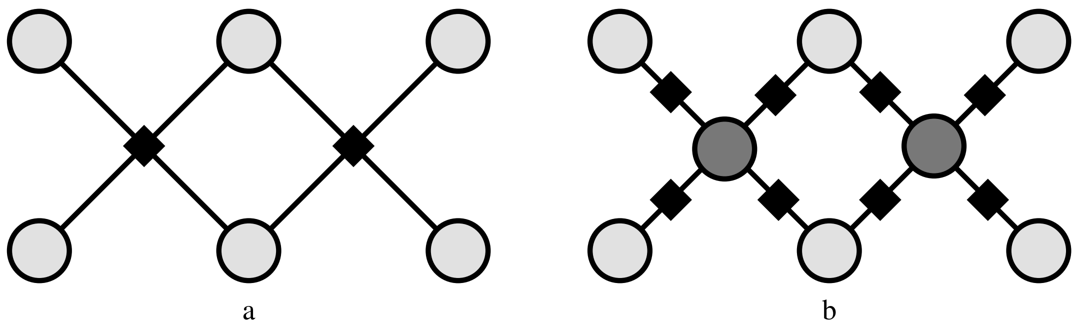


Figure 3.1: EM suitable decomposition. If the functionals of the factor graph shown in (a) use a latent, variable parameter (dark grey) to generate the dependencies between clique output variables (light grey), such that it can be decomposed as shown in (b), then the MRF represented by this graph is suitable for optimization using EM if the parameters are continuous, or two step 'EM-style' ICM if not.

3.1.3.3 Avoiding local minima

The local inference methods discussed in this section deterministically find a local minimum, starting from some initial state. If there exist many local minima then these methods are likely, without a good initialization, to get stuck in one of them. Several approaches are commonly used to avoid these local minima. The first, "Simulated Annealing" (SA) [KGV83], specifically the version of Geman & Geman [GG84], can be seen as an extension of both ICM and Gibbs sampling [GG84], which generates random samples from a joint distribution by sequentially drawing from conditional distributions. In SA, a temperature parameter, T, makes each conditional distribution more peaked around the maximum as $T \to 0$, as follows

$$p(X_i = \mathrm{x}|\mathbf{X}_{\not{i}}) = \frac{\exp\left(-E(\mathbf{X}_{\not{i}}, X_i = \mathrm{x})/T\right)}{\sum_{\mathrm{y}} \exp\left(-E(\mathbf{X}_{\not{i}}, X_i = \mathrm{y})/T\right)}, \tag{3.3}$$

where $\mathbf{X}_{\not{i}}$ is the vector $\mathbf{X}$, excluding the i^{th} variable, X_i. In practice, the value for each variable is drawn from this distribution (either synchronously or sequentially) using the Monte Carlo (MC) approach of proposing a random new value, x, for X_i, then accepting it according to the Metropolis algorithm [MRR+53]—letting $e = E(\mathbf{X})$ and $e' = E(\mathbf{X}_{\not{i}}, X_i = \mathrm{x})$, the new label is always accepted if $e' < e$, otherwise it is accepted with probability $\exp((e - e')/T)$. T is slowly reduced according to a user-defined cooling schedule. When $T = 0$, SA becomes ICM, but at higher temperatures, because the draws are random, the algorithm can jump out of local minima.

Another, common approach is to use a multi-resolution framework, whereby each scale of the problem is initialized with the solution to a coarser scale approximation of problem, where there are generally fewer local minima to get stuck in. A related approach is Blake & Zisserman's "Graduated Non-

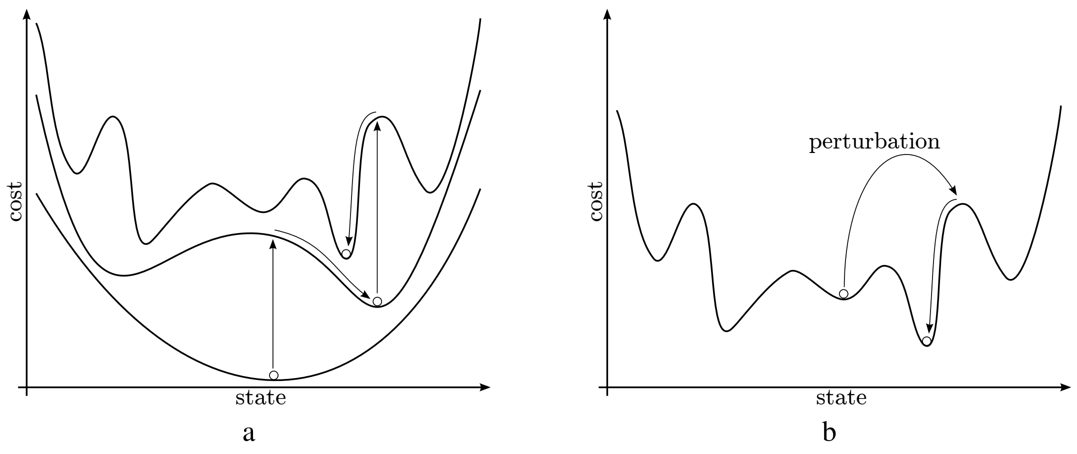


Figure 3.2: Avoiding local minima. (a) The GNC algorithm [BZ87] solves a series of problems, each less convex, but closer to the original problem, initializing the succeeding problem with the previous solution, in order to avoid local minima. A multi-resolution framework is very similar in this respect. (b) ILS [LMS02] searches through the space of local minima by perturbing the current solution, then using a deterministic algorithm to reach a local minimum, and iterating this process, keeping track of the lowest minimum found.

Convexity" (GNC) algorithm [BZ87], which creates a one parameter family of cost functions, the function at one end of the scale being convex, and at the other being the original objective function; the convex problem is solved, and the solution is used to initialize the next problem in the family, and so on until the original objective function is solved, as shown in figure 3.2(a).

"Iterated Local Search" (ILS) [LMS02] uses the fact that there are many times fewer local minima than possible solutions (given a deterministic, local minimizer), so searching in the space of local minima is more efficient. These local minima are themselves searched locally, as shown in figure 3.2(b), by perturbing the current solution slightly (enough to jump into a nearby convergence basin), computing the new local minimum, then iterating this process, always starting from the lowest cost solution found to date.

3.2 Surface geometry priors

Surface geometry, or smoothness, priors are used to regularize the estimated surface geometry of a scene, usually extracted from images. The form of the prior depends heavily on the parametrization of the surface; here I concentrate on parameterizations which generate a dense depth map for a reference input image, as these have been shown to be suitable for NVS [Sch96, Sze99, ZK07]. In such depth maps, each pixel in the reference image is given a value of depth or disparity perpendicular to the image plane, which defines the location of the visible surface along the ray associated with that pixel, and is either computed per pixel, or a plane is assigned to entire image regions. These depth maps are generally the product of *surface reconstruction* [Gri81, BZ87] algorithms, where noisy depth samples, extracted from image features through stereopsis [MP79] on two or more images, given sparsely or per pixel, are to be regularized and interpolated between (in the case of sparse samples), or the more general *dense stereo* problem [SS02], in which each pixel has a data-likelihood cost distribution over depth, and these costs are to be minimized while generating a plausible surface.

The per-pixel depth maps are smoothed by regularizing the first or second derivatives of depth or disparity. Early algorithms [Gri81, Ter83, Hor86, BZ87, Gen88] penalized the square of the second derivative, which is called the *thin plate* model, and can be shown to encourage planarity (see appendix A). The convex nature of the quadratic kernel generates a preference for many small changes in gradient over one large one, which has the effect of smoothing over surface discontinuities. This effect was overcome with the introduction of penalty functions which become concave at a certain gradient threshold [Ter85, BZ87, SS96], known as *discontinuity preserving* kernels, such as the truncated quadratic kernel which created what Blake & Zisserman [BZ87] called the *weak plate* model. These early algorithms all used local inference algorithms for optimization, using GNC [BZ87] and multi-resolution strategies [Ter83, Gen88] to avoid local minima.

In the approaches that followed, priors on the first derivative of disparity became more popular [Bar87, FK98, ADSW02, SFVG06], primarily due to reasons of computational efficiency, even though the optimizers remained local. These priors encourage surfaces to be parallel to the image plane (*i.e.* fronto-parallel), and are known, similarly to the quadratic and truncated quadratic kernels, as the *thin membrane* and *weak membrane* models respectively. First-order priors became more entrenched with the development of DP [Bel96, GLY95, Vek05], graph cuts [BVZ01, IG98, KZ01, RC98] and BP [SZS03, FH04a] based stereo algorithms, as these optimizers find much better minima, but also become much more costly with larger cliques. The kernel also changed, with truncated linear and Potts models becoming more common, for two reasons: firstly, graph-cuts-based [BVZ01, KZ01, WQ05] stereo algorithms tend to use α-expansion, which cannot optimize convex kernels [BVZ01]; secondly, the Middlebury stereo evaluation framework [SS08] introduced by Scharstein & Szeliski [SS02], on which such algorithms are currently judged, only penalizes the number of gross disparity errors, and the top performing methods generally use such kernels, suggesting they perform better under this metric. The result is that the output

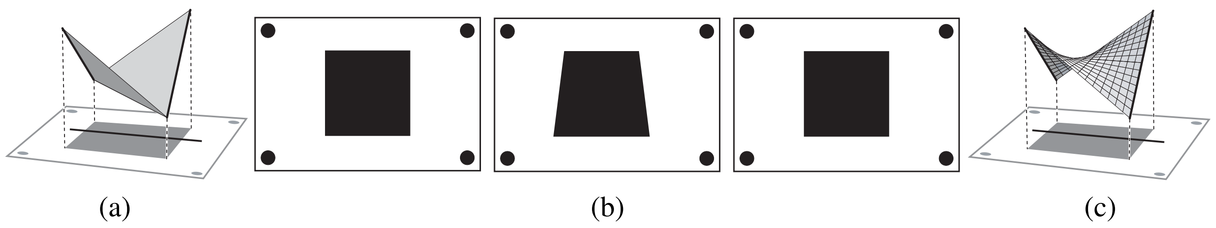


Figure 3.3: Stereogram of Ishikawa & Geiger. (b) shows a stereogram, with the centre image being the right view, and the outer images being the left view, allowing viewing by either crossing or diverging one's gaze. (a) shows one of two surface hypotheses that the human visual system hallucinates, while (c) shows a lower energy hypothesis under a convex fronto-parallel prior. Figure reproduced from [IG06].

depth maps tend to be more piecewise-fronto-parallel, with few to no areas with any gradient—clearly not an accurate model of the real world.

It has been noted that not only do such first order priors not model surfaces in the real world well [LZ06], but that they also do not model the prior used by the human visual system [IG06]. While first order priors are adequate in textured areas, which generate local regions of high data-likelihood over depth, Ishikawa & Geiger [IG06] showed, through the use of special stereograms (see figure 3.3), that in textureless regions the human visual system will hallucinate surfaces of zero Gaussian curvature[8], tending to generate more planar surfaces, while a first-order prior will generally not (as shown by figure 3.3).

Attempts have been made to model surfaces more accurately, while using the powerful graph cuts and BP optimizers, but these attempts have all employed pairwise cliques rather than the higher order cliques required for a true second order prior. Such attempts include *layered* [BSA98, BT99, TSA01] and *segment-based* [TSK01, BG04, HC04, KSK06, YWY$^+$06, BG07] approaches, which segment the reference image, and enforce the constraint that such regions be planar. The pairwise regularization of the latter algorithms encourages neighbouring regions to be coplanar, while the former algorithms effectuate the same simply by iterating the segmentation and plane fitting processes. Li & Zucker [LZ06] retain the pixel-based model while incorporating both second and third order priors, therefore merely encouraging planarity when there is ambiguity, rather than enforcing it across entire regions. However their algorithm precomputes local surface normals and in fact optimizes a first-order prior on the normals, which is an approximation to the true problem. The reason for the current absence of true higher order priors, despite their improved scene modelling capability, can be found in the literature: [LZ06], on using triple cliques,

"such an endeavour quickly makes the problem computationally infeasible",

[8]Gaussian curvature of a point on a surface is the product of the principal curvatures, so a zero Gaussian curvature surface will have no curvature in at least one direction.

and [BV06] on graph cuts,

"it is not clear if [triple cliques] can be used to encode a higher order smoothness".

Indeed, it has recently been shown [Koh07] that a second order smoothness prior generates non-submodular terms, precluding optimization using graph cuts.

3.2.1 CRF on smoothness

The canny reader will note that discontinuities in a disparity map, D, are generally aligned with colour (or texture) discontinuities of the corresponding noiseless reference image, $\mathbf{I}_0^*$. In a true Bayesian stereo framework this dependency between output variables would be modelled using a prior over the joint distribution, $p(\mathsf{D}, \mathbf{I}_0^*)$. However, very few dense stereo algorithms [SFVG04] attempt to generate $\mathbf{I}_0^*$; the usual, implicit assumption is that the input reference image, $\mathbf{I}_0$, *is* noiseless, *i.e.* $\mathbf{I}_0^* = \mathbf{I}_0$. The joint prior is therefore replaced with the conditional probability $p(\mathsf{D}|\mathbf{I}_0)$, creating a CRF framework.

Writing the MRF clique functional for a given neighbourhood, $\mathcal{N}$, of the smoothness prior as $\psi(\mathcal{N}, \mathsf{D})$ $= S(D(\mathcal{N}))$, the majority of CRF models [GP87, Fua93, BT99, BVZ01, ADSW02, KZ02, SZS03, SFVG04, SP07] simply modulate this functional with a weight based on an image feature in that neighbourhood, thus

$$\psi(\mathcal{N}, \mathsf{D}, \mathbf{I}_0) = W(\mathbf{I}_0, \mathcal{N}) \cdot S(D(\mathcal{N})), \tag{3.4}$$

where $W(\mathbf{I}_0, \mathcal{N})$ is said weight. The commonest image feature used in $W(\cdot)$ is the magnitude of the local image gradient of that neighbourhood, parallel to the neighbourhood [Fua93, BT99, BVZ01, ADSW02, KZ02, SFVG04, SP07]. The form of $W(\cdot)$ is generally hand-picked, but Scharstein & Pal [SP07] learn ML weights for a range of image gradient magnitudes, given a Potts smoothness model, using a gradient ascent approach. Other image features have also been used, such as the output of the Canny edge detector [GP87], or an image over-segmentation [SZS03]. The latter model's smoothness constraint is strengthened if the pixels in $\mathcal{N}$ are part of the same segment, encouraging discontinuities to align with segment boundaries. Note that this contrasts with the segment-based stereo methods [TSK01, BG04, HC04, KSK06, YWY$^+$06, BG07], which *force* discontinuities to align with segment boundaries.

3.2.2 Occlusion models

An important aspect of stereo frameworks, though affecting the data likelihood term rather than the prior, is the use of an occlusion model to determine when a correspondence is not visible in an input image. The data term assumes that a point on a surface will be the same colour when viewed from all angles, but for non-Lambertian reflectance properties (which are generally ignored) and sampling noise, generally with a distribution peaked around zero. However, it may be the case that such a point may be visible in some views, but not in others, because it is occluded by another part of the scene, in which case the colour sampled at the location of the correspondence will be that of an entirely different scene point. Data likelihood distributions are therefore often a mixture of a Gaussian or other such distribution modelling sampling noise, and a uniform (for a naïve prior on the colour of outliers) [Sze99] or other [SFVG04, SFVG06] distribution modelling the likelihood of occlusion and any other form of outlier, the result being an energy cost function that is robust to outliers. There is effectively an implicit, latent, visibility variable for each input sample [BR96], whose value is determined by photoconsistency.

A more successful approach to modelling occlusions has been to make explicit the visibility variables, and determine these either geometrically, by warping the depth map into each input view [KZG03, SLKS05, WQ05, BG07], or by constructing a generative colour model of occluding surfaces [SFVG04, SFVG06]. One can also apply a smoothness prior to visibility across each input image [SLKS05]. Optimization of visibility is achieved either in an EM algorithm, iterating between optimizing visibility and optimizing depth [SFVG04, SFVG06, SLKS05], or synchronously with depth [KZG03, WQ05, BG07]. While the latter approach is less prone to fall into local minima, it has been shown to be practically limited to optimization using graph cuts [KR06a].

Modelling occlusions, and other outliers, in this principled way not only improves the quality of results, but has also been shown to reduce the need for a surface smoothness prior [WQ05].

3.2.3 Backward transfer NVS

In addition to regularizing the explicit geometry of a reference view, which can be used for forward transfer NVS, some stereo algorithms have been repurposed to the task of regularizing the implicit geometry of an output image in backward transfer NVS. As the PDE-based multi-resolution stereo algorithm of Strecha *et al.* [SFVG04] explicitly reconstructs a noiseless version of the reference image, the authors convert the algorithm simply by setting the reference image to be a new view; a colour-based occlusion model is used to reject outliers. Criminisi *et al.* [CSB+07] use a scanline-based algorithm which does model occlusions geometrically, but is limited to two input views, and enforces the ordering constraint on correspondences. Both these algorithms use first order priors on smoothness.

3.3 Image priors

Image priors model the likelihood of images—images here meaning natural images, *i.e.* photographs of real world scenes—using models and parameters measured or learned *a priori*, *i.e.* before solving the problem at hand. They generally do this by modelling the local statistics of images, that is to say the statistics of patches within images, and as such can generally be written as a product of functionals over cliques of pixels, such as the following MRF formulation:

$$p(\mathbf{I}) = \frac{1}{\zeta} \prod_{\mathcal{N} \in \mathbb{N}} \psi\left(I(\mathcal{N})\right), \tag{3.5}$$

where each clique, represented by the neighbourhood of pixels, $\mathcal{N}$, is a sub-window, or patch, of the image, $\mathbf{I}$, and the set of all cliques, $\mathbb{N}$, is generally the set of all overlapping patches in the image. Note that in equation 3.5 the clique functional, $\psi(\cdot)$ does not differ between cliques—this gives rise to translational invariance of the prior, a feature common to most image priors.

The relative computational tractability and efficacy of the optimization of first order priors over higher order priors has, similar to geometry priors, encouraged a wealth of priors of this form. However, image texture can also exhibit higher order textural structure, as demonstrated by the patches in figure 3.4(b), which is not well modelled by first order priors. This has given rise to many higher order priors also, modelling the spatial statistics of image patches (figure 3.4(c) indicates that histogram-based models are not suitable), and consequently showing a greater capability to distinguish natural texture from noise or errors. In the next section I review the models used for pairwise priors (plus some other, related low-order priors), and in following sections I describe the three main types of higher order prior (typically $|\mathcal{N}| > 3$): filter-based, sparse coding and example-based.

3.3.1 Pairwise priors

The majority of pairwise image priors are derivative priors—priors on the magnitude of the colour difference between neighbouring pixels, the neighbourhoods generally being the set of all 1×2 and 2×1

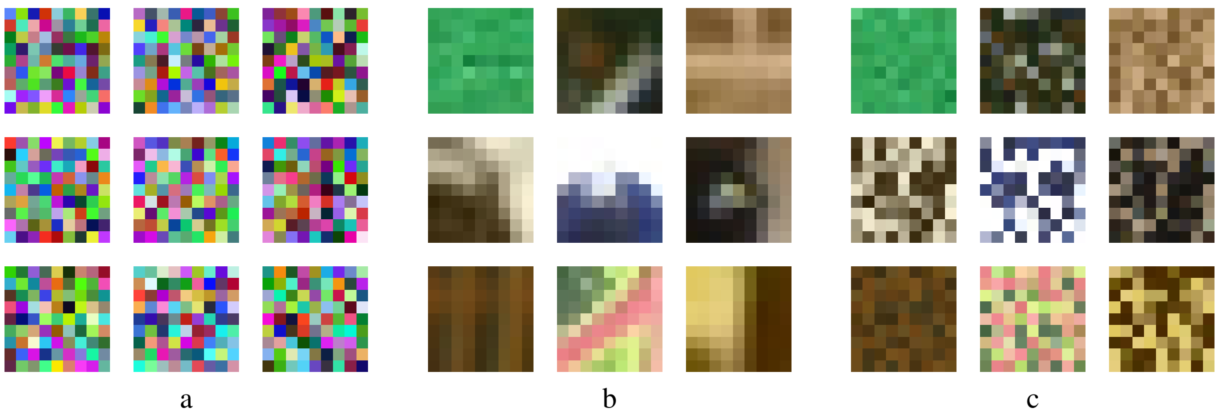


Figure 3.4: Random *vs*. natural image patches. (a) Patches drawn randomly from the space of all possible image patches. (b) Natural image patches manually selected from the test images of §2.5 to show texture commonly found in natural images. (c) The patches from (b) with their pixels rearranged in a random order. Colour histograms for these patches will exactly match the histograms for their partner patches in (b).

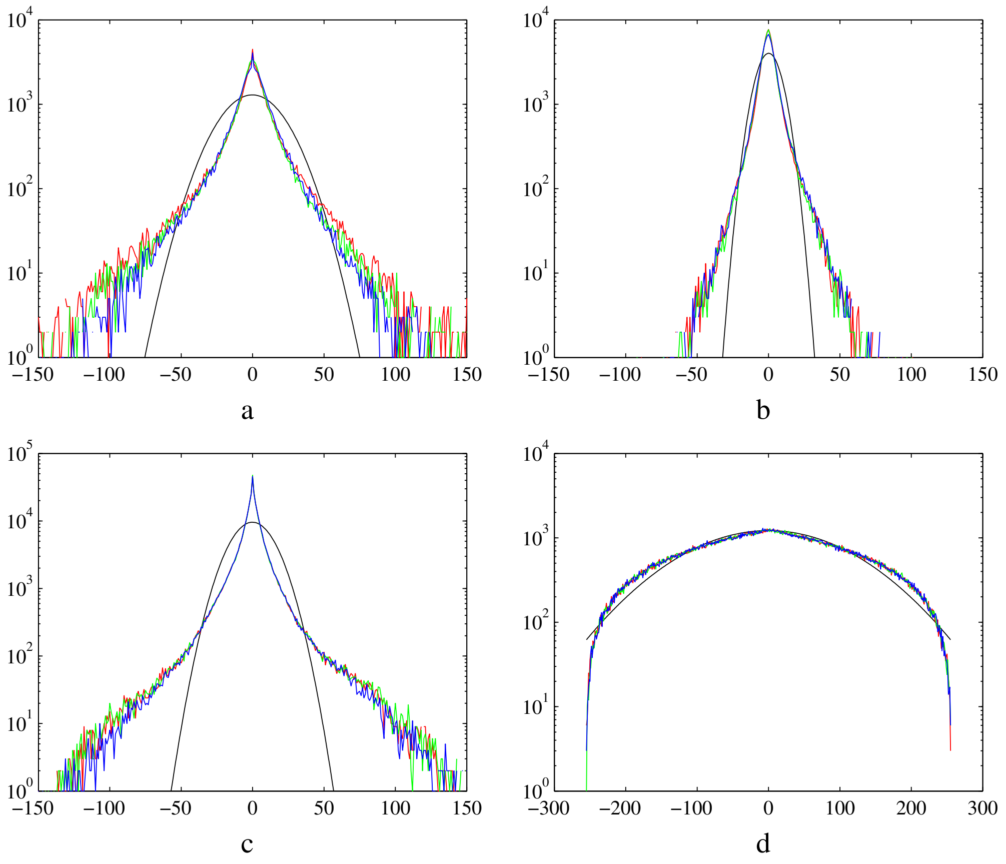


Figure 3.5: Image derivative statistics. Histograms of first order image derivatives (horizontal and vertical combined) for each colour channel (indicated by the corresponding line colour) for each of the three images in figure 2.4(a): (a) Plant, (b) Monkey and (c) Edmontosaurus, and (d) for a 400×400 random image, similar to the patches shown in figure 3.4(a). The Gaussian distribution that best fits the mean histogram over all channels is also shown (in black).

patches in an image (known as the 4-connected neighbourhood), defined by

$$p(\mathbf{I}) = \frac{1}{\zeta} \prod_{\mathbf{x}\in\mathcal{X}} \psi_{\mathrm{h}} \left(\|I(\mathbf{x}) - I(\mathbf{x} + [1,0]^{\top})\| \right) \cdot \psi_{\mathrm{v}} \left(\|I(\mathbf{x}) - I(\mathbf{x} + [0,1]^{\top})\| \right). \tag{3.6}$$

Often the horizontal and vertical clique functionals, ψ_{h} and ψ_{v}, are the same, giving the prior rotational invariance, as well as translational. The earliest such priors used Gaussian clique functionals, *e.g.* [MS85], penalizing the square of the intensity gradient, which, as with the thin membrane model for geometry, over-smooths colour discontinuities.

It was later shown that the statistics of derivatives, indeed of any derivative-like filter, of natural images are not at all Gaussian [Fie87, Sim97]—they're highly kurtotic[9] as shown in figure 3.5. Parametric, highly peaked, heavy-tailed distributions have therefore more recently been used as the form of the clique functionals [LZW02, TRF03, FSH$^+$06], in order to match the statistics of natural images, creating what is sometimes called the sparse derivative prior. Being concave rather than convex[10] such priors tend to encourage intensity gradients to concentrate in small areas, rather than to spread out, preserving texture discontinuities while generally encouraging piecewise constant colour. Levin *et al.* [LZW02] used a clique functional of the form $\psi(x) \propto \exp(-x^{0.7})$, with an additional, quadruple clique term modelling the likelihood of intensity corners, for the purpose of extracting from a superposition of two images the original images, using BP for inference. They then used lookup-table-based functionals on both derivatives and also gradient angles (necessitating the use of triple cliques), learned directly from the input data, to inpaint holes in images using BP [LZW03], giving the results shown in figure 3.6. While the pairwise priors perform well at continuing the simple image structure of figure 3.6(b), they struggle to reproduce the texture seen in figure 3.6(d). Tappen *et al.* [TRF03] used the same form of clique functional as [LZW02] (without the corner term) for single image super-resolution and Bayer pattern demosaicing, again using BP for inference. Levin & Weiss [LW04] extended [LZW02] for the purpose of separating reflections and shadows from images, approximating the sparse clique functional with a mixture of two Laplacian distributions. This approximation makes the partition function of the posterior tractable, allowing simultaneous optimization of the prior parameters and the output images in an EM framework. Fergus *et al.* [FSH$^+$06] use a zero mean Mixture of Gaussians (MoG) model for the clique potentials in their blur kernel estimation framework, also giving a tractable partition function, which is required for

[9]Kurtotic is an adjective used to describe distributions with a high degree of kurtosis, this being the sharpness of a distribution's peak relative to its tails, measured as the fourth moment of the distribution divided by its squared variance.

[10]As most kernels are symmetric about zero their convexity is determined over the range $(0, \infty)$.

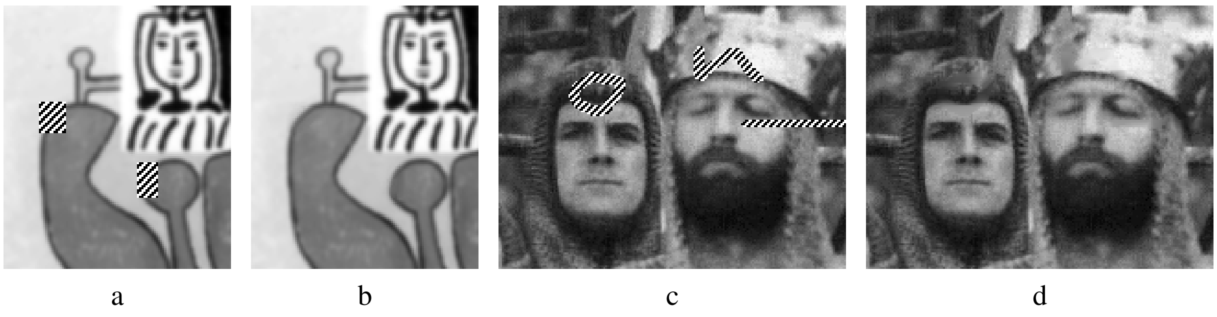


Figure 3.6: Inpainting using a sparse derivative prior. (a,c) Input images, with regions to be inpainted shown with diagonal black & white stripes, and (b,d) the corresponding output images, inpainted using the method of Levin *et al.* [LZW03], using a prior on image derivatives and also gradient angles. Images reproduced from [LZW03].

their variational Bayes approach,[11] using coordinate descent for inference.

Two pairwise alternatives to the derivative prior are to use clique functionals that are functions over both colours in each clique, *i.e.* take into account the mean colour as well as the difference, and also to use neighbourhoods that are arbitrarily connected, *i.e.* pixels in a given neighbourhood are not necessarily adjacent to each other. Gagalowicz & Ma [GM85] use both approaches in their texture synthesis algorithm, modelling textures with a description vector made up of a set of 2-d histograms, one for each type of two pixel neighbourhood, defined by a unique offset $[i, j]^\top$ between the two neighbours, up to a certain size. The energy of a texture image is given by the squared error of its description vector from that of the training texture, which can be written as

$$E(\mathsf{I}) = \sum_i \sum_j \sum_s \sum_t \left(|\mathcal{X}| \cdot H_{ij}(s,t) - \sum_{\mathbf{x} \in \mathcal{X}} [I(\mathbf{x}) = s] \cdot [I(\mathbf{x} + [i,j]^\top) = t] \right)^2 , \qquad (3.7)$$

where i,j are pixel offsets, s,t are intensity values, $[\cdot]$ is the Iverson bracket, and H_{ij} is the empirical, normalized 2-d distribution of values for a given neighbourhood (defined by offset $[i, j]^\top$), learned from a training texture. This approach was later simplified [ZVG00] by reducing the number of neighbourhoods used to a sparse set, and using only 1-d histograms on intensity difference. Note that the approach is not an MRF formulation, but a prior on a global statistic, which creates a single clique encompassing the whole image. Cremers & Grady [CG06] used a related approach in an MRF framework for denoising binary (*i.e.* two tone) textures, relaxing the global constraint to a set of much weaker local constraints.

[11] Variational Bayes [Mac03, Chapter 33] approaches do not seek to find the maximum of the joint posterior distribution, as per MAP approaches, but rather to model the posterior distribution itself. Variables are then set to either the mean or mode of their marginal probabilities, computed from the posterior.

All these arbitrarily connected pairwise models have been limited to texture denoising and synthesis, as the pairwise statistics for specific textures are much more constrained than for natural images as a whole, which may contain a range of textures drawn from a huge set. However, recent work [LH08] has used arbitrarily connected triple cliques for image denoising.

While pairwise (first order) priors can clearly model piecewise smooth images well, they are not suited to modelling the complex textures generally found in natural images (see figure 3.4) as figure 3.6(d) shows. In order to distinguish natural images from the unnatural, random images of figure 3.4(a), whose pairwise cliques are in themselves plausible in natural images, but for their frequency (as evinced in figure 3.5, a property not taken into consideration by MRF models), the pairwise models must heavily penalize colour discontinuities, regardless of whether they form part of some plausible larger scale image structure. The result is regularization that overly smooths images and fails to reconstruct complex texture. As a result, many researchers have turned to MRFs that incorporate higher order models of images.

3.3.2 Filter-based priors

The first type of higher order image priors to appear is what shall be referred to as "filter based"—the clique functional of equation 3.5 can be written as a product of M subfunctions, $f_m : \mathbb{R} \to \mathbb{R}^+$, over responses to a set of corresponding filters, $\mathbf{J}_m$, thus

$$\psi\left(I(\mathcal{N})\right) = \prod_{m=1}^{M} f_m\left(\mathbf{J}_m^\top \overrightarrow{I(\mathcal{N})}\right), \tag{3.8}$$

It can be seen that the pairwise derivative priors described above are actually a specific class of filter-based prior in which the filters are the numerical first derivative filters. Geman & Reynolds [GR92] extended this by using second and third order derivative filters to regularize the reconstruction of blurry or noisy images. They again employed a concave kernel, theirs of the form $f(x) \propto \exp(1/(1+|x|))$, in their clique functionals to preserve discontinuities, encouraging images of constant intensity gradient or constant gradient variation, depending on the derivative used. Second derivatives were additionally used in two of the sparse derivative prior methods described above [LZW02, LZW03].

However, as the filters become larger, so too does the range of potential filters, and the question arises as to which filters should be used. Zhu *et al.* [ZWM98] attempted to address this with their

FRAME framework, which selects a small number of filters from a larger, hand-crafted set, by greedily minimizing entropy. These filters can be up to 33×33 pixels in size, *i.e.* $|\mathcal{N}| = 1089$. Rather than using parameterized subfunctions, a discrete histogram is learnt for each filter, again by minimizing entropy (which turns out to be equivalent to ML learning), such that each f_m is a lookup table. Not only is inference, which is based on Gibbs sampling, in this framework very slow, but the results on image denoising [ZM97] are well below average.

3.3.2.1 Products of Experts

Hinton [Hin99] noted that functions, such as the subfunctions, f_m, of equation 3.8, which are multiplied together can model concomitant constraints—one near-zero value from a subfunction will make the whole clique probability small. This means that each subfunction can be an expert in a single feature that marks the input data as *unlikely*. In the case of filter-based priors, the filter response is the feature, indicating that the filters, $\mathbf{J}_m$, of equation 3.8 should each be crafted to recognize a feature that is unlikely. In the previous filter-based methods, filters had always been selected on the basis that they recognize features that are likely. In his approach, which he called "Products of Experts", Hinton proposed to learn the experts themselves, thus solving the problem of filter selection. However, as discussed in §2.3, when model parameters are learnt, the partition function must be taken into account, and, as discussed in §2.3.1, when a distribution is made up of other distributions multiplied together, the partition function for the complete distribution is not generally tractable. For this reason, Hinton developed a gradient-ascent-based approach to ML learning which avoids explicit computation of the partition function by approximating the gradient of likelihood w.r.t. the model parameters, using a technique he called "contrastive divergence" [Hin02].

A PoE model was first used in an image prior by Welling *et al.* [WHO02]. They modelled the probability of natural image *subwindows*, with a single clique encompassing the whole subwindow, such that the filters, $\mathbf{J}_m$, were the size of the subwindow. It is well known that the frequency response of an arbitrary linear filter of zero mean,[12] applied to natural images, is highly kurtotic [Bad86]. For this reason the distribution over each subspace was modelled using Student's t-distribution (a parametric, kurtotic

[12]The authors of [WHO02] subtract the mean from each of their training patches to avoid learning a D.C. offset, producing a set of derivative like filters.

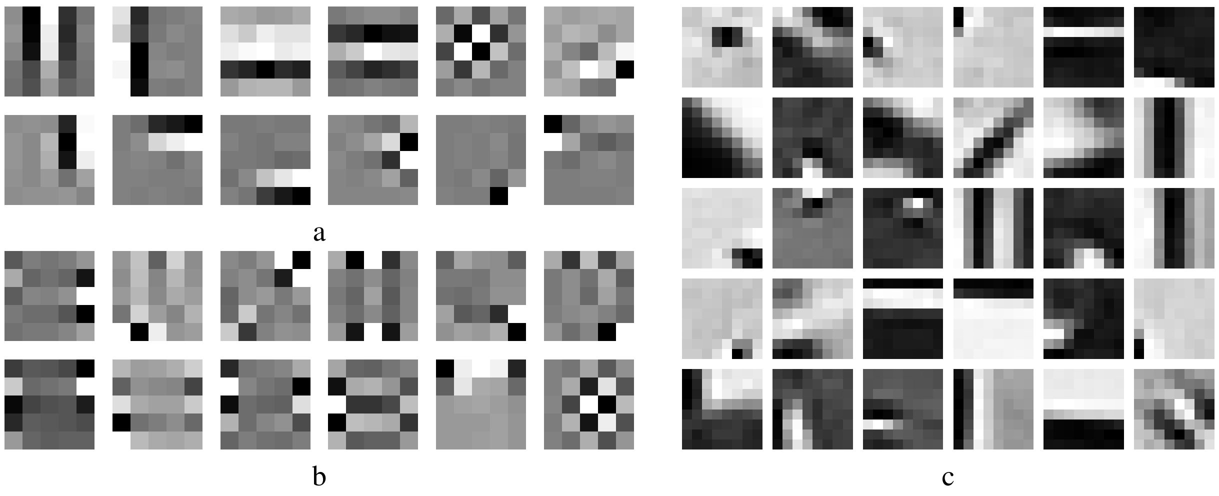


Figure 3.7: Learned filters. A selection of the linear filters of the (a) PoE model of [WHO02], and (b) the FoE model of [RB05], learned on greyscale 5×5 and 15×15 patches respectively, reproduced from [RB05], and (c) a selection of the sparse coding bases learned, using the method of [AEB06], on greyscale 8×8 patches.

distribution), producing the following joint probability model:

$$p\left(I(\mathcal{N})\right) = \frac{1}{\zeta} \prod_{m=1}^{M} \left(1 + \frac{1}{2}\left(\mathbf{J}_m^{\top} \overrightarrow{I(\mathcal{N})}\right)^2\right)^{-\alpha_m}, \tag{3.9}$$

with $\Theta = \{\mathbf{J}_m, \alpha_m\}_{m=1}^{M}$ being the parameters to be learned. The authors learned 99 experts with a 10×10 filter (and therefore neighbourhood) size, and used this model to denoise larger images in an MRF framework, by approximating $\psi\left(I(\mathcal{N})\right)$ of equation 3.5 with $p\left(I(\mathcal{N})\right)$ given above. As discussed in §2.3.1, this approximation cannot be assumed to be valid. Roth & Black [RB05] addressed this issue by giving $\psi\left(I(\mathcal{N})\right)$ the identical form as $p\left(I(\mathcal{N})\right)$ in equation 3.9, then learning the parameters, Θ, for a set of 24 5×5 experts by maximizing the likelihood of a set of 15×15 training *images*, rather than patches, in what they call the "Field of Experts" (FoE) framework. The resulting model generated cutting edge results in image denoising, and also showed impressive results in image inpainting, given the general nature of the prior.[13] Figure 3.7 demonstrates the difference in filters learned using the PoE and FoE models of [WHO02] and [RB05] respectively, showing that there is some difference in the filters and therefore the shape of the two resulting distributions. This suggests that $p\left(I(\mathcal{N})\right)$ is a poor approximation to the optimal $\psi\left(I(\mathcal{N})\right)$ of equation 3.5. Improvements have since been made to the learning of Θ for both models [WF07], by using a zero mean MoG model in place of the t-distribution,

[13]Most inpainting algorithms use a prior trained on a very similar image or texture to that being inpainted, while the FoE model is learned over a varied set of natural images.

allowing an approximation of ζ to be quickly computed, and also by rotating the coordinate frame of a given set of filters, a transformation which does not change ζ, in order to further improve the likelihood of the training set.

The filters of all the above PoE and FoE models are learned on intensity (*i.e.* single channel) images. They have been used to denoise and inpaint colour images [RB05], by treating each colour channel as an independent image. An attempt has been made to learn an FoE model explicitly over colour images [MCSF06], but the computational complexity of the full learning procedure for the higher-dimensional filters required was sufficiently costly that the authors reverted to hand-selected filters, based on a *principal components analysis* (PCA) decomposition of image patches. Despite that, they showed improved denoising results by exploiting the correlation between colour channels.

3.3.2.2 Inference

A range of inference algorithms have been used to optimize problems using these priors, amongst them SA [GR92], Gibbs sampling [ZWM98] (for sampling from the posterior distribution, rather than finding the maximum) and gradient descent [ZM97, RB05, WF07]. In the PoE model of [WHO02], inference was achieved by optimizing the probability of each overlapping output patch independently, using an iterative adaptation of Wiener filtering, then averaging the results. This approach can be considered to be employing a mean field approximation to the MRF probability given in equation 3.5. All these approaches are prone to find quite poor, local minima, but as discussed in §3.1, more powerful inference techniques are limited to smaller cliques. For this reason the FoE model was also learned on 2×2 cliques [LRHB06], allowing the resulting energy to be optimized using BP [LRHB06, Pot07].

3.3.3 Sparse coding priors

Sparse coding priors [AEB06, EA06, OF97] are based on the observation that an image patch can generally be represented by the superposition of a sparse set of basis patches modelling common image features. Such a model is believed to be a potential strategy for image formation in the human visual system [OF97]. Regularization can therefore be achieved by minimizing the difference between an image patch and its optimal reconstruction from the basis patches [EA06], producing the following clique functional

$$\psi\left(I(\mathcal{N})\right) = \exp\left(-\min_{\boldsymbol{\alpha}_{\mathcal{N}}}\left(\|\mathrm{F}\boldsymbol{\alpha}_{\mathcal{N}} - \overrightarrow{I(\mathcal{N})}\|^2 + \mu\|\boldsymbol{\alpha}_{\mathcal{N}}\|_0\right)\right), \tag{3.10}$$

where F is a matrix with basis patches along the columns and $\boldsymbol{\alpha}_{\mathcal{N}}$ is a vector of weights for the basis patches which reconstruct the output neighbourhood, $\mathcal{N}$. The vector $\boldsymbol{\alpha}_{\mathcal{N}}$ is a latent variable that must be optimized in the MAP estimation, and the second term,[14] $\|\boldsymbol{\alpha}_{\mathcal{N}}\|_0$, regularizes this variable, making it as sparse as possible; μ is a fixed weight on the regularization.

A useful aspect of the sparse coding framework is that, even though the partition function is intractable, it can safely be ignored because without it the optimal parameters will not revert to some trivial solution (unlike the filter-based priors, whose learned filters would become a vector of zeros), so learning the basis patches, F, is much easier. Aharon *et al.* [AEB06] develop an iterative learning framework which updates F and $\boldsymbol{\alpha}_{\mathcal{N}}$ (for the training data) simultaneously, using a method they call K-SVD. They generate an overcomplete basis, of which a subset of patches are shown in figure 3.7(c), demonstrating the qualitative difference between the sparse coding patches, which represent likely image features, and FoE filters, which distinguish unlikely features.

A downside to using the sparse coding prior for inference is that there are extra parameters to optimize in the energy minimization. Inference for the non-differentiable objective function given above is achieved using an EM algorithm [EA06], whereby each of the $\boldsymbol{\alpha}_{\mathcal{N}}$ are optimized first using a greedy algorithm, then $I(\mathcal{N})$ is optimized, and the process is repeated until convergence. The prior has been applied to both image denoising [EA06] and image compression [AEB06] with excellent results. Interestingly, for denoising, the authors of [EA06] also tried simultaneously learning F on the noisy image as they denoised it, and for reasonable noise levels this approach proved as effective as using the pre-learned basis, more so for scenes with a high-level of uncommon texture.

3.3.4 Example-based priors

Example-based priors are so called because the clique functionals of equation 3.5 are a based on a set of experts, each of which is a function of the difference of the output image neighbourhood from an exemplar patch. These priors might also be referred to as "Sum of Experts" (SoE) models, in contrast to the PoE models described earlier, as the responses of the experts are summed (or the maximum value output) rather than multiplied. The clique functional will therefore return a high response as long as at least one of the experts returns a high response, allowing the experts to be expert in different features, each of which make the input data *likely*. This contrast with the experts in unlikely features of PoE

[14]The L0 operator defines $0^0 = 0$, and is more strictly written as $\lim_{p\to 0} \|\mathbf{X}\|_p^p$. It is therefore equal to the number of non-zero entries in the vector.

models is summarized well by Welling [Wel07]:

> "Metaphorically speaking, a single expert in a [SoE model] has the power to pass a bill while a single expert in a [PoE model] has the power to veto it".

Clique functionals of example-based priors can generally be written as

$$\psi\left(I(\mathcal{N})\right) = \sum_{m=1}^{M} f_m\left(\left(\boldsymbol{\mu}_m - \overrightarrow{I(\mathcal{N})}\right)^{\top} \Sigma_m \left(\boldsymbol{\mu}_m - \overrightarrow{I(\mathcal{N})}\right)\right), \tag{3.11}$$

with $\boldsymbol{\mu}_m$ being a mean vector for a given expert, Σ_m being a covariance matrix for the expert and $f_m : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ being the response function of the expert. As each expert is modelling a high probability part of the space of patches, an intuitive way of achieving this is to have each vector, $\boldsymbol{\mu}_m$, be a particular example of a likely image patch, and have the expert functions encourage lower inputs.

One of the earliest example-based priors, by Popat & Picard [PP93], was a MoG model for use in texture synthesis, compression and classification. The model was learned on a set of $n \times n$ (coinciding with the size of neighbourhood, $\mathcal{N}$) image patches from an input texture, by clustering the patches into M clusters, then independently computing the parameters for each Gaussian based on the patches in each cluster, but limiting Σ_m to a diagonal matrix. Rather than *learning* Gaussian experts, two later texture synthesis algorithms, by Efros & Leung [EL99] and Wei & Levoy [WL00], introduced the idea of using Gaussian kernel density estimation to model the distribution, based on a large set of exemplar patches; each exemplar patch is the mean, $\boldsymbol{\mu}_m$, of an expert, Σ_m is some fixed value which applies weights to pixels of each patch (weighting centre pixels more in [EL99], uniform, *i.e.* $\Sigma_m = \mathtt{I}$, in [WL00]). This not only negates the need for learning, but, by using many more experts, also allows a much richer distribution to be modelled, especially in regions of patch-space where there are very few exemplars. Both these approaches [EL99, WL00] also replaced the sum of equation 3.11 with a maximum, which is then able to go inside the exponent (and minus sign, becoming a minimum) of the kernel, generating the following clique functional:

$$\psi\left(I(\mathcal{N})\right) = \exp\left(-\min_{m\in\{1,..,M\}} \left(\boldsymbol{\mu}_m - \overrightarrow{I(\mathcal{N})}\right)^{\top} \Sigma_m \left(\boldsymbol{\mu}_m - \overrightarrow{I(\mathcal{N})}\right)\right). \tag{3.12}$$

Using this form of functional over that of equation 3.11 has several consequences. Computation is faster (for a given M) as the exponent is now computed only once per clique, and it can also be accelerated by using tree structures to find the closest exemplar [WL00]. The distribution's partition function is not tractable, and its shape is much flatter (note that $I(\mathcal{N})$ identical to an uncommon exemplar has the same likelihood as $I(\mathcal{N})$ identical to a very common exemplar), but since in an MRF the optimal $\psi(I(\mathcal{N}))$ is not necessarily the same shape as $p(I(\mathcal{N}))$, the effect of this is difficult to judge.

The clique functional of equation 3.12 has since been used, often with slight variations, in many other applications. Freeman *et al.* [FPC00, FJP02] use the prior for single image super-resolution, by matching band-passed patches of an upsampled input image to exemplars for which high-frequency versions are known; the corresponding high-frequency exemplars are then integrated into the output image, with the additional constraint that the overlapping boundaries of these exemplars match well. Hertzmann *et al.* [HJO$^+$01] use the prior to transform an output image based on an input image and a pair of images which represent an analogous transformation, basing the exemplars on concatenated patches (luminance only) from the analogous pair. Fitzgibbon *et al.* [FWZ05] use the prior for NVS, constructing a library, $\mathbb{T}$, of exemplars from all the patches in the input images. As with [WL00, FJP02], they set $\Sigma_m = \mathtt{I}$, allowing the clique functional to be expressed more familiarly as

$$\psi\left(I(\mathcal{N})\right) = \exp\left(-\min_{T\in\mathbb{T}} \|\overrightarrow{T} - \overrightarrow{I(\mathcal{N})}\|^2\right). \tag{3.13}$$

A drawback of this approach is its high computational cost, given a large $\mathbb{T}$ and lack of patch search acceleration.

Example-based priors have also been successfully applied to large scale (*i.e.* large hole) inpainting [CPT03], and constrained texture synthesis [KEBK05]. Criminisi & Blake [CB04] demonstrate near realtime, cyclopean view synthesis, based on the work of [FWZ05], but constraining the texture library, $\mathbb{T}$, for each output pixel independently, to be the patches that straddle the corresponding epipolar lines of that pixel in the input images; in addition, it allows output patches to be split in two, using an alpha matte, and each half to be regularized independently, for improved performance over discontinuity boundaries. These priors have even been extended to 3-d, for use in video completion [WSI04] and solid texture synthesis [KFCO$^+$07], demonstrating the flexibility of this model. While not itself a prior, the "Image Epitome" [JFK03] provides a means to compute and efficiently store a compact, redundancy-free set of experts (exemplar patches, $\boldsymbol{\mu}_m$, and diagonal covariance matrices, Σ_m) learned from a larger, redundant set of input patches, in the form of a small image—the experts are all the overlapping patches in the image.

3.3.4.1 Inference

Where the MRF forms a regular four-connected grid [FPC00, FJP02], then optimization of these priors using BP is feasible. However, in most cases the neighbourhoods overlap much more; for example, in the case of the 5×5 patches used in [FWZ05], each clique contains 25 variables, each of which are in 24 other cliques also, creating a problem that it is computationally infeasible to solve using a global inference method. Several algorithms therefore approximate the MRF of equation 3.5 with a Bayesian network [PP93, EL99, WL00, HJO+01, FJP02, CPT03][15]:

$$p(\mathsf{I}) = \prod_{\mathbf{x}\in\mathcal{X}} p\left(I(\mathbf{x})|I(\mathcal{N}_\mathbf{x})\right), \tag{3.14}$$

where $\mathcal{N}_\mathbf{x}$ is a neighbourhood of pixels around $\mathbf{x}$ that appear earlier than $\mathbf{x}$ in the list, $\mathcal{X}$, of all pixels, thus allowing sequential (*e.g.* in raster scan order [PP93, WL00, HJO+01, FJP02], or order determined at runtime [EL99, CPT03]), optimal assignment of values to variables. A benefit of this approach, which was discussed in §2.3.1, is that the conditional probabilities extracted directly from a joint distribution learned over patches of the same size will have the correct shape, removing the need to learn clique functionals on images rather than patches (which no patch-based method does anyway).

Those algorithms which do not approximate the problem are forced to use local inference methods, generally finding quite poor local minima. Fitzgibbon *et al.* [FWZ05] use an ad hoc algorithm loosely based on ICM, while others [JFK03, KEBK05], noticing that pixels sharing a clique are completely decoupled from each other given the expert (allowing the decomposition of figure 3.1), use an EM-style algorithm.

Multi-resolution frameworks have been employed in many of these algorithms [PP93, WL00, HJO+01, KEBK05], not only to improve the minima found, but also to reduce the effect of the ordering of pixels in the Bayesian Network methods, and to allow smaller patches at each scale for the same effective regularization as a large patch at a single scale, making the algorithms more efficient.

[15]In fact, several of these approaches are ad hoc, and do not actually refer to the MRF [WL00, HJO+01, CPT03] I attribute to them, while [PP93] explicitly seeks to model their problem using the Bayesian network.

3.4 NVS state of the art

I conclude with a recap of the current state-of-the-art priors for NVS. The majority of priors for NVS regularize geometry, either that of a known reference view [Sch96, Sze99, ZK07] for forward transfer NVS, or that of the output view itself [SFVG04, CSB$^+$07]. The priors used are universally first order, either encouraging fronto-parallel surfaces [Sch96, Sze99, SFVG04, CSB$^+$07] or enforcing a piecewise-planar reconstruction [ZK07], neither of which are accurate models for real world scenes. Optimization methods for the backward transfer methods are PDE [SFVG04] or scanline-based [CSB$^+$07], creating the problem of weak inference.

Recently, image priors have been used to regularize the texture of the output image in backward transfer NVS [FWZ05, CB04]. While a promising approach, these priors require large cliques in order to distinguish between natural and unnatural texture, and are hampered by the optimization of the resulting objective functions.

The aim of the work presented in this thesis is to develop NVS (and stereo) frameworks that both differentiate the natural from the unnatural, *i.e.* regularize well, and are tractable, *i.e.* optimize well.

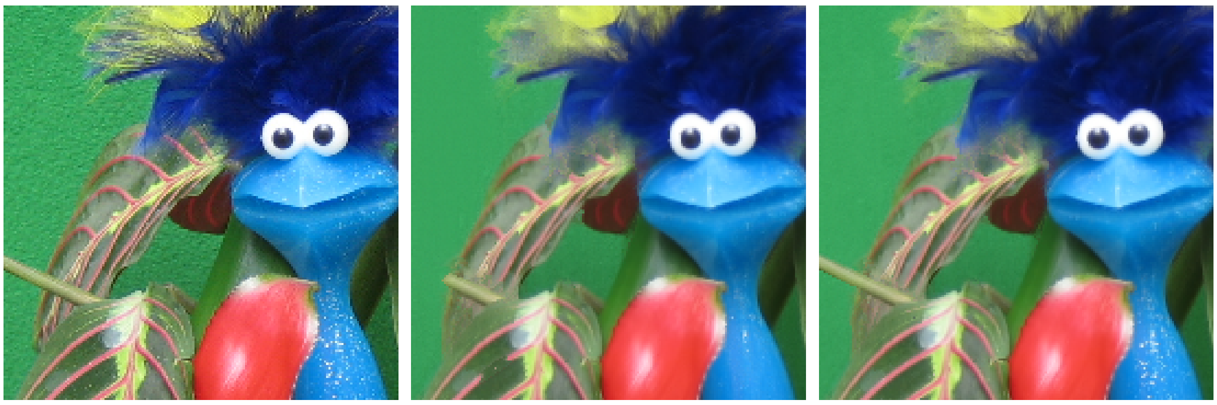
Chapter 4

A hierarchical texture prior

4.1 Introduction

This chapter is concerned with efficient optimization of the non-parametric texture prior introduced by Fitzgibbon *et al.* [FWZ05]. As discussed in §3.3.4, this higher order, example-based prior has a good ability to distinguish between natural and unnatural texture, but suffers from two drawbacks: current inference techniques can only find a local solution, requiring that the initial estimate to be close to the optimal solution, and inference is extremely slow, due to the high cost of evaluating the texture energy.

I investigate three different improvements to the optimization algorithm used in [FWZ05], evaluating



(a) Ground truth image. (b) Single resolution approach—1900s to render. (c) Multi-resolution approach—58s to render.

Figure 4.1: Multi-resolution gains. (a) Ground truth for the plant sequence. (b) Image rendered in 1900 seconds using a single resolution approach – the image has an r.m.s. error of 12.3, and 8.83% gross pixel errors from ground truth. (c) Image rendered using a multi-resolution approach, in only 58 seconds – the image has an r.m.s. error of 10.4, and 7.45% gross pixel errors.

their impact on speed and efficacy. I also investigate how a multi-resolution strategy can be employed to further overcome both drawbacks of this prior, by improving the quality of solution found, as well as accelerating the optimization, as shown in figure 4.1.

In §4.2 I outline the objective energy and optimization algorithms used at each scale of the algorithm, and compare these algorithms in a single resolution framework. In §4.3 I describe how scale-space is employed to reduce the problem size, and how the construction of an exemplar hierarchy can be used to constrain the size of the texture library at the finer scales of the multi-resolution approach. Qualitative and quantitative evaluations of the algorithms' performance are presented, and comparisons made between the single and multi-resolution framework.

4.2 Computational strategy: single resolution

This section introduces the objective energy and optimization algorithms used in a single resolution strategy, which will then form the building block of the multi-resolution algorithm.

4.2.1 Objective function

First I will present the energy that is to be minimized, and the way in which the problem state-space is discretized.

4.2.1.1 True objective energy

The energy functional of the non-parametric prior of Fitzgibbon *et al.* [FWZ05] is a prior on image texture, which regularizes only the output image, and is defined as

$$E_{\text{prior}}(\mathsf{I}_0^*, \mathsf{D}) = E_{\text{texture}}(\mathsf{I}_0^*) = \sum_{\mathcal{N} \in \mathbb{N}} \min_{T \in \mathbb{T}} \|\overrightarrow{T} - \overrightarrow{I_0^*(\mathcal{N})}\|^2, \tag{4.1}$$

where $\mathcal{N}$ is the list of output pixels in an $n \times n$ patch, such that $\overrightarrow{\mathsf{I}_0^*(\mathcal{N})}$ is a vector of the colours of those pixels and $\mathbb{N}$ is the set of indices of all overlapping $n \times n$ patches in I_0^*, and $\mathbb{T}$ is a texture library containing all the overlapping $n \times n$ patches in the set of input images, $\overrightarrow{T}$ being a vectorized patch from this library. This is exactly the form of the prior used in this chapter.

For the sake of computational speed I assume that sampling noise is Gaussian, and that there will be no sampling outliers due, for example, to occlusions or specularities. This gives the data likelihood term the following form:

$$E_{\text{photo}}(\mathsf{I}_1, .., \mathsf{I}_N | \mathsf{I}_0^*, \mathsf{D}) = \lambda_{\text{d}} \sum_{\mathbf{x} \in \mathcal{X}_0} \sum_{i=1}^{N} \| I_i\left(\pi_i\left(\mathbf{x}, D(\mathbf{x})\right)\right) - I_0^*(\mathbf{x}) \|^2. \tag{4.2}$$

where λ_{d} is a parameter that weights the influence of the data likelihood term relative to the prior, and can be considered to be a noise parameter. The total energy is that defined by equation 2.12.

4.2.1.2 Minimizing over disparity

It can be seen that disparity appears only once in the objective function, in equation 4.2, and that the value $D(\mathbf{x})$ affects only the data likelihood of the pixel at $\mathbf{x}$. The authors of [FWZ05] use this fact to minimize out D from the problem in a costly pre-processing step, which is discussed in chapter 6, giving a set of colour modes to choose from for each pixel. In this case the problem becomes one of finding the I_0^* that globally minimizes $E(\mathsf{I}_0^*)$.

In this chapter I do the opposite, which is to minimize out colour (given by I_0^*), and formulate the problem as one of optimizing over disparity. The reasons for this are two-fold. Firstly, the multi-resolution strategy employed makes use of the disparity map, D. Secondly, given the Gaussian noise model, it is orders of magnitude faster to do this in a pre-processing step rather than to minimize out disparity. In fact, this improvement in efficiency was first made by Yao & Cham [YC04] in their own adaption of the NVS algorithm of [FWZ05], but, as disparity does not otherwise factor in their method, they take only the modes of E_{photo} over disparity, rather than the whole distribution.

It is trivial to show that the colour, $I_0^*(\mathbf{x})$, that minimizes E_{photo} at a pixel, $\mathbf{x}$, given a disparity $D(\mathbf{x})$, is given by the mean of input samples, thus:

$$I_0^*(\mathbf{x}) = \frac{1}{N} \sum_{i=1}^{N} I_i\left(\pi_i\left(\mathbf{x}, D(\mathbf{x})\right)\right). \tag{4.3}$$

I assume that the palette of colours computed over all disparities is sufficient to reconstruct the image, I_0^*, which minimizes the overall energy, $E(\mathsf{I}_0^*, \mathsf{D})$, and therefore parameterize colour by the function of disparity given above. An interesting implication is that although the optimization is over the disparity map, D, the objective is to find the D that generates the correct I_0^*, rather than to find the correct D itself.

4.2.1.3 Disparity discretization

The disparity space is discretized so that discrete optimization algorithms can be used in the inference stage—these can avoid some of the local minima found by continuous-space algorithms. This is done by setting a minimum and maximum disparity for the image (d_{min} and d_{max} respectively), either defined by the user or computed from the 3-d point correspondences found by the *structure from motion* algorithm used to calibrate the input images—the disparities of these points in the coordinate frame of $\mathbf{I}_0^*$ can be computed, and some factor[1] of the minimum and maximum amongst these used as bounds. The disparity range is then regularly discretized into N_{L} disparity labels, where N_{L} is chosen such that the maximum distance between neighbouring disparity labels for any given pixel when projected into any of the input images is no greater than half a pixel in the horizontal and vertical directions.

The label given to pixel $\mathbf{x}$, $L(\mathbf{x}) \in \{0, .., N_{\text{L}} - 1\}$, therefore defines the disparity at $\mathbf{x}$ thus:

$$D(\mathbf{x}) = d(L(\mathbf{x})) = d_{\text{min}} + L(\mathbf{x})\frac{d_{\text{max}} - d_{\text{min}}}{N_{\text{L}} - 1}. \tag{4.4}$$

The set of labels over all pixels forms a labelling, L.

While the search spaces when optimizing over $\mathbf{I}_0^*$ and D are $\mathbb{R}^{\text{channels}\times\text{width}\times\text{height}}$ and $\mathbb{R}^{\text{width}\times\text{height}}$ respectively, the latter being much smaller, once these spaces have been discretized in a pre-processing step the latter approach generally generates an order of magnitude more labels per pixel. However, the increase in computation time caused by the increased label count is negligible in comparison to the time saved in generating the labels (using the method of [FWZ05]).

4.2.1.4 Approximate objective function

As in [FWZ05, YC04], the objective function has been transformed from one over the continuous, joint space of colour and disparity, to one over discrete labels:

$$E(\mathbf{I}_0^*, \mathsf{D}|\mathbf{I}_1, .., \mathbf{I}_N) \longrightarrow E(\mathsf{L}|\mathbf{I}_1, .., \mathbf{I}_N). \tag{4.5}$$

For the continuous values of $\mathbf{I}_0^*$ and D which map onto a labelling, L, the energies are identical; however, only a tiny proportion of possible values of $\mathbf{I}_0^*$ and D will have such an exact mapping. The assumption

[1] In this work d_{min} and d_{max} are computed as 0.8 and 1.2 times the minimum and maximum disparities of the point correspondences respectively.

made here is that the minimum of $E(\mathsf{L}|\mathbf{I}_1,..,\mathbf{I}_N)$ is close to the minimum of $E(\mathbf{I}_0^*, \mathsf{D}|\mathbf{I}_1,..,\mathbf{I}_N)$.

It is possible to cache the values of colour and E_{photo} for each of the labels at each pixel, using equation 4.3 and equation 4.2 respectively. This generates the data structures $\mathbf{C}$ and $\mathbf{E}_{\text{p}}$, where

$$C(\mathbf{x}, l) = \frac{1}{N}\sum_{i=1}^{N} I_i\left(\pi_i\left(\mathbf{x}, d(l)\right)\right) \tag{4.6}$$

$$E_{\text{p}}(\mathbf{x}, l) = \lambda_{\text{d}}\sum_{i=1}^{N} \|I_i\left(\pi_i\left(\mathbf{x}, d(l)\right)\right) - C(\mathbf{x}, l)\|^2 \tag{4.7}$$

The objective function of equation 2.12 can now be phrased as the following function over L:

$$E(\mathsf{L}|\mathbf{I}_1,..,\mathbf{I}_N) = \sum_{\mathbf{x}\in\mathcal{X}_0} E_{\text{p}}(\mathbf{x}, L(\mathbf{x})) + \sum_{\mathcal{N}\in\mathbb{N}} \min_{T\in\mathbb{T}} \|\overrightarrow{T} - \overrightarrow{C(\mathcal{N}, L(\mathcal{N}))}\|^2 \tag{4.8}$$

where $\overrightarrow{C(\mathcal{N}, L(\mathcal{N}))}$ is the vector of colours of the neighbourhood of pixels, $\mathcal{N}$, given the neighbourhood's labelling, $L(\mathcal{N})$. The labelling, L, effectively defines a slice through the colour cube, $\mathbf{C}$, shown in figure 4.2, such that the output image, $\mathbf{I}_0^*$, can be written as

$$\mathbf{I}_0^* = C(\mathcal{X}_0, \mathsf{L}). \tag{4.9}$$

Figure 4.2 provides visualizations of the two cached data structures, $\mathbf{C}$ and $\mathbf{E}_{\text{p}}$, showing that, while E_{photo} values in textured areas, *e.g.* along the top cube face, show clear minima, minima in textureless areas, *e.g.* on the right side cube face, are more ambiguous.

4.2.2 Optimization

Now I will describe the algorithms used to minimize the energy defined above.

4.2.2.1 Algorithms

In the section above the problem was defined as one of choosing a disparity label for each pixel in $\mathbf{I}_0^*$. It can be seen from equation 4.1 that the objective function contains cliques of size $n \times n$ (where n is typically greater than two) such that, as n increases, it quickly becomes impossible to cache the values of E_{texture} for all labellings (${N_{\text{L}}}^{n^2}$ labellings) of a given clique, let alone use a global inference algorithm to minimize the energy.

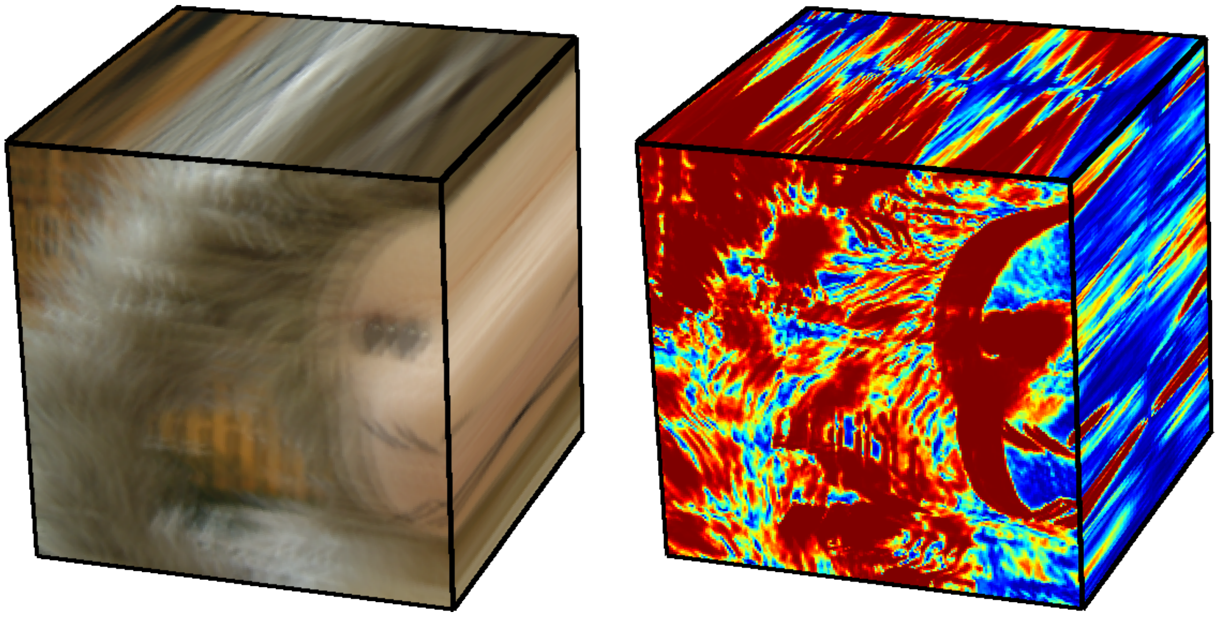


Figure 4.2: Cached C and E_p . Visualizations of the two cuboid data structures used in my objective function, equation 4.8—C (*left*), containing colour values for each output pixel at each disparity, and E_p (*right*), containing values of E_photo for the same. E_p has been coloured blue for low values through to red for high values and the scale has been altered for improved visualization.

As such, Fitzgibbon *et al.* [FWZ05] employ an ad hoc, discrete inference algorithm which can be viewed as an approximation of both ICM [Bes86] and EM [DLR77]. Briefly, their approach iterates two steps: 1) for each overlapping $n \times n$ patch, excluding its centre pixel, the closest patch in the library is found, then 2) each pixel is simultaneously updated to the label which is closest to a linear combination of the previous iteration's colour (which approximates E_photo) and the colour of the centre pixel of the closest library patch (or *exemplar*) centred on that pixel (found in the previous step). Drawbacks of this approach are its synchronous updates, which can lead to oscillations between labellings, the fact that only one of the n^2 cliques each pixel is a factor in is used to update its label, allowing updates to actually increase the energy, and the way in which E_photo is merely approximated.

In this chapter I will compare three inference approaches, all of which differ from that above, but which offer improvements in certain respects. These algorithms are as follows:

Iterated Conditional Modes (ICM): The label of each output pixel is updated sequentially, such that the energy, $E(\mathsf{L}|\mathbf{I}_1, .., \mathbf{I}_N)$, conditioned on that pixel, is minimized, thus

$$L(\mathbf{x}) = \underset{l \in \{0,..,N_\mathrm{L}-1\}}{\operatorname{argmin}} \; E(\{L(\mathcal{X} \backslash \mathbf{x}), l\} | \mathbf{I}_1, .., \mathbf{I}_N), \tag{4.10}$$

where the backslash operator indicates that the set to the left of the operator excludes the value to the right, *e.g.* $\mathcal{X}\backslash\mathbf{x}$ is the set of all pixels, excluding the pixel $\mathbf{x}$. This cycle over all pixels is repeated until convergence (to a local minimum). In practice the conditional E_{texture} energy need only be computed for those patches which overlap the pixel being conditioned on, but this still requires n^2 patch searches per pixel update, compared to the one patch search per pixel of the inference approach of [FWZ05]; however, all the drawbacks of the latter approach are overcome.

Approximate ICM (AICM): As per ICM, except that the conditional energy is approximated by being conditioned only on the texture patch centred on $\mathbf{x}$, defined by the neighbourhood $\mathcal{N}_{\mathbf{x}}$, so that the update equation becomes

$$L(\mathbf{x}) = \operatorname*{argmin}_{l\in\{0,..,N_{\text{L}}-1\}} E_{\text{p}}(\mathbf{x},l) + n^2 \min_{T\in\mathbb{T}} \|\overrightarrow{T} - \overrightarrow{C(\mathcal{N}_{\mathbf{x}}, \{L(\mathcal{N}_{\mathbf{x}}\backslash\mathbf{x}), l\})}\|^2. \tag{4.11}$$

The E_{texture} term is multiplied by the number of cliques that pixel $\mathbf{x}$ is in, so as to approximate the true conditional texture energy generated by the n^2 cliques. This approach is the most similar, of those employed here, to that of [FWZ05], differing in that it updates pixels sequentially, ensuring conditional energies are calculated on an up-to-date labelling, that the colour of each patch's centre pixel in E_{texture} is taken into consideration in the patch search, and that the true value of E_{photo} is used. As this approach is not guaranteed to converge, the iterations are halted if the sum of conditional energies defined by equation 4.11 rises from one iteration (*i.e.* cycle through all pixels) to the next.

Expectation Maximization Style (EMS): This approach follows [FWZ05], and also the texture synthesis methods of [KEBK05, KFCO+07], in first finding the exemplar which minimizes E_{texture} for each output image patch, in the E-step, then simultaneously choosing the label of each pixel which minimizes the energy, in the M-step, given the exemplars found in the E-step, in an EM-style[2] algorithm. A new, latent data structure, B, is created, which stores, for every pixel, $\mathbf{x}$, the exemplar[3] $\overrightarrow{B_{\mathbf{x}}}$, found in the E-step for the output patch centred on $\mathbf{x}$, thus:

$$\overrightarrow{B_{\mathbf{x}}} = \operatorname*{argmin}_{T\in\mathbb{T}} \|\overrightarrow{T} - \overrightarrow{C(\mathcal{N}_{\mathbf{x}}, L(\mathcal{N}_{\mathbf{x}}))}\|^2. \tag{4.12}$$

[2]Since exemplar indices are not continuous variables, the ML value is computed in the E-step, rather than the expected value, as discussed in §3.1.3.2, making this algorithm EM-style ICM.

[3]In reality, B stores, in $B(\mathbf{x})$, the exemplar's index into the patch library, from which the exemplar, $\overrightarrow{B_{\mathbf{x}}}$, can easily be reconstructed.

The M-step for a given pixel is defined as

$$L(\mathbf{x}) = \underset{l \in \{0,..,N_\text{L}-1\}}{\operatorname{argmin}} E_\text{p}(\mathbf{x}, l) + \sum_{\mathbf{y} \in \mathcal{N}_\mathbf{x}} \|B_\mathbf{y}(\mathbf{x}) - C(\mathbf{x}, l)\|^2, \tag{4.13}$$

where $B_\mathbf{y}(\mathbf{x})$ is the pixel of the patch $\overrightarrow{B_\mathbf{y}}$ which maps onto pixel $\mathbf{x}$ in the output image. The method stops when L does not change from one iteration to the next. In contrast to the approach of [FWZ05], the E-step includes the centre pixel of each patch, and the M-step uses every patch that each pixel overlaps with, as well as an accurate value for E_photo.

4.2.2.2 Implementation details

The initial labelling chosen is that which minimizes E_photo at each pixel, thus:

$$L(\mathbf{x}) = \underset{l \in \{0,..,N_\text{L}-1\}}{\operatorname{argmin}} E_\text{p}(\mathbf{x}, l). \tag{4.14}$$

In all three algorithms the label of a pixel depends only on a small number of cliques, such that the pixel need only be updated if any of those cliques were updated in the previous iteration. This saves considerable computation time, as the number of pixels which need updating generally decreases rapidly with each iteration. In addition, the data structure, B, introduced for the EMS method, is actually recorded in all three optimization methods, so that it can be used in the multi-resolution framework.

I set $\lambda_\text{d} = n^2/N$, making the energy minimized invariant to both the number of input images used and the size of the patch used for regularization. A more reasoned approach to setting the weight could be used, such as learning the optimal value [LL08], but this is not the focus of this work. I use the same patch size as that used by Fitzgibbon *et al.* [FWZ05], $n = 5$.

4.2.3 Results and discussion

Figure 4.3 shows the qualitative results on the four test sequences of the three energy optimization algorithms used, while figure 4.4 gives some quantitative measures of their performance. The qualitative results tend to bear out the following: that ICM produces the best results, visually, followed by EMS, then AICM. This can be seen most clearly (where highlighted) under the arm in Monkey, on the lower left leaf in Plant, the cabinet door frame in Dino1 and the brickwork in Dino2. The exception to this trend is the dinosaur's snout in Dino2, which is reconstructed most pleasingly by AICM.

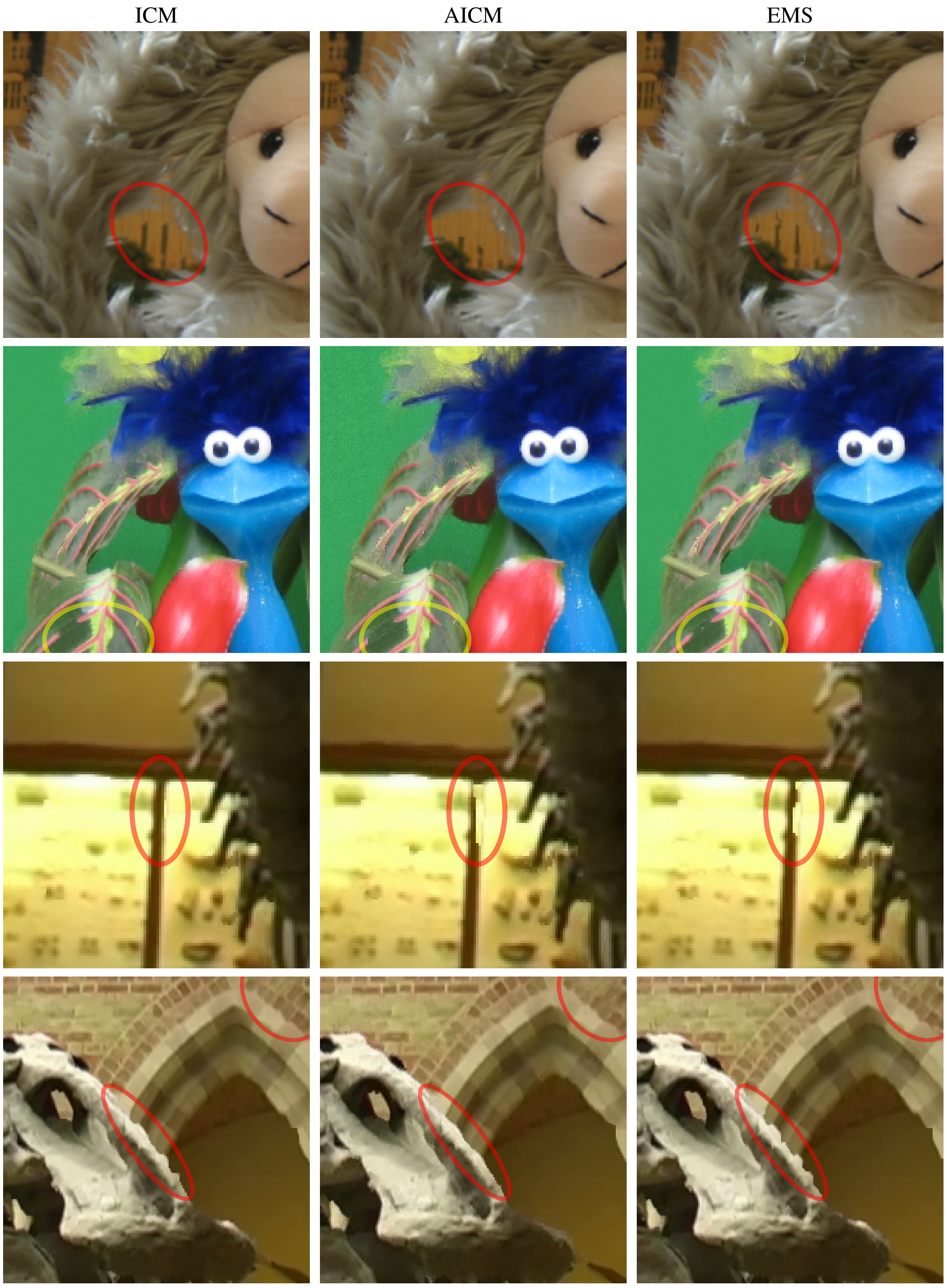
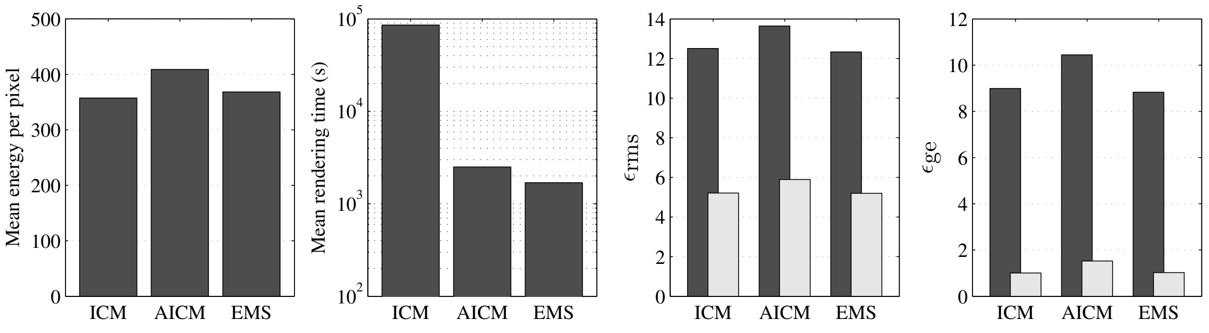


Figure 4.3: Single resolution output images. Each row contains output images for one of the test sequences, with the three columns containing the results of the different optimization algorithms, labelled top. Differences across each row are highlighted with ellipses.



(a) Energy (b) Rendering time (c) R.m.s. error (d) Gross errors (%)

Figure 4.4: Quantitative results. Quantitative results for the three optimizers for (a) final energy per pixel, (b) rendering time, (c) r.m.s. error and (d) gross errors. Results for (a) & (b) are averaged over the four test sequences, while results for (c) & (d) are given separately for the Plant (dark grey) and Monkey (light grey) sequences.

All the algorithms tend to fail to some degree in those areas (with the exception of ICM on the cabinet door frame), as well as the background baize (texture is lost), the stalk and leaf just to the left of the head in Plant, and the arch in Dino2, and all these areas tend to coincide with large scale errors in the ML output (shown in figure 2.4). These large scale errors are generated either by occlusions in some of the input images, by fine details surrounded by large homogenous regions, in regions of low contrast, stochastic texture, or simply by noisy data (as in the case of Dino2). These large scale errors tend to form local minima in E_{texture}—patches in the middle of an incorrect region look plausible, so changing one pixel to the correct colour will generate a high frequency artifact, while pixels at the boundary are in equilibrium from the pressure from both correct and incorrect regions to flip colour. As a result, the optimization algorithms presented struggle to correct large scale errors.

The quantitative results, shown in figure 4.4, suggest a slightly different ordering, in terms of quality of reconstruction (based only on the sequences with ground truth), with EMS just pipping ICM in both overall r.m.s. error and gross errors, and AICM some way behind. However, the aim of these optimization algorithms is to minimize energy, and in this regard ICM again performs best, with EMS performing 3% worse overall, and AICM performing 14% worse.

In terms of speed, EMS is the fastest algorithm, with AICM 47% slower, and ICM is by far the slowest algorithm, at 50 times slower than EMS. Therefore, while AICM can be rejected as being both slower and a poorer minimizer than EMS, there is a trade-off to be made between speed and efficacy when selecting between EMS and ICM. As the focus of this chapter is predominantly one of efficiency, I will accept the small drop in quality in favour of a 50 times speed-up, and use EMS by preference in the next section.

4.3 Computational strategy: multi-resolution

The single resolution algorithms of the previous section are extremely slow, taking between 0.05 and 2.5 seconds per output pixel to run to completion. Given that all the other data is cached, the step which takes up the vast majority of this time is the patch lookup of the E_{texture} term. It is also possible for the solutions of these approaches to be far from the global optimum if the initialization itself is far from the optimum, as the optimization algorithms used are very local in nature. In this section I use a multi-resolution framework that helps to avoid poor local minima, and directly incorporate into it a novel approach to speeding up the patch search.

4.3.1 Previous work

It is necessary to briefly review previous work in multi-resolution and patch search acceleration, in order to provide a point of reference for the contributions made in this section.

4.3.1.1 Multi-resolution

Multi-resolution approaches have been used in Computer Vision since at least the work of Marr & Poggio [MP79]; indeed, they are extremely common in Computer Vision algorithms which employ example-based priors [PP93, PL98, WL00, WSI04, LH05, KEBK05, KFCO$^+$07]. The general strategy is to start with a low resolution problem, solve this and upsample the solution in order to initialize a higher resolution (finer scale) problem, which is then solved, and so on until the desired resolution is reached. The reasons for using such a framework in these applications are four-fold:

- **Better optimum** – there are fewer optima at lower resolutions, so poor optima are more likely to be avoided here and a good optimum propagated up to initialize the next scale, where an optimum close to the initialization is found. Hence a good optimum can be propagated up through all the scales.
- **Faster convergence** – long distance interactions are propagated faster at lower resolutions, then propagated up through the scales, leading to improved initializations and therefore faster convergence at the finer scales.

- **Smaller patches** – the job of a large patch in enforcing large scale structure can be achieved by a smaller patch at a lower resolution. In this way, coarse structure is added at the coarser scales, while only finer details need be added at the finer scales, using much smaller patches.
- **Smaller search space** – solutions of coarser scales can be used to constrain the search space at finer scales, allowing, for instance, fewer labels to be considered for each pixel.

4.3.1.2 Fast patch search

Much research has gone into speeding up the evaluation time for computing the sum of squared differences (SSD) between a patch and a library of patches. Some exact approaches have been developed in the field of motion estimation for video compression, where accurate evaluation is important; one of the fastest techniques amongst these uses the *fast Fourier transform* (FFT) [KDM02], an approach which has also been employed for texture synthesis [KSE$^+$03]. Greater speed-ups can be achieved where time allows the offline construction of a data structure, such as a k-d tree, to enable efficient search. Nene & Nayar [NN97] give a brief overview of these methods, saying that the majority become intractable in high-dimensional search spaces, and propose a nearest-neighbour algorithm which overcomes this problem. This, along with *approximate* nearest-neighbour (ANN) search [AMN$^+$98], which can perform even faster[4] when the exact closest exemplar is not required, has been used in various exemplar-based Computer Vision algorithms [LLX$^+$01, HJO$^+$01, FJP02, WSI04, KFCO$^+$07]. However, all these search algorithms and structures are general, and do not, in themselves, make use of the properties inherent in image patches.

Pixel values across an image patch are generally highly-correlated, and, as such, the vast majority of information in these patches can be stored in a small number of dimensions, found using PCA. Several patch-based algorithms use PCA [LLX$^+$01, HJO$^+$01, LH05, KFCO$^+$07], then throw away the lowest components in order to lower the dimensionality of the data while retaining a high proportion of the variance; this approach is usually used in combination with one of the search tree algorithms listed above. Patches in an image are generally also highly correlated with each other, such that the libraries of patches themselves can be compressed. Popat & Picard [PP93] developed a cluster-based texture synthesis algorithm which parameterized their exemplar space into a much smaller, *mixture of Gaussians* (MoG)

[4]For a given dimensionality of exemplar, the exact nearest-neighbour algorithm of [NN97] has complexity $\mathcal{O}(n)$ while the approximate method of [AMN$^+$98] has complexity $\mathcal{O}(\log n)$, where n is the number of exemplar points to be searched through.

model with diagonal covariance matrices. Wei & Levoy [WL00] use *tree-structured vector quantization* (TSVQ) for the same problem, to not only given them a reduced-size "codebook" of patches, but also allow fast, approximate search of the codebook through the use of a binary tree (a 2-d k-d tree). Their algorithm uses a multi-resolution framework, and the neighbourhoods used at each scale include pixels from the scale below, exploiting the correlation between patches and their lower-resolution counterparts. Liang *et al.* [LLX$^+$01] exploit the same to speed up ANN search in their realtime, single-resolution[5] texture synthesis framework.

4.3.2 Implementation

In the implementation presented here a multi-resolution framework with S scales is employed. At each scale in the framework an energy similar to that given by equation 4.8 is minimized, using the methods described for the single-resolution approach, but with different data structures for each scale, denoted by the superscript index, s, where $s = S$ is the coarsest (lowest resolution) scale and $s = 1$ is the finest scale. The aim is that, by the final scale ($s = 1$), the initial labelling is closer to the global optimum, and that the texture library for each output patch is constrained to be smaller than the original library, but that the optimal exemplar for a given (optimal) output patch is in the constrained library. The following sections explain the details of this framework.

4.3.2.1 Exploiting scale-space

The image rendered at scale $s + 1$ is a factor of two smaller than that rendered at s in both the horizontal and vertical directions. The downsampling is achieved by removing every second row and column of the finer scale image, such that the centres of pixels at coarser scales map directly onto the centres of pixels at finer scales. This simplifies the interpolation of data from each scale to the next, as only the removed values need to be interpolated.

Given that each coarse-scale output pixel covers a large area in the input images, point samples of the original input images will no longer be representative of the colour of this area as a whole. To overcome this, a Gaussian stack [LH05] is created for each input image—the input image for $s = 1$ is the original input image, while the input image used at each consecutive coarser scale ($s + 1$) is a Gaussian filtered version of the image of the preceding, finer scale. The standard deviation of the filter used here is 1.

[5]Even though the algorithm of [LLX$^+$01] is single-resolution in nature, the input exemplars and query vectors are hierarchically downsampled to reduce their dimensionality and therefore speed up the patch search for that single resolution.

The patches used at each scale remain $n \times n$ in size, despite the downsampling. The texture library for a given scale, $\mathbb{T}^s$, is constructed from the input images at that scale, which are not downsampled. To account for this, pixels with offset $[i, j]$ in an output patch (where $i, j \in 0, .., n-1$) must be compared with pixels offset by $[2^{s-1}i, 2^{s-1}j]$ in the exemplars from the texture library. The reason that the input images are not also downsampled (to generate a Gaussian pyramid instead) is that every fine scale exemplar needs to have an exactly corresponding coarse scale exemplar for the patch search acceleration (described in §4.3.2.2) to work well.

As an $n \times n$ patch covers a much larger area of the output image at a lower resolution, the regularization at coarser scales can correct much larger scale errors in the output image. However, up to this point, no information is being passed from one scale to the next, so the work done by the prior at coarser scales has no impact on the final result. This is resolved by initializing the labelling at each scale but the coarsest with the labelling of the previous scale, rather than using equation 4.14. The labelling of the previous scale must first be upsampled to the correct size, using linear interpolation, then rounded to the nearest label. Not only does the initialization propagate information on larger scale structures up through the framework, but the linear upsampling of the previous labelling also constrains the initial labelling to be smoother, introducing some implicit regularization of scene geometry.

As the input images have been blurred (*i.e.* low-pass filtered) at coarser scales, fewer disparity samples are required to build an accurate table of E_{photo} values, $\mathbf{E}_{\text{p}}^s$. As a result I decimate the disparity levels at each coarser scale by a factor of two from the preceding, finer scale, by removing every second level, such that $N_{\text{L}}{}^s = N_{\text{L}}/2^{s-1}$. This reduction in the number of labels not only accelerates the construction of $\mathbf{E}_{\text{p}}^s$, but also speeds up the patch lookup of the conditional optimization approaches ICM and AICM, as well as, though to a lesser extent, the M-step of the EMS method.

The number of labels can optionally be reduced further, at finer scales, by constraining the disparity labels searched over for each pixel to be around the initialization value generated by the previous, coarser scale. In this mode, which will be called "constrained disparity", the number of labels is fixed to be $N_{\text{L}}/2^{S-1}$ at *every* scale, with the disparity levels at all but the coarsest scale computed individually for each pixel, and centred on the initialization disparity[6] for that pixel. This can increase speed further, but at the cost of fixing disparity in a coarse-to-fine manner, which can lead to errors if the wrong disparity is chosen at a coarse scale.

[6]As label sets are defined per pixel, the labelling generated at the previous scale must first be converted to disparity, then interpolated. However, rounding is no longer required, as the new labels then define offsets from this base initialization disparity.

4.3.2.2 Constraining the search-space

In order to speed up the patch search at each resolution, I follow [WL00, LLX$^+$01] in exploiting the correlation between patches and their lower-resolution counterparts. However, in contrast to [WL00], which uses the same codebook for every output patch at a given scale, the goal of the method presented here is to constrain the set of exemplar patches searched over, for each output patch independently, based on the closest exemplar patch found at the previous scale. This is similar to the approach of Liang *et al.* [LLX$^+$01], who constrain a search for the closest exemplar at full resolution to the k-approximate-nearest-neighbours of the closest exemplar found at a lower resolution (each exemplar being represented at every resolution); however, in our approach the closest exemplar used to constrain the search is at a lower scale (not resolution), having been computed as a by-product of the optimization of the previous scale.

I generate the library $\mathbb{T}^s_{\mathbf{x}}$ specifically for the output patch centred on pixel $\mathbf{x}$ of $\mathbf{I}^{*s}_0$, using the index of the closest exemplar to the same output patch of the previous scale, which will have been stored in $\mathbf{B}^{s+1}$. An "exemplar hierarchy", $\mathcal{H}$, is constructed, which returns, for each exemplar (defined by its index, i) at one scale, a list of indices of exemplars, $H^s(i)$, to search through at the next scale, such that the library $\mathbb{T}^s_{\mathbf{x}}$ is defined as

$$\mathbb{T}^s_{\mathbf{x}} = \left\{ \overrightarrow{T}^s(i) | i \in H^s\left(B^{s+1}(\mathbf{x})\right)\right\}, \tag{4.15}$$

where $\overrightarrow{T}^s(i)$ is the ith exemplar in $\mathbb{T}^s$. The exemplar hierarchy is therefore traversed a level at a time for each output pixel simultaneously, as each scale of the problem is solved. The intertwining of the search-tree structure and the multi-resolution approach in this way is a novel contribution of this chapter.

Each level of the exemplar hierarchy is formed by finding the nearest neighbours of each exemplar at the *previous* scale, then using their equivalent versions at the current scale. The reason for not simply using the nearest neighbours at the current scale is that small differences between coarse scale patches can lead to large differences in their fine scale counterparts, therefore a small error at coarse scale could lead to the required exemplar being missing from $\mathbb{T}^s_{\mathbf{x}}$ at the next scale.

As a small subset of input images from a much larger sequence are used for each output image, the input images, and hence the texture library, will differ from output image to output image. An ANN search tree takes $\mathcal{O}(dn \log n)$ time to create [AMN$^+$98], d and n being the dimensionality and number of exemplars in the library respectively. Given that a single, brute-force closest patch search takes $\mathcal{O}(dn)$

time, and ANN search takes $\mathcal{O}(d \log n)$ time at best[7], if n is much larger than the number of output patches (which it is) then constructing a new search tree for each output image will not improve the situation much—a factor $\mathcal{O}((1/m + 1/n) \log n)$ at best, m being the number of output patches.

The aim is therefore to construct the exemplar hierarchy only once, on the entire image sequence, but one which can rapidly be pruned down to contain only exemplars from the relevant input images. This is achieved by dissociating the exemplars from the hierarchy, and rather have a table, called a "cluster link table", linking a much smaller number of representative exemplars, from which the real hierarchy can quickly be constructed once the input images are selected.

4.3.2.2.1 Clustering The representative exemplars are generated by clustering exemplars from across the entire image sequence, making use of the correlation between patches, in the vein of [PP93, WL00]. This clustering is carried out independently at each scale.

Any clustering technique can be used to cluster the exemplars, but there is usually a trade-off between speed (clustering time) and accuracy, in terms of how often the true nearest-neighbour is found when traversing the exemplar hierarchy. Given that a single image sequence can be used to render many novel views, a high one-off clustering cost is acceptable, so I aim for accuracy and use *k*-means clustering, computing 2^{20-2s} cluster centres at each scale but the finest, for which no clustering is required. I use approximately 20 exemplars per cluster for training at each scale, sampled uniformly over the entire input image sequence. I initialize the cluster centres to random exemplar patches and run EM for 15 iterations, repeating this 5 times and choosing the set of cluster centres which give the lowest mean SSD per patch from its associated cluster centre. The triangle inequality is used to accelerate the *k*-means clustering [Elk03].

PCA is used to project the patches onto their principal components prior to clustering, keeping sufficient dimensions to ensure that the maximum RMS reconstruction error over all training patches is less than a grey-level per pixel. PCA not only reduces the dimensionality of the data, but generally allows termination of SSD computation between cluster centres and training patches after fewer dimensions, as a "not closest" decision can often be made earlier when the majority of information is stored in the first few dimensions, making clustering faster. Only every tenth training patch is used in the PCA analysis.

In addition to the cluster centres, I compute the cluster index image for every input image at every

[7] According to [AMN+98], ANN lookup time is $\mathcal{O}(c \log n)$, where $c \leq d(1 + 6d/\epsilon)^d$, ϵ being the error tolerated distance between actual and approximate nearest neighbours.

scale once, directly after computing the cluster centres. These index images contain, for every patch in the input image, the index of the cluster centre that patch is closest to. The cluster centres exist in the lower dimensional, principal component space, therefore the input image patches are projected into this space before being assigned to the cluster they are closest to (using the L2 norm).

Once the input images have been selected for a given output image, a cluster index image, K^s, for each texture library, $\mathbb{T}^s$ (except the first, as this scale has not been clustered), can be generated from the pre-computed cluster index images of the relevant input images. Figure 4.7 shows a visualization of these index images.

4.3.2.2.2 Cluster link table Clusters contain similar exemplars, so a search of the nearest neighbours of the exemplar $B^{s+1}(\mathbf{x})$ could simply use exemplars in the same cluster. An exemplar hierarchy based on this approach is shown in figure 4.5(a). However, as cluster index is a hard assignment for each exemplar, exemplars which are very similar to $B^{s+1}(\mathbf{x})$ but lie on the other side of a nearby cluster boundary will be excluded from the search. Ideally one would overcome this by linking all neighbouring clusters, *i.e.* generate the Delaunay triangulation of clusters, to form a cluster link table, but in high-dimensional spaces this is intractable. Instead, I do the following: link each cluster to all clusters within 1.5 times the Euclidean distance of its closest neighbour (in the PCA-projected space); then make all one-way links symmetrical. This generates the exemplar hierarchy shown in figure 4.5(b).

The nearest neighbours of $B^{s+1}(\mathbf{x})$ are then approximated as those exemplars in the same cluster, and also those exemplars in clusters linked to the cluster of $B^{s+1}(\mathbf{x})$, thus:

$$H^s(i) = \left\{ J^{s+1}(j) | j \in U^{s+1}\left(K^{s+1}(i)\right)\right\}, \tag{4.16}$$

where $K^{s+1}(i)$ is the cluster index of the i^{th} exemplar, $U^{s+1}(i)$ returns a list of indices of clusters linked to cluster i (including itself), and $J^{s+1}(j)$ returns the list of exemplars which have cluster index j, all at scale $s+1$.

4.3.2.2.3 Upsampling exemplar indices At the coarsest scale the entire texture library $\mathbb{T}^S$ must be used for each output patch, but this library contains as many exemplars (ignoring boundary effects) as the single resolution library, $\mathbb{T}$, making the patch lookup as slow (ignoring the small effect on the conditional ICM and AICM algorithms of having fewer labels) as for the single resolution framework, on a per pixel

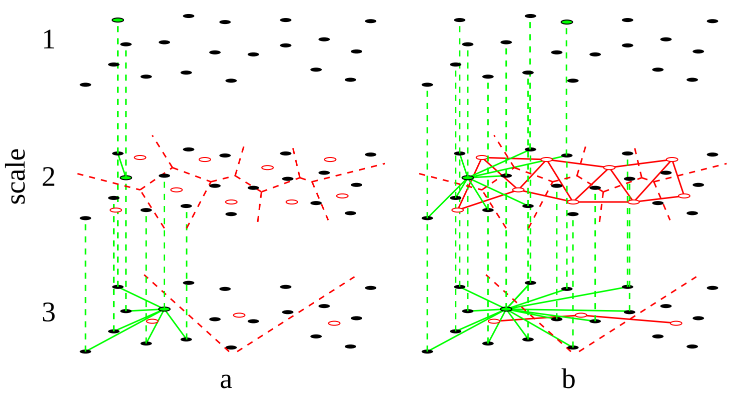


Figure 4.5: Exemplar hierarchy. Two 3-scale hierarchies with different levels of connectivity. (a) Constrained libraries contain all exemplars in the same cluster as the best match from the previous scale. (b) Constrained libraries contain all exemplars in all clusters linked to (and including) the cluster of the closest match.

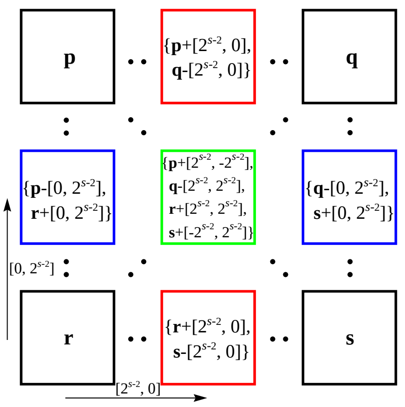

p
{p+[2^{s-2}, 0], q-[2^{s-2}, 0]}
q
{p-[0, 2^{s-2}], r+[0, 2^{s-2}]}
{p+[2^{s-2}, -2^{s-2}], q-[2^{s-2}, 2^{s-2}], r+[2^{s-2}, 2^{s-2}], s+[-2^{s-2}, 2^{s-2}]}
{q-[0, 2^{s-2}], s+[0, 2^{s-2}]}
[0, 2^{s-2}]
r
{r+[2^{s-2}, 0], s-[2^{s-2}, 0]}
s
[2^{s-2}, 0]

Figure 4.6: Upsampling $\mathbf{B}^s$. Indices are extrapolated from nearest known neighbours, and concatenated. Colours signify different neighbourhoods.

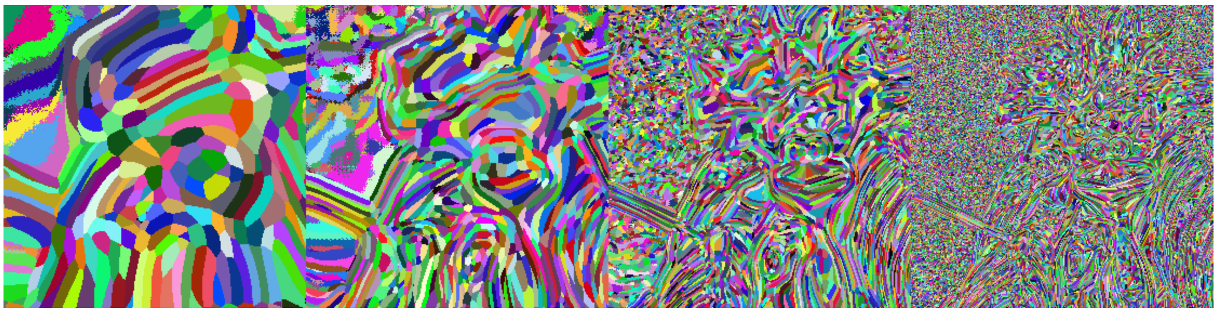


Figure 4.7: Cluster index images. The cluster index images for the first input image from figure 4.10, at scales $s = S$ to $s = 2$ (the finest scale has no cluster index image), with each cluster index represented by a random colour.

basis. Fortunately, due to downsampling, the coarsest scale has 4^{S-1} times fewer pixels. However, a side-effect of the downsampling is that $\mathbf{B}^{s+1}$ is one quarter the size of $\mathbf{I}_0^{*s}$, therefore it needs to be upsampled before it can be used to constrain the texture libraries at the next, finer scale.

Instead of having a downsampled exemplar index image, let us reparameterize $\mathbf{B}^s$ to be an image at the resolution of the finest scale, but with known values only at pixels whose centres map directly onto the centres of downsampled pixels (*i.e.* at locations 2^{s-1} pixels apart in both horizontal and vertical directions). The values in $\mathbf{B}^s$ are integer indices into the space of exemplars, so they cannot simply be interpolated. However, as these values give an index into an image (the texture library) at the same resolution, one can extrapolate an individual index from a known pixel to an unknown one $[i, j]$ pixels away simply by adding $[i, j]$ to the index (which is converted to a location vector first).

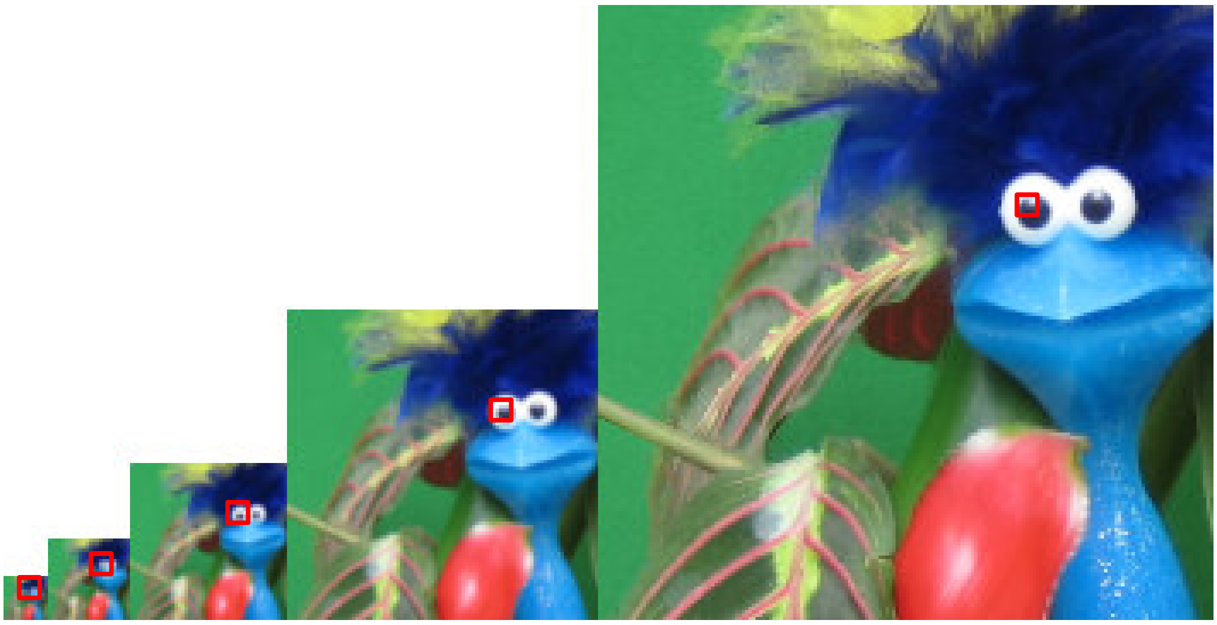


Figure 4.8: Output images $\mathbf{I}_0^{*S}$–$\mathbf{I}_0^{*1}$. Images output at each scale of the multi-resolution framework, from $s = S$ (*left*) to $s = 1$ (*right*), for the plant sequence. The red square in each image indicates the patch used at that scale to demonstrate patch search and the constrained texture libraries in figure 4.9 and figure 4.10.

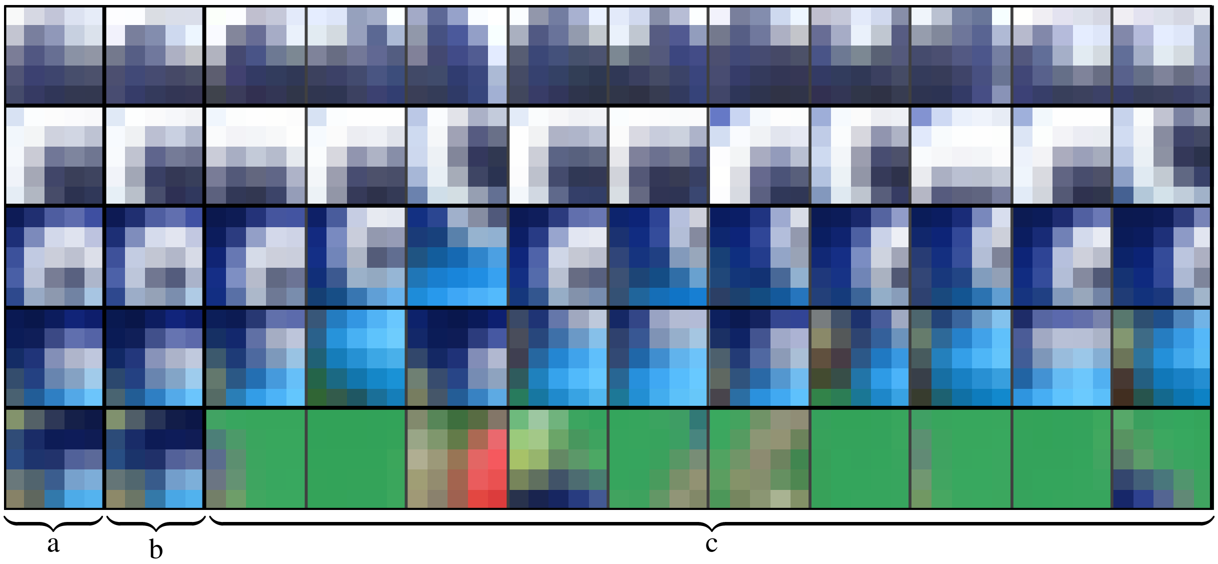


Figure 4.9: Output patches and closest exemplars. Each row contains patches from a given scale, from $s = S$ (*bottom*) to $s = 1$ (*top*). Column (a) contains the output patch from $\mathbf{I}_0^{*s}$ highlighted in figure 4.8. Column (b) contains the exemplar found to be closest to the output patch highlighted in figure 4.8 (at the relevant scale), given by the index $B^s(\mathbf{x})$ (where $\mathbf{x}$ is the location of the output patch in $\mathbf{I}_0^{*s}$), and also highlighted within the texture library, $\mathbb{T}^s$, in figure 4.10. Column (c) contains 10 random exemplars from the constrained texture library, $\mathbb{T}^s_{\mathbf{x}}$, in order to show the variation within the constrained set of exemplars.

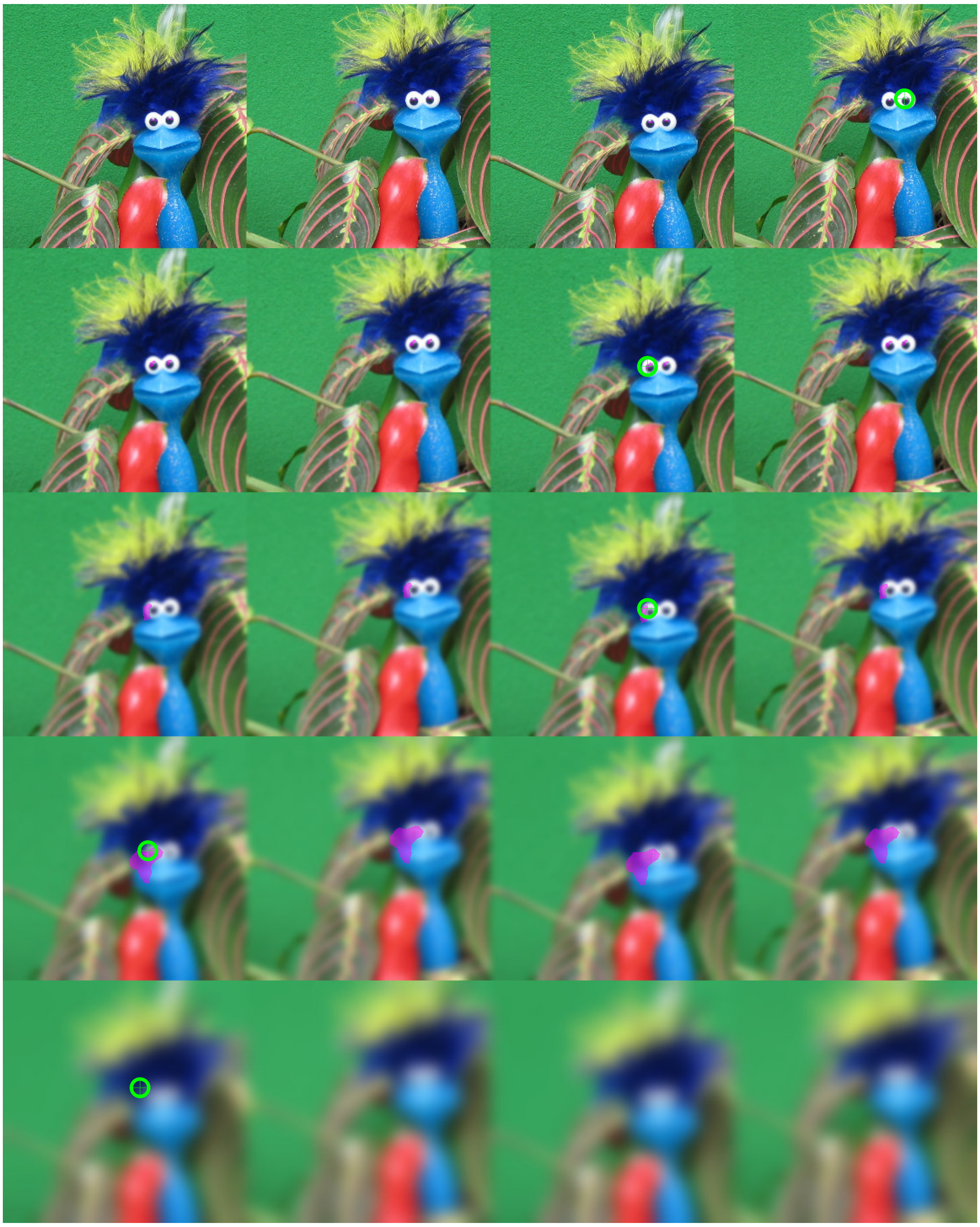


Figure 4.10: Texture libraries $\mathbb{T}^S$–$\mathbb{T}^1$. Each row contains the input images filtered for each scale, from $s = S$ (*bottom*) to $s = 1$ (*top*), which also make up the texture libraries, $\mathbb{T}^s$, at those scales. At each scale, the exemplar found to be closest to the output patch highlighted in figure 4.8 (at the relevant scale), given by the index $B^s(\mathbf{x})$ (where $\mathbf{x}$ is the location of the output patch in $\mathbf{I}_0^{*s}$), is highlighted by a green circle centred on the exemplar. In addition, the exemplars in the constrained texture library, $\mathbb{T}^s_{\mathbf{x}}$, are also highlighted, with magenta dots on their centre pixels (except at the coarsest scale, where all exemplars are in the texture library).

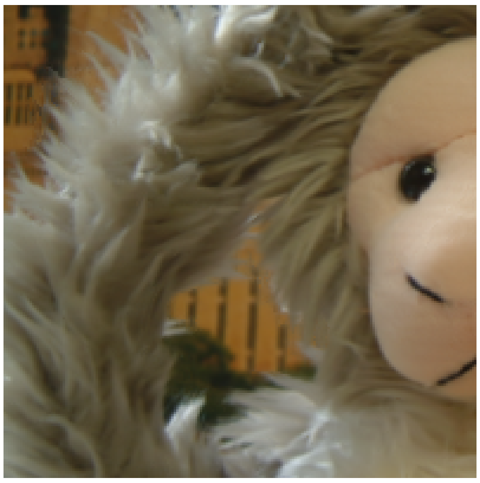
For the locations in $\mathbf{B}^s$ I wish to assign values to, I extrapolate values from the nearest known pixels, and concatenate these into a list of exemplar indices. Figure 4.6 gives an example of this upsampling: the values $\mathbf{p}$, $\mathbf{q}$, $\mathbf{r}$ and $\mathbf{s}$ are the known exemplar indices in $\mathbf{B}^s$, converted to index vectors giving the 2-d location of the centre pixel of the exemplar in the texture library; the values of the blue-bordered pixels are formed by concatenating extrapolated values of the closest known pixels, which are above and below, the values of red pixels from the two pixels either side, and the green pixels from the pixels in the four diagonal directions. Equation 4.15 can then be used to generate the libraries for these interpolated pixels also, accounting for the fact that $B^s(\mathbf{x})$ may contain multiple indices.

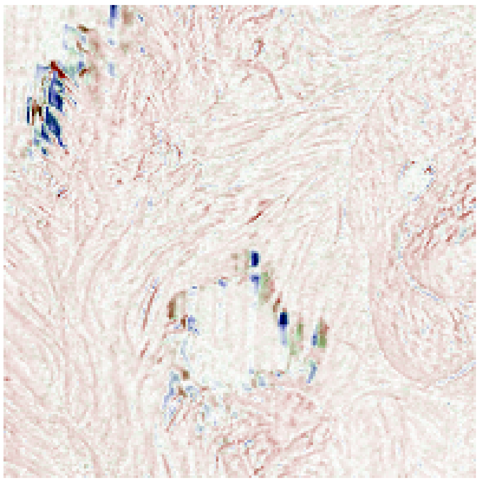
4.3.3 Results and discussion

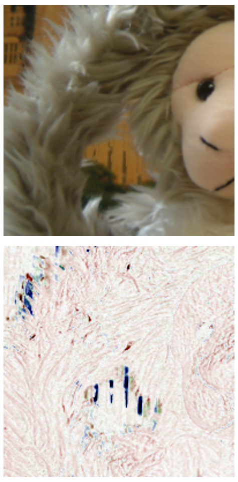
Constrained disparity offers another trade-off between speed and efficacy, as shown in figure 4.15, but this time more finely balanced: a 34% reduction in rendering time brings a 40% increase in energy. The rise in energy is significantly higher in E_{photo} than in E_{texture}, and figure 4.16 suggests the reason for this: discontinuity boundaries are blurred in disparity-space (*e.g.* on the snout), and errors in disparity are propagated up from coarser scales to form larger regions of error at finer scales (*e.g.* in the brickwork), and these disparity errors constrain E_{photo} as a result of the constraint on disparity; however, though the colours have a higher data cost, the texture prior still does a good job of making the output look sensible. The increase in energy using constrained disparity is accompanied by an increase in both measures of visual fidelity (r.m.s. error and gross errors), and for these reasons constrained disparity is not used in generating the following results.

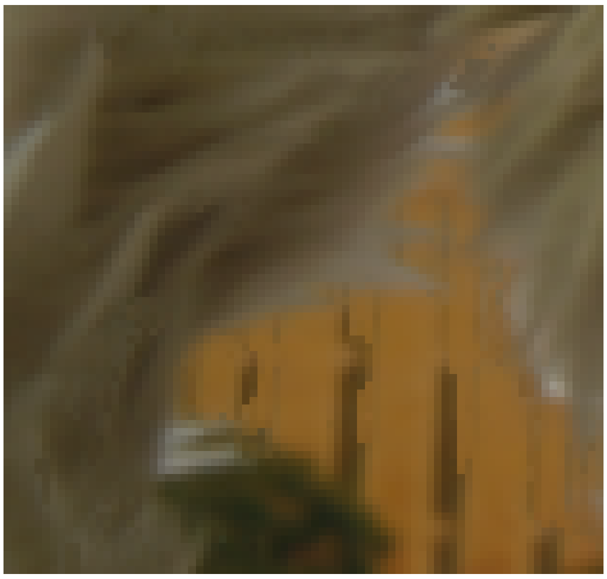
A comparison of qualitative results using the single and multi-resolution frameworks (using EMS and not constraining disparity), shown in figure 4.11, figure 4.12 and figure 4.13, clearly shows an improvement in quality when using the multi-resolution framework. This improvement is most obvious in similar places to those that ICM improved over EMS in the single resolution framework—under the arm in Monkey, on the lower left leaf in Plant, the cabinet door frame in Dino1, and now the snout and arch in Dino2—but the results far surpass those of ICM in the single resolution framework. The multi-resolution approach is much more effective at correcting the larger scale errors seen in the ML images of figure 2.4, and this is due to the regularization at coarser scales working over a larger image area (with the same patch size), and also the fact that there are fewer local minima to get trapped in at these scales.

However, two areas in which the multi-resolution framework still fails are background areas at occlu-


(a) Multi-resolution $\mathbf{I}_0^*$


(b) Diff. of (b) from ground truth


(c) Single res.


(C) Zoom from (c)

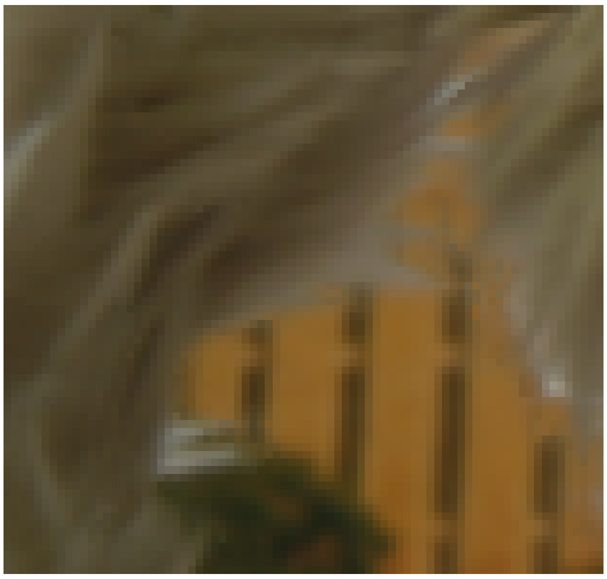
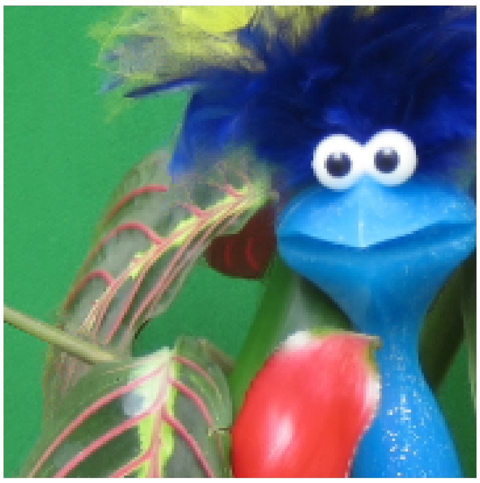
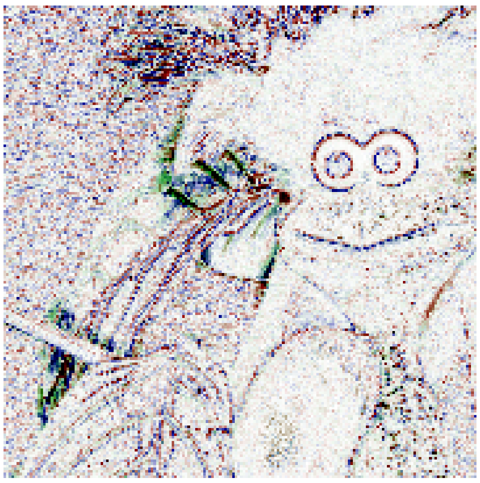

(A) Zoom from (a)

Figure 4.11: Monkey sequence results. (a) Output image, $\mathbf{I}_0^*$, for the Monkey sequence, generated using the EMS algorithm in multi-resolution framework. (b) The differences between (a) and the ground truth image. (c) Results equivalent to (a) & (b) for the single resolution framework. (C & A) Zooms of (c) and (a) respectively.

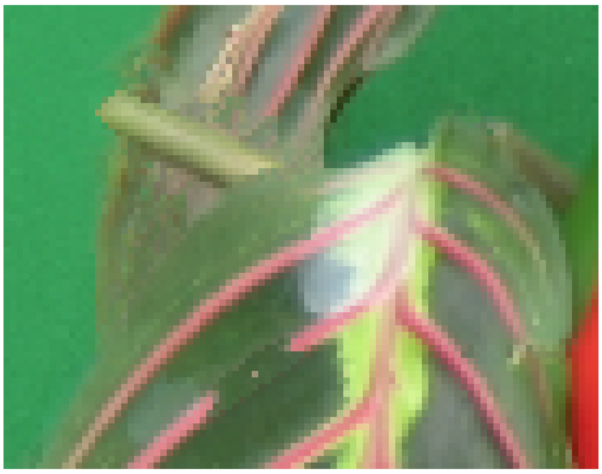
sion boundaries, and regions of low contrast, stochastic texture, both visible in figure 4.12. The former class of errors can be attributed to our assumption that noise is Gaussian, without outliers. The latter areas come down to the fact that these type of errors cannot be fixed at a coarse scale because the texture simply does not exist at a coarse scale—the blurring and downsampling turns these regions into textureless areas, making the choice of disparity arbitrary, and these errors are propagated up to the finer scales, where they are difficult to overcome.

The quantitative results (figure 4.14) show a decrease in r.m.s. error and gross errors of 15% and


(a) Multi-resolution $\mathbf{I}_0^*$


(b) Diff. of (a) from ground truth

(c) Single res.


(C) Zoom from (c)

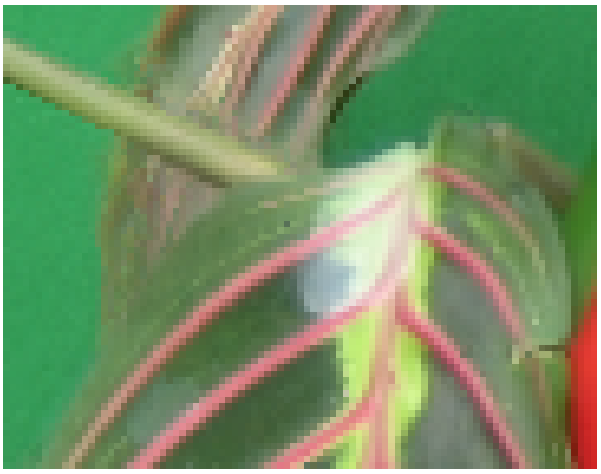

(A) Zoom from (a)

Figure 4.12: Plant sequence results. (a) Output image, $\mathbf{I}_0^*$, for the Plant sequence, generated using the EMS algorithm in multi-resolution framework. (b) The differences between (a) and the ground truth image. (c) Results equivalent to (a) & (b) for the single resolution framework. (C & A) Zooms of (c) and (a) respectively.

20% respectively, but an increase in energy of 5%, moving to a multi-resolution framework. However, it should be noted that the values of E_{texture} universally decrease; only those of E_{photo} increase. This suggests that a lower value of λ_{d}, weighting the prior more strongly against the data likelihood, might bring the minimal energy labelling more into line with the minimum under the two measures of visual fidelity.

As a side point, it is interesting to note that the exemplars indexed in $\mathbf{B}^s$ for a given output patch centred on $\mathbf{x}$ do not necessarily come from the location in an output image that corresponds to $\mathbf{x}$. For example, the output patches selected in figure 4.8 reconstruct the left eye of the feathered toy, but the closest exemplar found at the finest scale actually comes from the right eye, as shown in figure 4.10.

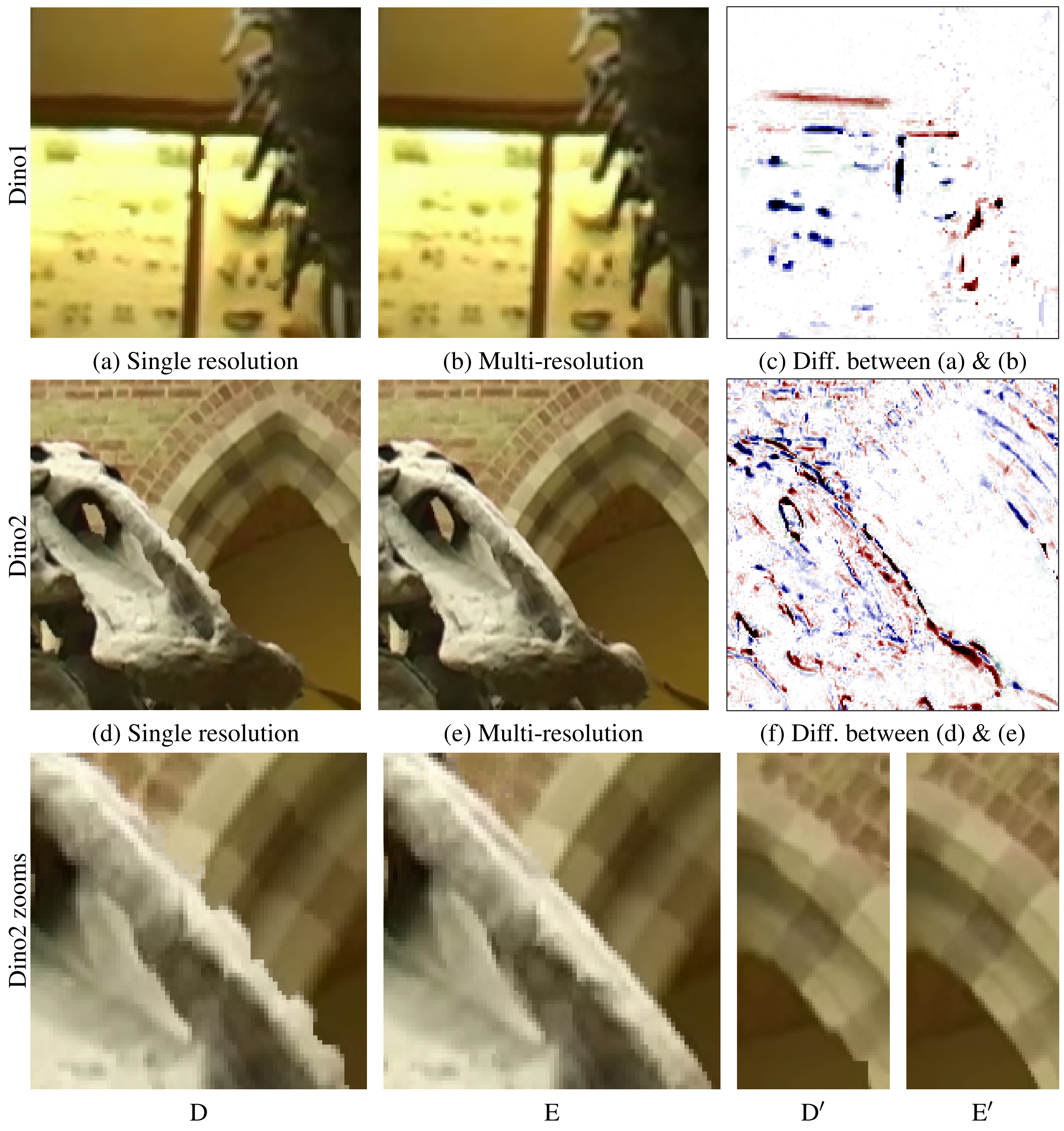
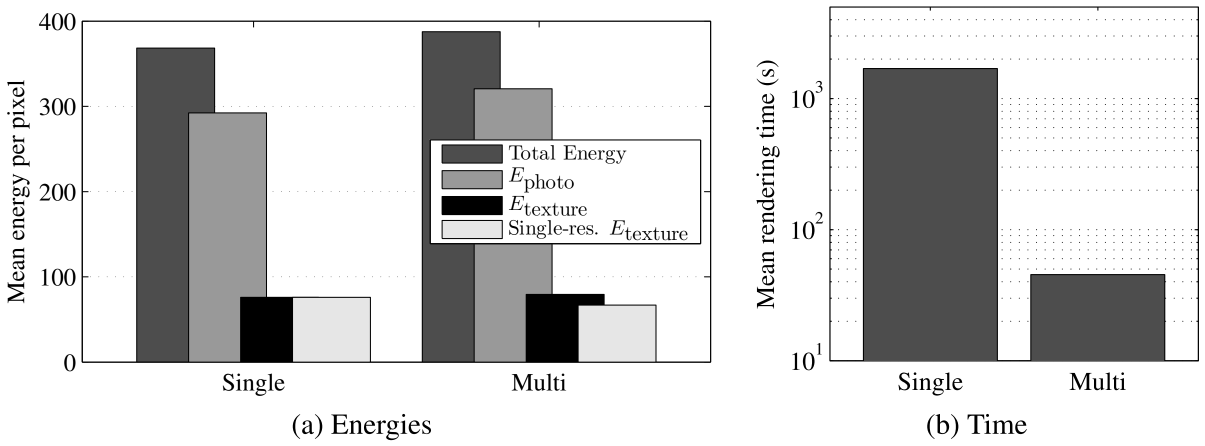


Figure 4.13: Edmontosaurus results. Output images for the Dino1 (*top row*) and Dino2 (*second row*) sequences, generated using the EMS algorithm in (a & d) a single resolution and (b & e) a multi-resolution framework, and (c & f) the differences between the two images. *Bottom row*: Zooms of the Dino2 results; (D & D′) are zooms of (d) while (E & E′) are zooms of (e).



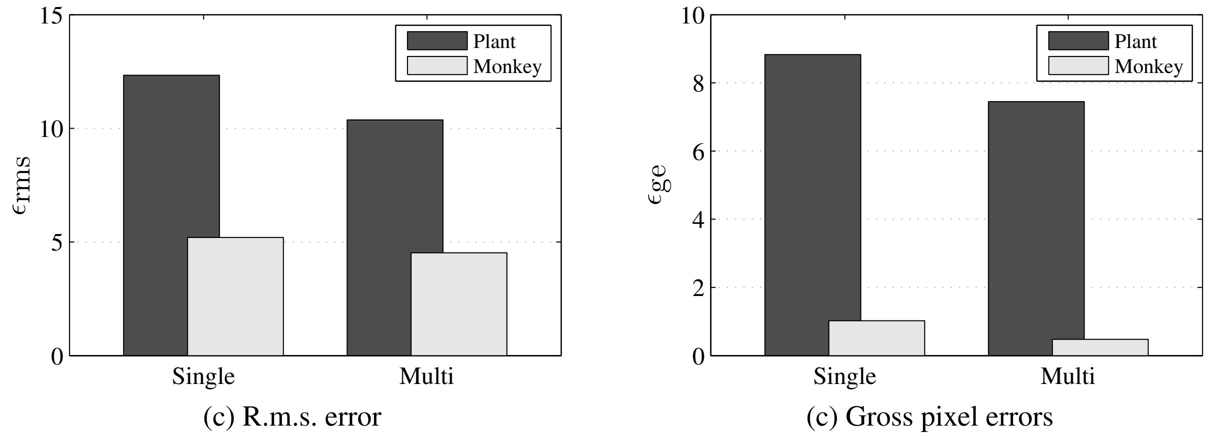


Figure 4.14: Single *vs.* multi-resolution. A quantitative comparison of single and multi-resolution frameworks, using EMS optimization. (a) Various energies of the output image (per image pixel). (b) Rendering time. (c) R.m.s. errors. (d) Gross errors. Results for (a) & (b) are averaged over the four test sequences, while results for (c) & (d) are given separately for the Plant and Monkey sequences.

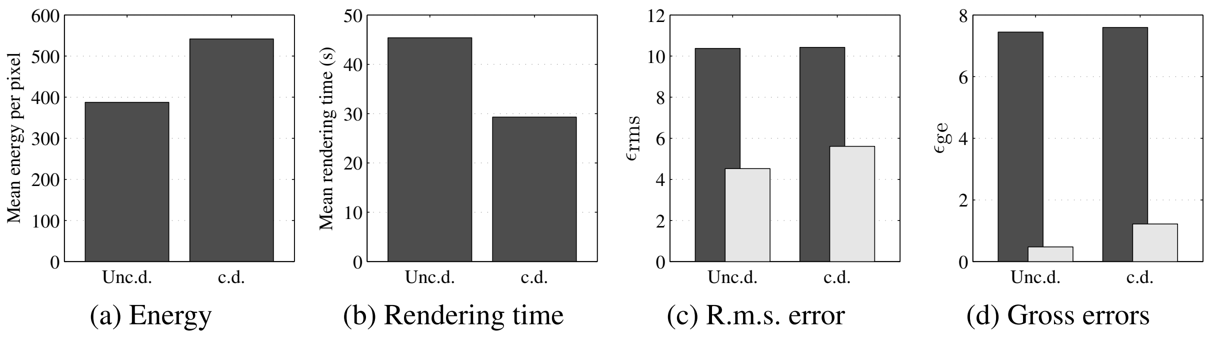


Figure 4.15: Quantitative effect of constrained disparity. Quantitative results for unconstrained (unc.d.) and constrained disparity (c.d.) for (a) the final energy per pixel, (b) rendering time, (c) r.m.s. error and (d) gross errors. Results for (a) & (b) are averaged over the four test sequences, while results for (c) & (d) are given separately for the Plant (dark grey) and Monkey (light grey) sequences.

4.4 Conclusion

This chapter has achieved two things: 1) comparing three optimization algorithms, applied to this problem for the first time; 2) introducing a multi-resolution framework in which a search-tree structure is traversed at a rate of one level per scale of the framework. In doing so, it has shown the following:

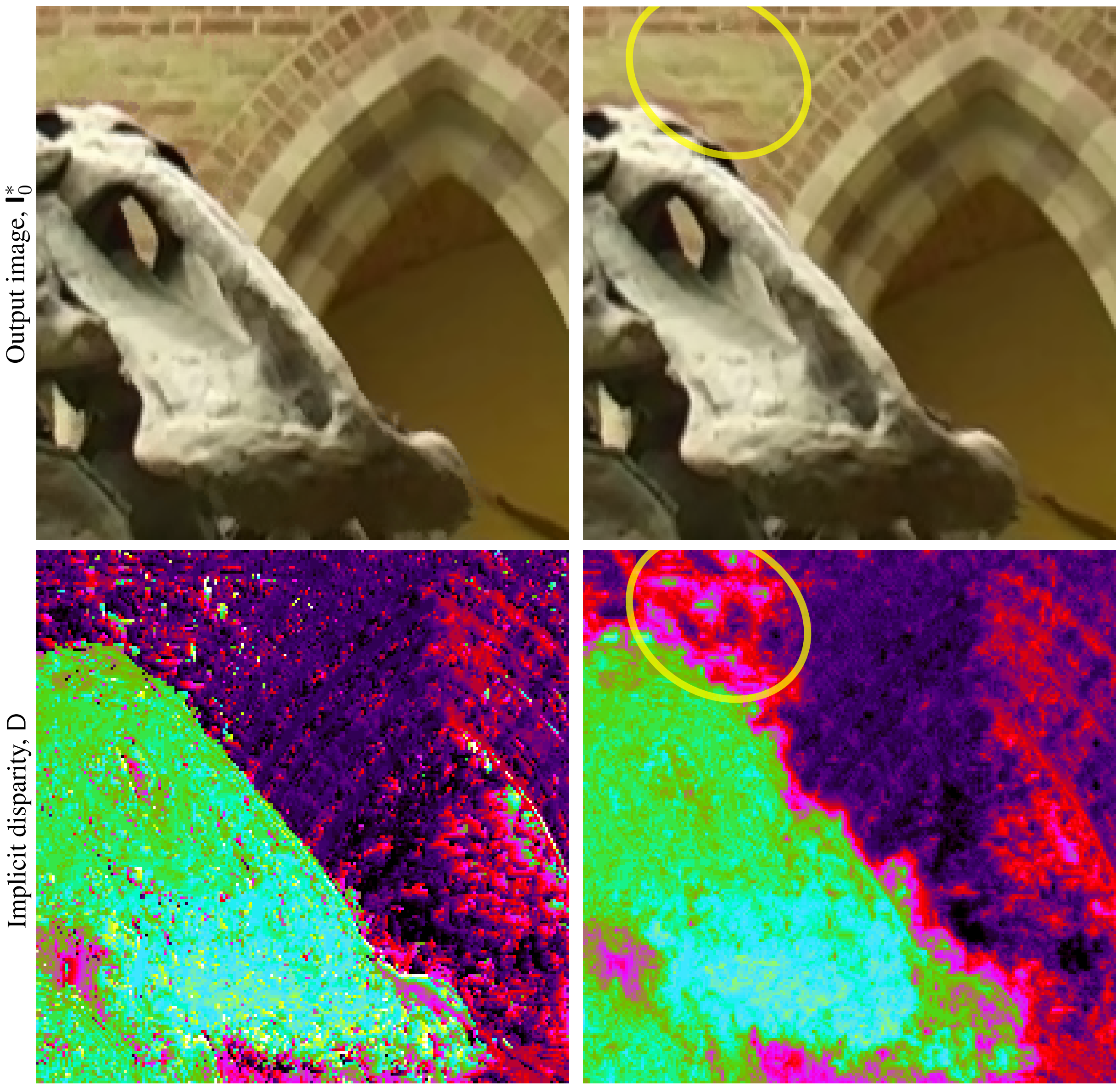
- While EMS minimizes the energy of this problem slightly less effectively than ICM, it is an order of magnitude faster.
- A multi-resolution strategy with the patch library search constrained independently for each output patch by results of the previous scale can increase the speed of rendering by an order of magnitude.
- The regularization of coarser scales in a multi-resolution framework can fix more large scale errors, improving output quality too.

Also shown are the main failure modes of the multi-resolution framework:

- Poor performance in reconstructing occluded background regions, due to an inability to reject occluded input samples.
- Regions of low contrast, stochastic texture are not reconstructed, due to the texture only existing at the finest scale, where it is too far from the initial estimate to be found.

These issues will be addressed in chapters 6 & 7.

The exemplar hierarchy presented here is very basic in form, and is by no means the most efficient look-up structure possible—however, I believe that the concept has been proven, and that future research may improve the approaches to produce even faster regularization.



(a) Unconstrained disparity (b) Constrained disparity

Figure 4.16: Qualitative effect of constrained disparity. Column (a) shows the reconstruction without constraining disparity, while column (b) shows the reconstruction with constrained disparity, with some of the resulting artifacts highlighted. *Top row*: output image. *Bottom row*: output disparity map.

Chapter 5

Field of Experts prior

In the previous chapter I investigated the performance of a non-parametric, exemplar-based image prior. In this chapter I investigate the performance of parametric image prior which also regularizes all $n \times n$ patches in the output image.

Roth & Black [RB05] developed a parametric image prior based on a product of Student's t-distributions of filter responses, which they called "Field of Experts" (FoE), and used this prior for inpainting and denoising images. Here I apply the prior to the NVS problem for the first time. In doing so I also use and improve an optimization technique called Iterated Local Search (ILS) [LMS02], which is relatively new to the field of Computer Vision, and compare its performance to two, more commonly used, large clique MRF optimizers: Iterated Conditional Modes (ICM) and Simulated Annealing (SA).

The chapter proceeds as follows. The first section introduces the parametric prior and defines the resulting objective function. The following section describes the various optimization strategies used to minimize the objective function. The third section presents results for both the relative performance of the optimizers and the images rendered using the FoE prior. The results with regard to the prior are discussed in the fourth section, before concluding in the final section.

5.1 Field of Experts

As discussed in §3.3.2.1, the FoE prior uses the same form of model as first used by Welling *et al.* (given in equation 3.9), but learned on and applied to all $n \times n$ patches in a larger image, rather than on/to a single $n \times n$ image, where $n \times n$ is the size of filter in each expert. The prior energy term can therefore

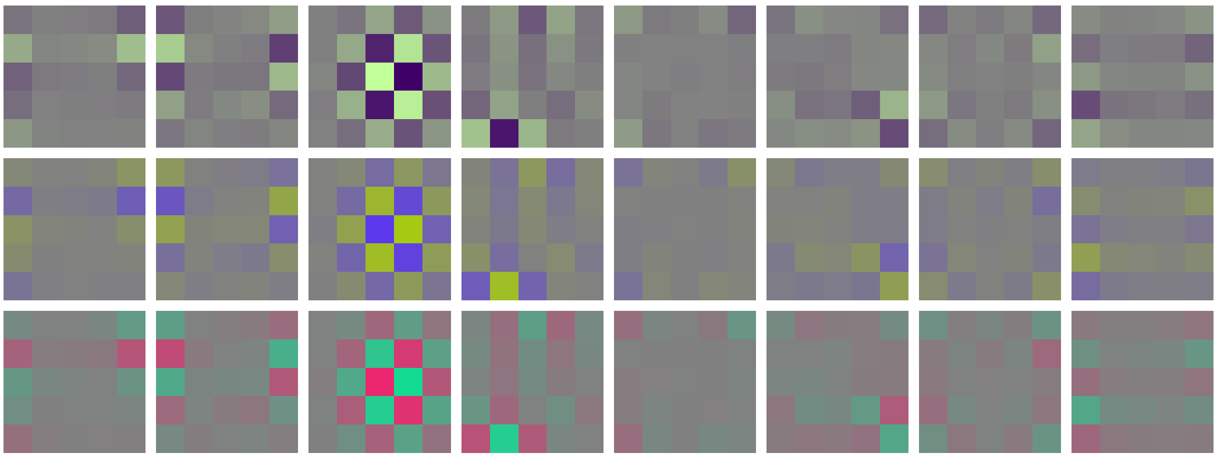


Figure 5.1: FoE filters. A selection of 24 of the 72 filters applied to RGB colour images. Each row shows the same eight greyscale filters applied to a different YCbCr colour channel, with the filters transformed into RGB space.

be written as

$$E_{\text{prior}}(\mathbf{I}_0^*, \mathsf{D}) = E_{\text{texture}}(\mathbf{I}_0^*) = \sum_{\mathcal{N} \in \mathbb{N}} \sum_{m=1}^{M} \alpha_m \log \left(1 + \frac{1}{2}\left(\mathbf{J}_m^\top \overrightarrow{I_0^*(\mathcal{N})}\right)^2\right), \tag{5.1}$$

where $\Theta = \{\mathbf{J}_m, \alpha_m\}_{m=1}^M$ are the prior model parameters, $\mathbf{J}_m$ being a vectorized filter, and α_m being the weight of the expert, which is always positive.

Roth & Black [RB05] use contrastive divergence [Hin02] to learn the model parameters from a diverse library of natural, greyscale images. As a result, it is a truly general image prior, unlike that of the previous chapter, which must be constructed from a set of images similar to the output image. A benefit of this is that the model parameters need only be learned once. Indeed, I use the parameters from [RB05]—24 5×5 experts—here.

As the parameters are learned for greyscale images, when generating colour images Roth & Black [RB05] apply the prior separately to each channel of the image in YCbCr colour space. This effectively generates a 72 expert model, but does not model any dependencies between the three channels, as can be seen from figure 5.1.

I use the same Gaussian noise model as in the previous chapter, therefore data costs are again given by equation 4.2. The total energy is given by equation 2.12.

5.2 Energy minimization

As with the non-parametric prior of the previous section, the prior used here generates an objective function containing large $n \times n$ cliques (5×5 cliques in the experiments here), which cannot be optimized by global optimization methods. Local methods must therefore be used.

Roth & Black [RB05] use gradient descent in colour space to minimize the objective energy in their denoising and inpainting problems. In these two problems the data costs are non-concave over colour, so there will not be many local minima created by the data costs, allowing the problem to converge on a good solution. In contrast, the data costs of the NVS problem *are* concave over colour, as Fitzgibbon *et al.* [FWZ05] show by finding multiple modes over colour in their NVS framework. Using gradient descent on such a problem would lead to the solution getting stuck in one of the many local minima caused by this concavity.

An alternative approach, which is used here, as in the previous chapter, is to discretize the state space, then use a coordinate descent method that globally minimizes the conditional energy of each pixel. As gradient descent merely finds a *local* minimum of the same conditional energies, it generates more local minima in the overall problem than the coordinate descent approach. The latter approach is therefore more likely to find a better solution.

The state space discretization used is identical to that of the previous chapter—the problem is parameterized as one over disparity, and the disparity is discretized as described in §4.2.1.3, transforming the objective energy as per equation 4.5. The colour and data cost of each label are again cached into the data structures $\mathbf{C}$ and $\mathbf{E}_\mathrm{p}$ using equations 4.6 and 4.7 respectively.

5.2.1 Optimization methods

Various coordinate descent optimization methods will be used to minimize the objective energy, as described below. All these methods compute the conditional energies of pixels, written as $E(\{L(\mathcal{X}\backslash\mathbf{x}), l\}| \mathbf{I}_1, .., \mathbf{I}_N)$ for pixel $\mathbf{x}$, but, as with the exemplar-based prior, the energy need only be computed for those n^2 cliques which contain pixel $\mathbf{x}$.

In the previous chapter I used ICM and EM-style ICM for optimization, as well as an approximation to ICM (AICM) which reduced its high computational burden. EM cannot be used with the FoE prior because there are no latent variables in the objective function given which the output variables are independent of each other. However, the time to compute the parametric prior's energy is much faster, making optimization using ICM practical, and removing the need for the faster but less effective AICM.

A problem with the local optimizers seen in the previous chapter is their tendency to get stuck in local minima. There this was overcome with a multi-resolution framework. Here the focus is on improving the optimization at a single resolution. As a result, more advanced local optimizers than ICM are used for comparison: SA and ILS.

The optimizers are described in more detail below.

5.2.1.1 Iterated Conditional Modes

The label of each output pixel is updated sequentially by minimizing its conditional energy, as per equation 4.10, and this is repeated until convergence. The order pixels are updated in can affect the solution converged to. Here I try two different orders.

Raster ICM: Pixels are visited in raster scan order, down each column in order, from left to right. The ordering reduces the number of cache misses in the computation, hence keeping computation time low.

Greedy ICM: This novel algorithm visits pixels in a greedy order, updating the pixel that reduces the energy most at every step. This requires that the current conditional energy distributions for all pixels are kept cached, *i.e.* all cached at the start, then distributions for neighbours[1] of each updated pixel recomputed, and also means that pixels are visited in an unordered fashion, both of which increase computation time.

In both cases the labelling is initialized to the ML solution, *i.e.* that which minimizes E_{photo}.

5.2.1.2 Simulated Annealing

SA [KGV83, GG84] chooses a labelling for each pixel by sampling from its conditional probability distribution, which can be derived from the conditional energy. In addition, the conditional probability distribution is made dependent on a global control parameter, T, known as the "temperature", as follows

$$p(L(\mathbf{x}) = l|L(\mathcal{X}\backslash\mathbf{x})) = \frac{\exp\left(-E(\{L(\mathcal{X}\backslash\mathbf{x}), l\}|\mathbf{I}_1, .., \mathbf{I}_N))/T\right)}{\sum_{i=1}^{N_{\text{L}}} \exp\left(-E(\{L(\mathcal{X}\backslash\mathbf{x}), i\}|\mathbf{I}_1, .., \mathbf{I}_N)/T\right)} \tag{5.2}$$

The algorithm works by sampling all pixel labels from the above distribution, either synchronously or sequentially, then lowering the temperature according to a cooling schedule and repeating the process.

[1]Neighbours in this context are pixels which share at least one clique with the pixel in question.

At $T = \infty$ the distribution is uniform, and the algorithm is essentially a random search through all labellings that will eventually find the global minimum. As $T \to 0$ the mass of the distribution becomes increasingly concentrated on the minimum energy label. It can therefore be seen that ICM is just a special case of SA where the MRF is essentially frozen, yielding a local minimum close to the initial labelling.

The aim of this sampling with slow cooling (annealing) is to strike a balance between these two extremes of computational expense and local optimization. The stochastic nature of the sampling process allows the labelling, L, to hop out of local minima, while the annealing causes the energy gradient to remain effective in gradually reducing the labelling energy towards what is hoped to be the global minimum.

In the implementation used here pixels are updated sequentially, in raster scan order, and the cooling schedule is given by

$$T_{\text{n}}^{t} = \left(\exp\left(\frac{t_{\text{max}} - t}{0.3 t_{\text{max}}}\right) - 1\right) \cdot \frac{80}{\exp(1/0.3)}. \tag{5.3}$$

where T_{n}^{t} is the temperature at time (or iteration) t, on a normalized scale described in appendix B, and t_{max} is the number of iterations used. In addition, samples are drawn by computing each conditional distribution in its entirety, then sampling randomly from this, rather than using the traditional Metropolis algorithm[2] [MRR+53] approach; the reason for this is discussed in appendix B.

After t_{max} iterations, Raster ICM is applied to the solution to ensure that the final solution is at a local minimum. The labelling is initialized to a random solution drawn from a uniform distribution over all labellings.

5.2.2 Iterated local search

ILS [LMS02] is a relatively new optimizer to the field of Computer Vision, introduced in 2006 by Cordón & Damas [CD06] for image registration, and Woodford *et al.* [WRTF06] for NVS, though a similar idea based around a genetic algorithm was presented for image restoration [HD91] in 1991.

At its core, ILS makes use of any local optimizer which finds a local minimum labelling, L^*, deterministically, given an initial labelling, L. For example, in this work I will be using Raster ICM. Such a function reduces many labellings to the same local minimum labelling, therefore the space of local minimum labellings, $\mathbb{L}^*$, is much smaller than the space of all labellings, $\mathbb{L}$, as well as having a considerably

[2]The Metropolis algorithm is an example of Monte Carlo (MC) sampling in which a value, l, is proposed for a given label, $\mathsf{L}(\mathbf{x})$, and accepted with probability $\min(\exp((e - e')/T), 1)$, where $e = E(\mathsf{L}|\mathbf{I}_1, .., \mathbf{I}_N)$ and $e' = E(\{L(\mathcal{X}\backslash\mathbf{x}), l\}|\mathbf{I}_1, .., \mathbf{I}_N)$.

lower average cost per labelling. The approach of ILS is to search the former space, in contrast to SA, which traverses the larger latter space.

The aim of ILS is to search the space of $\mathbb{L}^*$ locally, in a pseudo-gradient-descent manner. Since the gradient of this space cannot be computed easily, random steps are made and accepted only if they turn out to be downhill steps. The steps are made by stochastically perturbing, or "kicking", the current solution. The full algorithm is described in pseudo-code below.

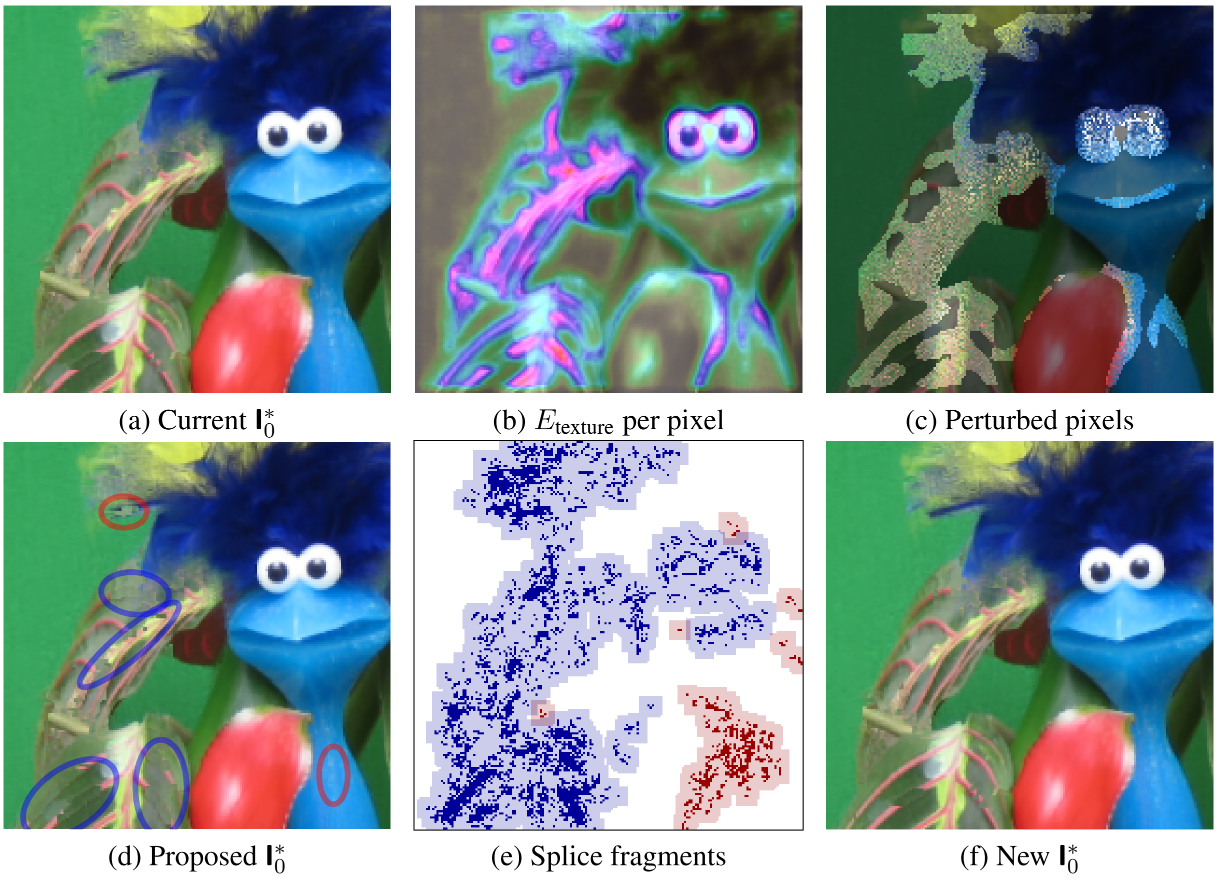
```
ITERATEDLOCALSEARCH
1 $\mathsf{L}_0 \leftarrow$ GENERATEINITALSOLUTION
2 $\mathsf{L}_0^* \leftarrow$ LOCALSEARCH($\mathsf{L}_0$)
3 repeat
4         $\mathsf{L}_1 \leftarrow$ KICK($\mathsf{L}_0^*$)                    $\triangleright$ Perturb the current labelling
5         $\mathsf{L}_1^* \leftarrow$ LOCALSEARCH($\mathsf{L}_1$)             $\triangleright$ Locally minimize the perturbed labelling
6         $\mathsf{L}_0^* \leftarrow$ LOWESTENERGY($\mathsf{L}_0^*, \mathsf{L}_1^*$)   $\triangleright$ Select the lowest energy labelling
7 until termination condition met
8 return $\mathsf{L}_0^*$
```

For the purposes of this work the termination condition is met when the total energy decrease over the last five iterations is less than 0.001%. As with ICM, the labelling is initialized to the ML solution.

5.2.2.1 Kicks

The kick plays a pivotal role in the efficiency of ILS. A good kick heuristic should perturb the labelling far enough to be in the convergence basin of a new local minimum, but not so far that it is effectively randomly restarting the algorithm, and in a direction that is likely to lead to a lower energy minimum. In this work I compare and combine two different kicks, as follows.

Blowtorching: This kick perturbs labels by sampling them simultaneously from the same distribution as used by SA, defined in equation 5.2. However, a high temperature, $T = 100\log(\sum_{\mathbf{x}}\sum_l \exp(-E_\mathrm{p}(\mathbf{x},l))/(N_\mathrm{L}\cdot|\mathcal{X}|))$, is used, for one iteration only, and, in order to reduce computation, only pixel labels deemed to be incorrect are perturbed. Candidate pixels for this process are chosen using the FoE energy of equation 5.1, which provides an energy for each patch in an image—the 25% of pixels whose average clique energies are greatest are selected, as demonstrated in figure 5.2(b & c), as these are the least natural pixels (based on prior knowledge) in the image.



(a) Current $\mathbf{I}_0^*$ (b) E_{texture} per pixel (c) Perturbed pixels (d) Proposed $\mathbf{I}_0^*$ (e) Splice fragments (f) New $\mathbf{I}_0^*$

Figure 5.2: ILS with a blowtorch kick. This figure demonstrates one iteration of ILS, in particular with a Blowtorch kick. (a) The current lowest energy output image, $\mathbf{I}_0^*$, a local minimum. (b) The current image overlaid with the FoE energy, E_{texture}, contributed by each pixel. The FoE energy associated with each clique is averaged over all the pixels in the clique. Greyscale indicates a lower energy; red indicates a higher one. (c) The 25% of pixels with the highest FoE energy are perturbed by simultaneously sampling from their conditional distributions, raised to a high temperature (*i.e.* almost uniform). The other 75% of pixels are darkened for clarity. (d) The perturbed labelling is then optimized using Raster ICM to find a new local minimum, which is the proposed $\mathbf{I}_0^*$ (with differences from (a) highlighted; ellipse colour is purely for clarity). (e) Pixels with different labels between the current and proposed labellings are found (dark pixels; light pixels show those pixels which share a clique with at least one dark pixel) and segmented into fragments (not shown). Any fragments which produce a lower energy with the proposed labelling (red pixels) are spliced into the current labelling, generating a new lowest energy $\mathbf{I}_0^*$ (f).

Disparity smoothing: This kick leverages the fact that most artifacts in the ML images occur when smooth surfaces are given incorrect disparity labels. Kicking the labelling to a smoother disparity solution will therefore stand a good chance of generating a lower energy local minimum. The smoothing is achieved using a square, 2-d median filter on the labelling L_0^*, with a filter side length which alternates between 3 and 5 pixels long each iteration that this kick is used.

Combined: Combined ILS uses a Disparity smoothing kick for one iteration followed by a Blowtorching kick for one iteration, and so on.

5.2.2.2 Optimal splice

A consequence of both the kicks described above is that the differences between L_0^* and L_1^* (from line 6 of the ILS pseudo-code) tend to be fragmented, surrounded by areas of identical labelling. The lower energy labelling of L_0^* and L_1^* is then used as the current solution, but it may be that for some fragments the labelling L_0^* might be better, while for others L_1^* might be better, so the piecemeal approach to updating misses an opportunity for further optimization—the good parts of the two solutions could be combined.

Two multi-label optimization approaches that combine pairs of solutions are genetic algorithms (GA), *e.g.* [HD91], which randomly swap regions of labels between two labellings, and graph-cuts-based fusion move approaches, *e.g.* [BVZ01], which select a label for each pixel from one of the two input labellings such that the energy of the output labelling is minimized. The latter approach cannot operate on the large cliques of the problem presented here, while the former approach does not guarantee to lower the energy, so is not suitable for a gradient descent approach such as ILS. However, it is possible to splice together regions of the two input labellings, similar to GA, while ensuring that the energy does not increase, thus creating a fusion move approach, in a method I introduce here called "optimal splice".

Continuing with the genetic metaphor, the genes to be spliced are the distinct fragments of differences between L_0^* and L_1^*. A fragment is defined to be the largest group of pixels, $\mathcal{P}$, grown from a seed pixel, for which $L_1^*(\mathbf{p}) \neq L_1^*(\mathbf{p}), \mathbf{p} \in \mathcal{P}$ and all the pixels share a clique with at least one other pixel in $\mathcal{P}$, such that each of the fragments can be swapped between L_0^* and L_1^* independently. For fixed size neighbourhoods, as in this case, fragments can be segmented quickly using morphological operations on the binary image, $\mathsf{L}_0^* \neq \mathsf{L}_1^*$.

The optimal splice is achieved by selecting for the output labelling each gene or fragment from whichever of L_0^* and L_1^* generates the lower energy. The energy of a particular fragment, $\mathcal{P}$, in a given labelling, L, is defined by

$$E(L(\mathcal{P})) = \sum_{\mathbf{p}\in\mathcal{P}} E_{\mathrm{p}}(\mathbf{p}, L(\mathbf{p})) + \sum_{\mathcal{N}\in\mathbb{N}_{\mathcal{P}}} \sum_{m=1}^{M} \alpha_m \log\left(1 + \frac{1}{2}\left(\mathbf{J}_m^\top \overrightarrow{C(\mathcal{N}, L(\mathcal{N}))}\right)^2\right), \qquad (5.4)$$

where $\mathbb{N}_{\mathcal{P}}$ is the set of neighbourhoods which contain at least one pixel in the set $\mathcal{P}$.

The function LOWESTENERGY in line 6 of the ILS pseudo-code is thus replaced with the new function, OPTIMALSPLICE, described in pseudo-code below.

```
OPTIMALSPLICE($\mathsf{L}_0^*, \mathsf{L}_1^*$)
1 $\mathsf{B} \leftarrow \mathsf{L}_0^* \neq \mathsf{L}_1^*$
2 $\mathbb{P} \leftarrow$ COMPUTEFRAGMENTS($\mathsf{B}$)
3 for each $\mathcal{P} \in \mathbb{P}$
4     if $E(L_0^*(\mathcal{P})) > E(L_1^*(\mathcal{P}))$
5         $L_0^*(\mathcal{P}) \leftarrow L_1^*(\mathcal{P})$
6 return $\mathsf{L}_0^*$
```

It is important to note that if the two input labellings to OPTIMALSPLICE are local minima found using ICM, no further local optimization is required on the output labelling, as it will also be a local minimum. This results from the independence of the fragments—no changed fragment changes the neighbourhood, and therefore the 'local minimum' status, of another fragment, and since all fragments start as local minima in both labellings, they must also therefore end that way, regardless of how they are swapped between labellings[3]

One iteration of the ILS algorithm is illustrated in figure 5.2, using the Blowtorch kick as an example.

5.3 Experiments and Results

This section presents results comparing the performance of the various optimizers described, and the quality of images generated using the parametric Field of Experts prior. The value of λ_d, the noise parameter, used in these experiments was $\lambda_\text{d} = 1/32N$, selected using a visual evaluation of results using Raster ICM and a range of values for λ_d, some of which are shown in figure 5.4.

5.3.1 Optimizer performance

All the optimizers described were run on the four test problems. SA was run for a range of values of t_max. The results for final energy and time taken to render the image, averaged over the four sequences, are shown in figure 5.3.

The first thing to notice is that the number of iterations, t_max, has a huge impact on the performance of SA—$t_\text{max} = 20$ gives the highest energy of all methods and $t_\text{max} = 500$ gives the lowest, with energy

[3]Any splicing of fragments is therefore a local minimum. Optimal splice simply finds the lowest energy one possible.

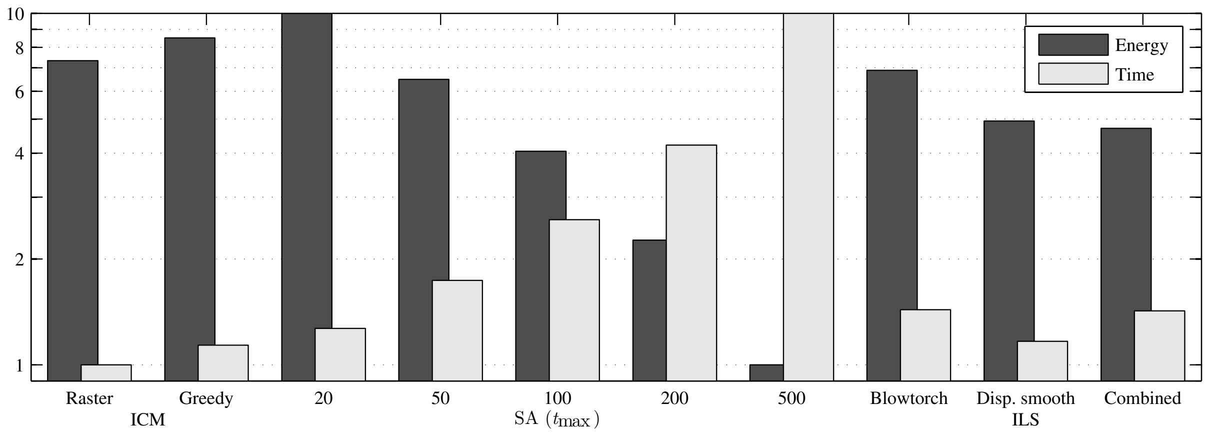
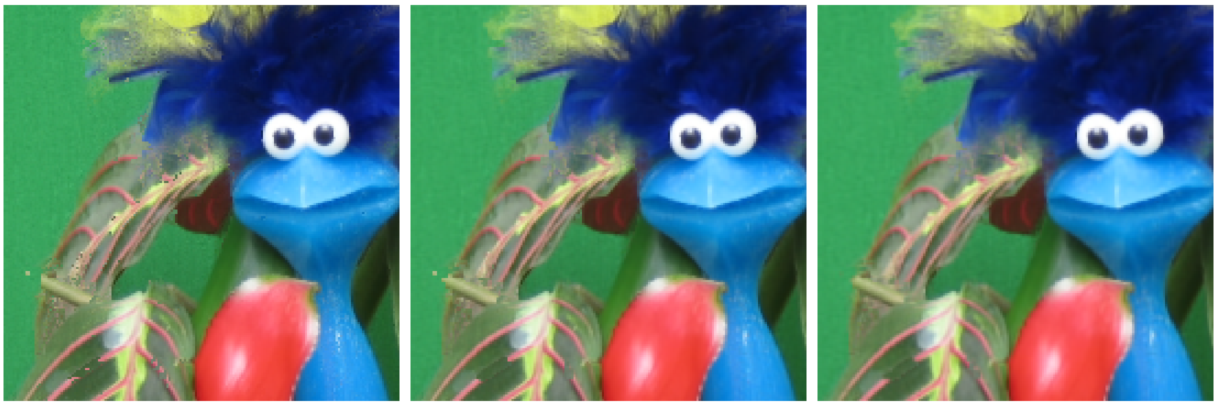


Figure 5.3: Optimization results This bar graph shows the final energy achieved by each of the described optimizers, with SA being run for a range of $t_{\max}$, and also the time taken by each optimizer. Results have been averaged over the four test problems. Both energy and time have been linearly normalized so that the lowest and highest values in each case are transformed to 1 and 10 respectively, and values are plotted on a log scale.

gradually decreasing in between, while time gradually increases over the same range, with $t_{\max} = 500$ taking longest by some distance. Figure 5.5(b) shows the decrease in final energy with $t_{\max}$ across the four test sequences. The rate of decrease in energy decreases as $t_{\max}$ increases, indicating that there is a diminishing rate of return with each extra iteration. Figure 5.5(a) validates the method given in appendix B for setting temperature according to a normalized schedule, by showing that the desired and achieved normalized temperatures closely match, after a period of 'burn in' during which the temperatures are initially aligned. It also shows how the energy decreases over time, with the occasional increase in energy hinting at the algorithm's stochastic nature.

Returning to figure 5.3, it can be seen that Greedy ICM is both slower than Raster ICM (in fact, it is almost six times slower), and also generates a higher energy on average; it does find a lower energy for one of the four sequences. The message seems to be that the greedy approach leads to a different local minimum, but one that is no more likely to be lower than using Raster ICM. Given that the latter approach is much faster, it is the obvious choice for a local search algorithm in this application.

For the ILS approaches, the Disparity smoothing method is faster than the Blowtorch method, and generates a lower energy. The Combined approach is comparable in speed to the Blowtorch method, but generates the lowest energy of all three methods. Figure 5.6 gives ILS results for each of the test sequences individually, and shows that the Disparity smoothing method converges in the fewest number of iterations in all cases, indicating why it is faster. Whilst this approach generates a lower energy than



(a) $\lambda_d = 1/N$ (a) $\lambda_d = 1/32N$ (a) $\lambda_d = 1/1024N$

Figure 5.4: Varying λ_d . The effect of different values of λ_d, effectively varying the strength of the FoE prior, on the Plant sequence, using Raster ICM for optimization.

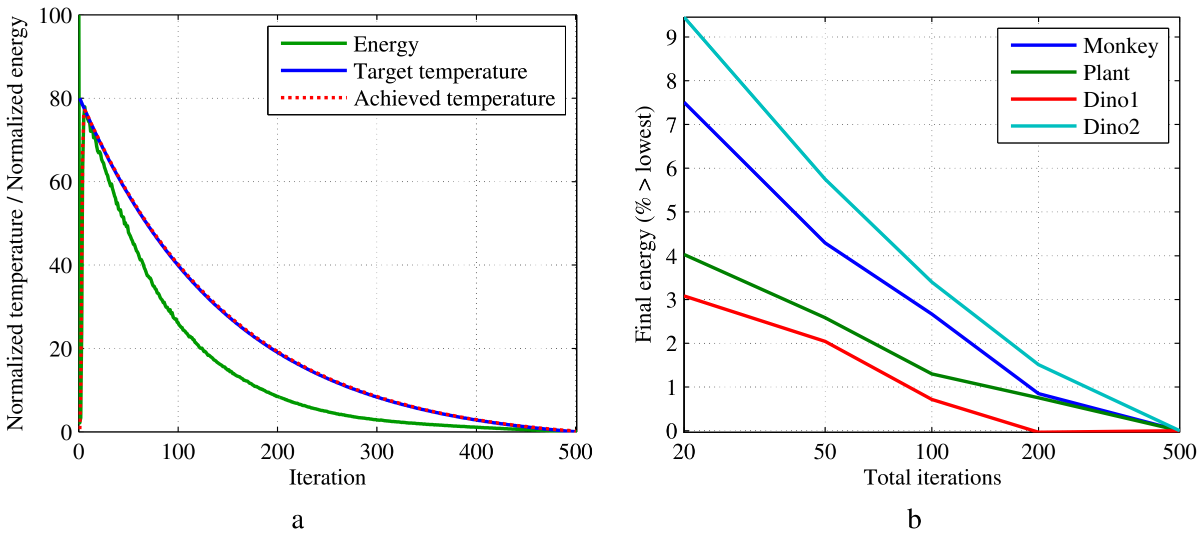
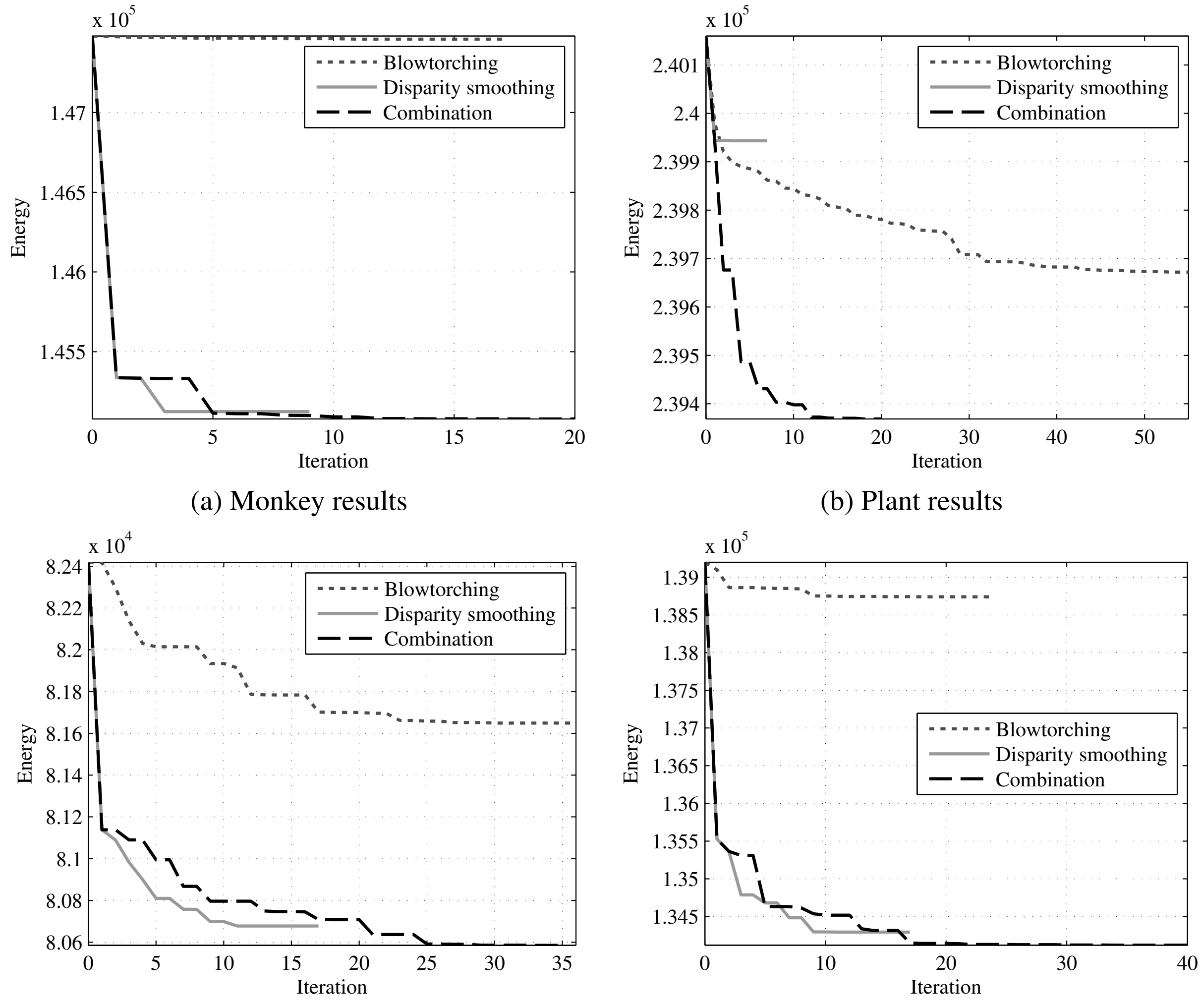


Figure 5.5: SA quantitative results. (a) A graph showing desired and achieved temperatures, on the normalized scale described in appendix B, at each iteration of SA carried out on the Plant sequence, with $t_{\text{max}} = 500$. Also shown is the energy at the end of each iteration, normalized to the range $[0, 100]$. (b) A graph showing the final energy achieved by SA against the number of iterations used, t_{max}, for each of the four test sequences. The energy is shown as a percentage increase over the energy achieved with $t_{\text{max}} = 500$.

Blowtorching on all but the Plant sequence, the Combined approach generates the lowest energy on all sequences, combining, as it does, the benefits of both approaches, and it also converges faster than the Blowtorch method on half the sequences.



(a) Monkey results (b) Plant results

(c) Dino1 results (d) Dino2 results

Figure 5.6: ILS quantitative results. Graphs of energy over time for each of the three ILS kick methods on each of the four test sequences (a–d).

5.3.2 Prior performance

The performance of the prior is judged by generating output images for the four test sequences using three of the optimizers: Raster ICM, Combined ILS and SA with $t_{\text{max}} = 500$. Quantitative results are given in table 5.1. These indicate that the image quality improves when using Combined ILS over Raster ICM, but then decreases by a greater amount when switching to SA (even though the latter produces a lower energy solution).

The output images themselves tell a similar story. Figure 5.7 shows results for the Dino1 and Dino2 sequences. Of the Dino1 results, Combined ILS is the only one to fix the cabinet frame. On Dino2, Raster ICM leaves many artifacts of a few pixels across (seen in the zoom), which Combined ILS is

	Monkey		Plant	
	ϵ_{rms}	ϵ_{ge}	ϵ_{rms}	ϵ_{ge}
Raster ICM	5.26	0.93	12.65	9.21
Combined ILS	4.92	0.76	12.39	8.93
SA ($t_{\max} = 500$)	7.11	2.53	15.01	11.36

Table 5.1: Rendering performance. A table of quantitative results for the quality of images rendered for Monkey and Plant sequences.

able to correct, as shown in the zoom—the brickwork and outer arch are reconstructed much better, but there are also larger, unnatural looking, horizontal and vertical edges. These unnatural edges occur more frequently in the SA output, giving many diagonal discontinuity boundaries (where data cost distributions tend to be multi-modal) a sawtooth appearance, and texture such as brickwork is also smoothed out. While the result using Combined ILS for Dino1 looks good, the jagged edges on the bridge of the snout and edge of the arch in Dino2 make that result look very unnatural.

For the Monkey sequence, shown in figure 5.8, Combined ILS again fixes a few small artifacts in the image generated by Raster ICM, producing what is a pleasing result, save for the occlusion artifacts under the monkey's arm. SA finds a solution that smoothes over the texture of the monkey's fur, as well as background image under the monkey's arm, but does so in a way that is difficult to spot at a glance. It generates more visible artifacts in the black stitching on the monkey's face. None of the approaches reproduce the low contrast texture on the monkey's face.

On the Plant sequence, shown in figure 5.9, the image rendered using Combined ILS corrects some artifacts of the Raster ICM output (as shown in the zoom), but some of the leaf 'ribs' still contain gaps, especially in the partially occluded region under the feathers, and the prior fails to reconstruct the stalk to the left, and the low contrast texture of the baize background. The results using SA remove much of the ribs of the leaves, making them more homogenous and highlighting the FoE prior's preference for textureless images. The zooms in figure 5.9 demonstrate the effectiveness of the Disparity smoothing kick. Small artifacts caused by disparity errors in otherwise smooth areas are corrected by this kick, removing the artifact and generating a lower prior energy. The approach encourages smooth surfaces, thus implicitly encoding a geometry prior of sorts. In contrast, the result of SA generates surfaces that are less smooth than those of Raster ICM.

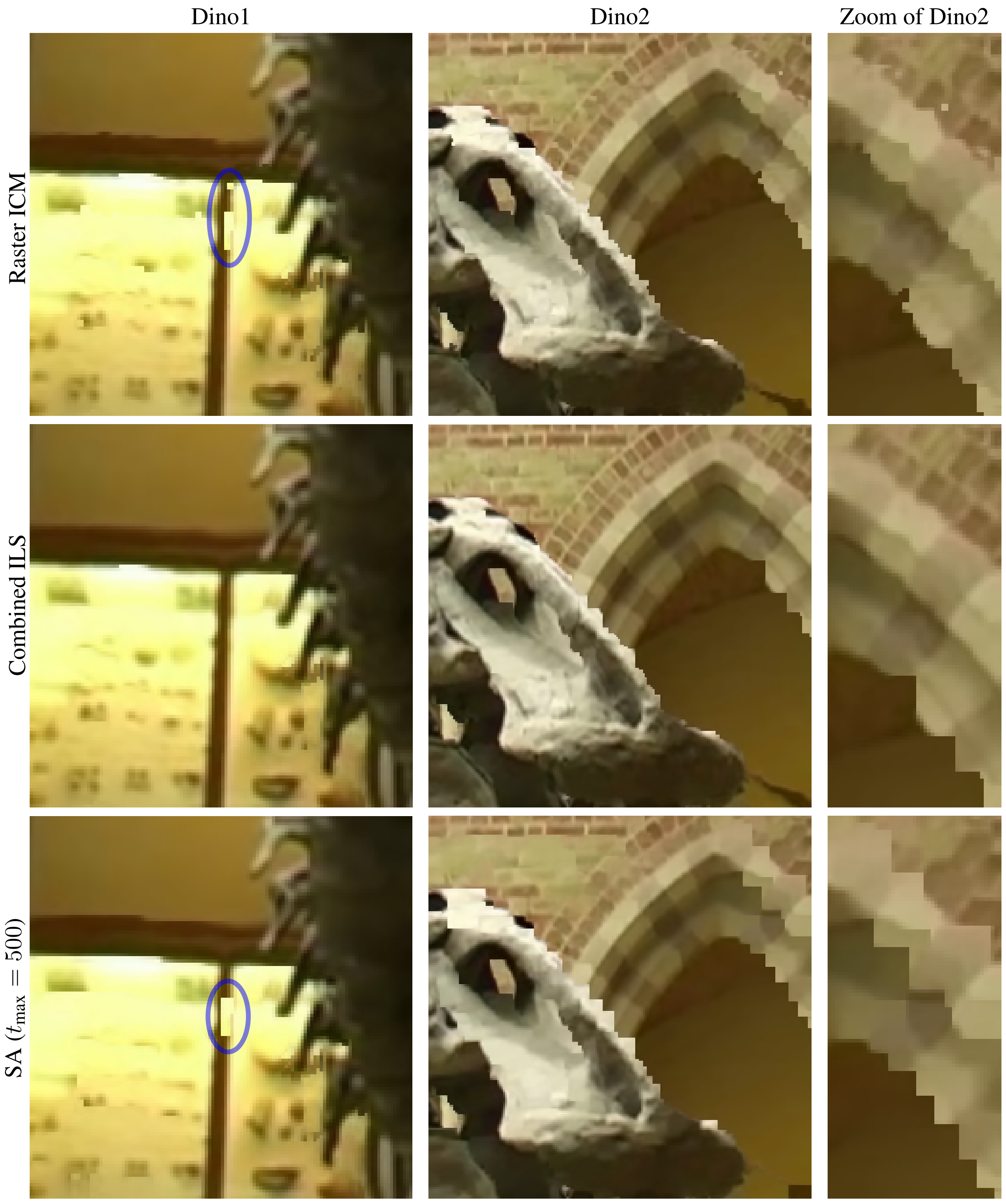


Figure 5.7: Edmontosaurus results. Output images for the Dino1 and Dino2 sequences, with blue ellipses in the Dino1 results and a zoom of the Dino2 results to highlight errors.

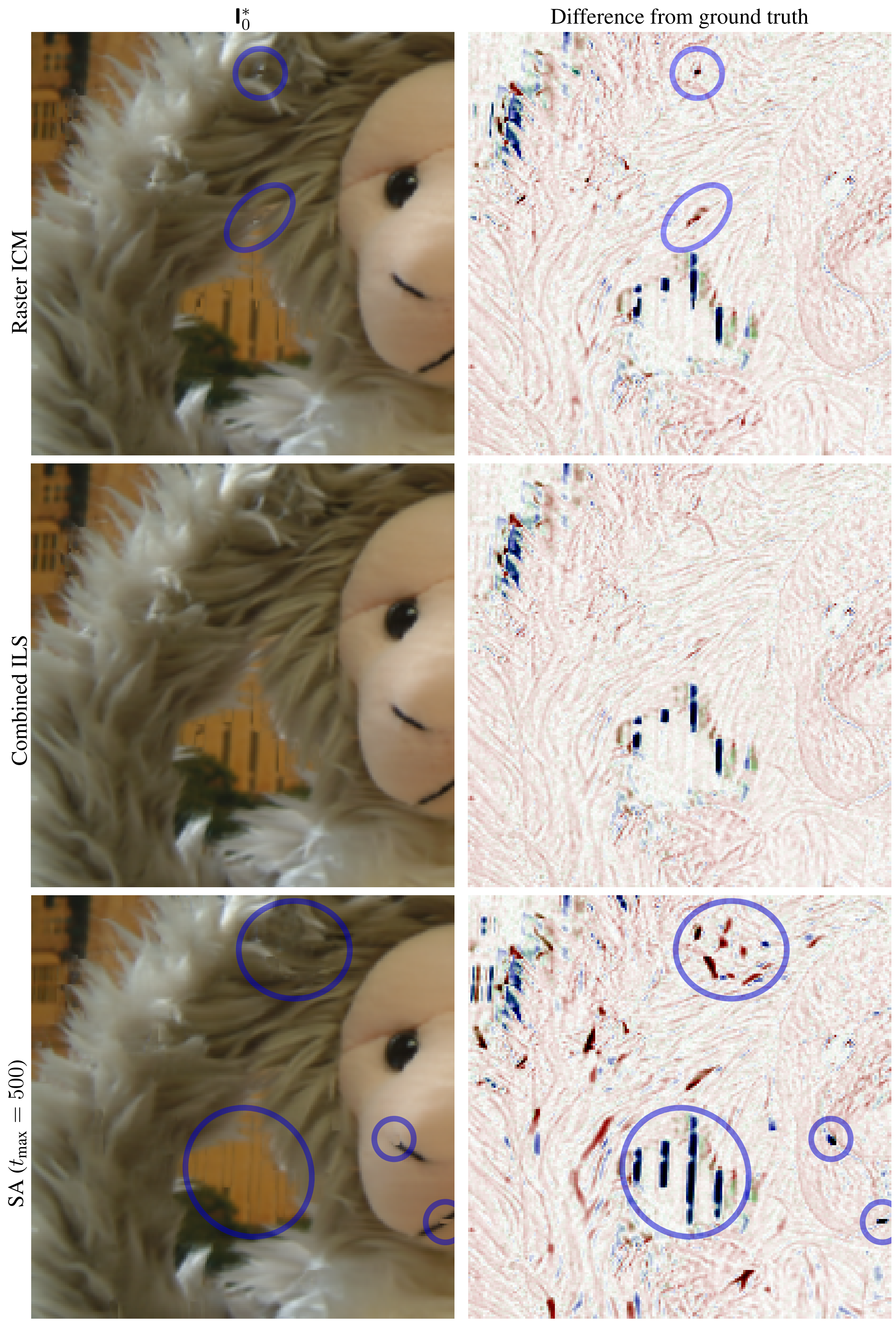


Figure 5.8: Monkey results. Output images and differences from ground truth for the Monkey sequence. Some artifacts are highlighted symmetrically across output and difference images for clarity.

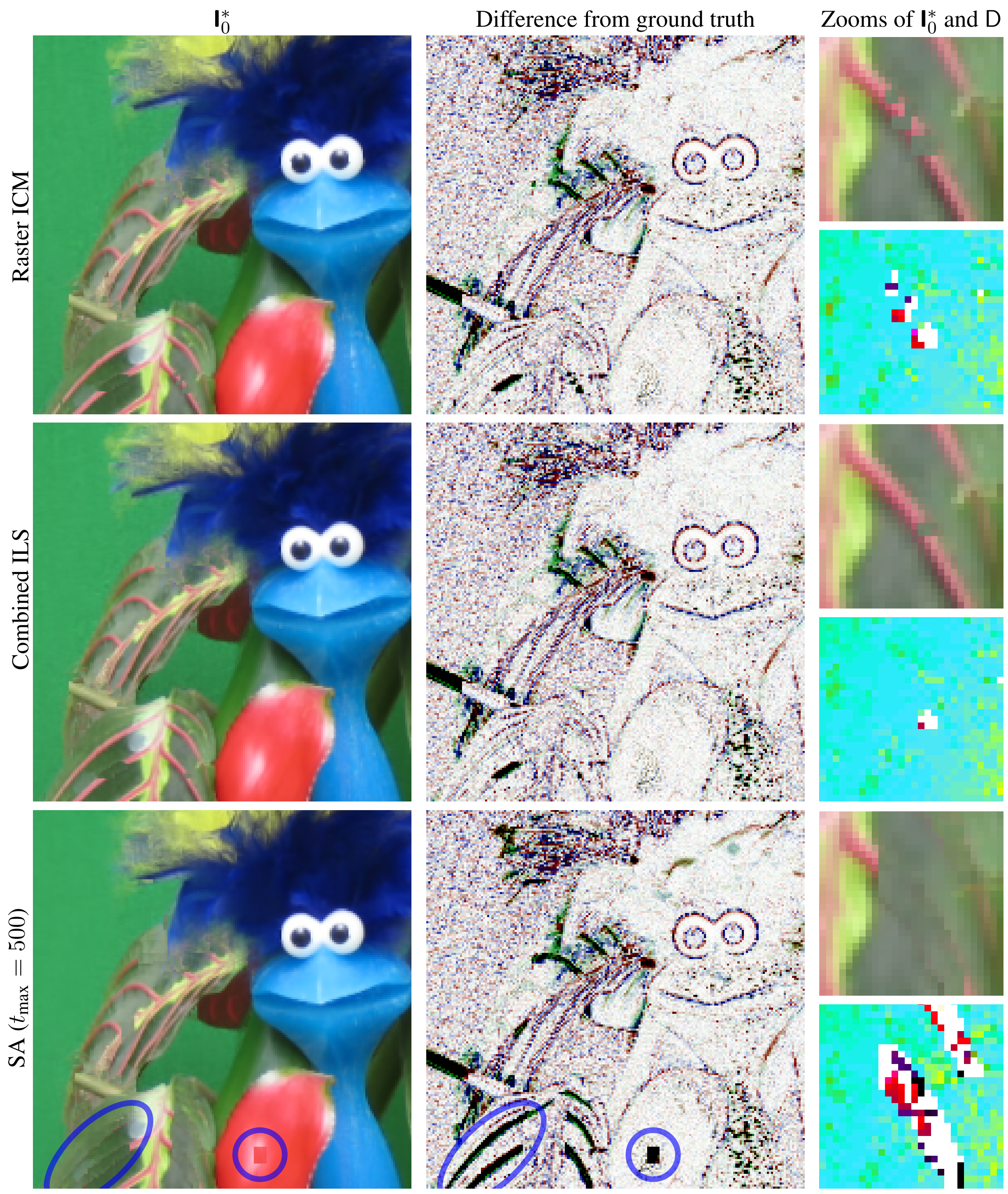


Figure 5.9: Plant results. Output images and differences from ground truth for the Plant sequence, with a zoom of the output image, and the corresponding section of the disparity labelling. Some artifacts are highlighted symmetrically across output and difference images for clarity.

5.4 Discussion

There are three main points of interest which arise from the results. Firstly, that minimizing energy ‘too much’ can actually reduce the quality of the images. This is, of course, very dependent on the value of λ_{d}, which, given that it was set based on results produced using Raster ICM (a very local optimizer), was certainly set too high in light of the results. However, it is also, I believe, a function of the form of the prior, as discussed below.

The second point is that the prior prefers textureless, or homogenous[4], patches. This is highlighted in figure 5.2(b)—the lower left leaf has a section of ‘rib’ missing, which the prior marks as likely because it blends in with the rest of the leaf, while the correct part of the ribs (which are textured) are marked as unlikely (high cost).

In fact, the FoE prior, as learned by Roth & Black [RB05], is a unimodal distribution with all homogenous patches (and no others) at the mode. This can be shown as follows. Each expert is a t-distribution on the response to a linear filter, which is a unimodal distribution whose mode is at zero. Any patch that is orthogonal to the linear filter will generate a zero response, so be at the mode. Since these unimodal distributions are multiplied together, if any patches are at the mode of every expert distribution they must also be at the mode of the overall distribution, *and*, since each expert can only decrease moving away from the mode, their product can only decrease also, therefore that distribution must itself be unimodal.

It now remains to be shown that homogenous patches are the only patches that are at the mode of every expert. Roth & Black [RB05] learn zero-mean filters, *i.e.*

$$\sum_i J_m(i) = 0, \qquad \forall\, m \in \{1,..,M\}. \tag{5.5}$$

If H is a homogenous patch of intensity h, it then follows that

$$\mathbf{J}_m^\top \overrightarrow{H} = \sum_i J_m(i) \cdot h = h \sum_i J_m(i) = 0, \tag{5.6}$$

therefore homogenous patches must be at the mode of every expert distribution. The reason one can be sure that no other patches are is that $\mathrm{rank}([\mathbf{J}_1,..,\mathbf{J}_M]) = 24$ for the 25-d filters learnt by Roth & Black [RB05], therefore only one linear set of patches can be orthogonal to all filters, which can only be the linear set of homogenous patches.

[4]By homogenous I mean that every pixel in the patch has exactly the same colour.

Given that the prior encourages homogenous texture, in situations where there are two almost equally photoconsistent colours for a given pixel the prior will choose that which makes the texture homogenous. Such an effect is seen in the Plant sequence (figure 5.9), where the leaf ‘ribs’ are replaced with the colour of the surrounding region. In this situation, a prior which gives the natural ribs a higher texture cost than homogenous texture will always generate such errors when the difference in data costs between the two colours is small, suggesting that the value of λ_{d} might have to be lowered to such a degree to stop this that other, genuine artifacts would not fixed. By contrast, the multi-modal prior of the previous chapter avoided this problem by making such patches equally likely. Appendix C describes a modification to the prior model of equation 5.1, introduced in [WRTF06], that allows multi-modal distributions to be modelled.

The final point of interest is that the prior aligns texture edges to the horizontal or vertical. If a texture discontinuity must exist then the robust nature of the t-distribution kernel encourages it to cover a smaller area, even if the resulting edge is of greater intensity. The reason for the aligned structure of such sharp edges is that fewer $n \times n$ patches will overlap the edge in this formation than in any other, reducing costs further. In the areas where this occurs, which tend to be partially occluded areas, the data costs do not discourage it because there are two likely colours, one at the background disparity and one at the foreground disparity.

5.5 Conclusion

This chapter has applied the general, parametric, Field of Experts prior to the task of NVS. In doing so it has introduced a relatively new, large clique, MRF optimizer—Iterated Local Search, novel approaches to perturbing the current solution, and a simple, deterministic method for fusing two solutions of a such a problem which guarantees to not increase the energy. This and two other, standard, large clique MRF optimizers were compared in the context of this problem.

The FoE prior was found to have two main drawbacks when applied to NVS: it can over-smooth images, and it can generate unnatural looking, sawtooth edges. Using ILS with Disparity smoothing kicks was found to mitigate these effects somewhat by finding solutions that are qualitatively more plausible, albeit of a higher energy.

Chapter 6

Pairwise clique priors

The previous two chapters have investigated the performance of large, 5×5 clique priors, and shown that such priors pose a challenge to optimize. This chapter marks a shift in the thesis towards lower order priors, which can be optimized more readily.

This chapter details work on developing a pairwise image prior that operates on 2×1 image patches, and can therefore leverage the power of global energy minimization approaches such as BP, TRW and graph cuts, while still being able to distinguish natural texture from unnatural texture. Various priors will be tested within a single optimization framework, therefore the optimization framework is introduced in the first section, followed by the forms of pairwise prior to be tested in the second section. The final two sections look at the NVS results using the various priors, and draw conclusions from these.

6.1 Optimization framework

The powerful optimizers mentioned operate on discrete labelling problems, so the original objective energy of equation 2.12 once again undergoes a discretization, similar to equation 4.5, though the exact mapping of label, $L(\mathbf{x})$, to colour, $I_0^*(\mathbf{x})$, and disparity, $D(\mathbf{x})$, is different, and is discussed further below. This problem is then stated as a sum over unary and pairwise cliques, which make up the data likelihood and prior terms respectively, thus:

$$E(\mathsf{I}_1,..,\mathsf{I}_N|\mathsf{L}) = E_{\text{photo}}(\mathsf{L}|\mathsf{I}_1,..,\mathsf{I}_N) + E_{\text{texture}}(\mathsf{L}), \tag{6.1}$$

$$E_{\text{photo}}(\mathsf{L}|\mathsf{I}_1,..,\mathsf{I}_N) = \sum_{\mathbf{x}\in\mathcal{X}_0} \phi_{\mathbf{x}}(L(\mathbf{x})), \tag{6.2}$$

$$E_{\text{texture}}(\mathsf{L}) = \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathbb{N}} \phi_{\mathbf{pq}}(L(\mathbf{p}), L(\mathbf{q})). \tag{6.3}$$

In particular, in this problem the neighbourhoods, $\mathbb{N}$, will be the set of all patches in the image with the form $\{\mathbf{x}, \mathbf{x}+[0,1]^\top\}$, $\{\mathbf{x}, \mathbf{x}+[1,0]^\top\}$, $\{\mathbf{x}, \mathbf{x}+[1,1]^\top\}$ or $\{\mathbf{x}, \mathbf{x}+[1,-1]^\top\}$. This makes the problem a regular *8-connected* graph, pixels being connected to their eight immediate neighbours. The unary and pairwise functionals, $\{\phi_{\mathbf{x}}\}_{\mathbf{x}\in\mathcal{X}_0}$ and $\{\phi_{\mathbf{pq}}\}_{\{\mathbf{p},\mathbf{q}\}\in\mathbb{N}}$, are effectively 1-d and 2-d lookup tables that can be precomputed from the respective data likelihood and prior terms, which are discussed below.

The objective function will be optimized using TRW-S [Kol06], an implementation of which is publicly available [Kol05]. This algorithm has been shown to have leading performance on regularly connected, low-level vision problems [SZS$^+$08], along with α-expansion graph cuts [BVZ01], but it additionally does not place any constraints on the form of energy, unlike the latter. TRW-S generates a lower bound on the energy which is nondecreasing with each iteration, therefore the algorithm can be terminated when this value is equal to the lowest energy found, as the solution that generates this energy is a global optimum. If the algorithm has not converged after 100 iterations it is terminated, and the current lowest energy solution output.

6.1.1 Data likelihood

Returning to the continuous problem over I_0^* and D, let us write E_{photo} as

$$E_{\text{photo}}(\mathsf{I}_1,..,\mathsf{I}_N|\mathsf{I}_0^*,\mathsf{D}) = \lambda_{\text{d}} \sum_{\mathbf{x}\in\mathcal{X}_0} f(I_0^*(\mathbf{x}), D(\mathbf{x})), \tag{6.4}$$

$$f(I_0^*(\mathbf{x}), D(\mathbf{x})) = \sum_{i=1}^{N} \rho_{\text{d}}\left(I_i\left(\pi_i\left(\mathbf{x}, D(\mathbf{x})\right)\right) - I_0^*(\mathbf{x})\right), \tag{6.5}$$

where λ_{d} is a noise parameter and $\rho_{\text{d}}(\cdot)$ is the photoconsistency kernel which models the noise. Work in previous chapters assumed that all surfaces were Lambertian and visible in all input views, which, given the assumption of *i.i.d.* Gaussian noise, produced a kernel of $\rho_{\text{d}}(\Delta I) = \|\Delta I\|^2$. In this chapter, following Strecha *et al.* [SFVG06], it is rather assumed that input samples are generated either by an *inlier* process—Lambertian reflectance from a visible surface—or by an *outlier* process, such as occlusion or specular reflectance. Inliers follow the noise model, which is again taken to be *i.i.d.* Gaussian, while outliers are assumed to come from a uniform distribution. A sample is determined to be an inlier or an outlier according to which distribution maximizes its likelihood. This generates the robust, truncated quadratic kernel:

$$\rho_{\text{d}}(\Delta I) = \min\left(\|\Delta I\|^2,\ \kappa\right), \tag{6.6}$$

where κ is a truncation threshold. The intention of using such a robust kernel is to not unduly penalize the correct colour and disparity if it happens to be occluded in some, even a majority of the input images. The threshold value chosen for the experiments is $\kappa = 50^2c$, where c is the number of colour channels.

6.1.2 Minimizing out disparity

The computational complexity of TRW-S for this problem is $\mathcal{O}(nM^2)$, n being the number of pixels and M being the number of labels per node. This quadratic dependence on the number of labels contrasts with the linear dependence of previous optimization approaches used in this thesis, and generates a need to reduce the number of labels as much as possible, in order to make the problem computationally tractable.

A standard approach to reducing the number of labels is to take them to be the modes of a distribution [FWZ05, YC04], rather than regularly sampling a distribution. In this way the important parts of the distribution are kept, while the redundant parts are removed.

The quadratic kernel used previously was convex, generating a single minimum over colour for equation 6.5 at each disparity. The label space was discretized over disparity, using the cost minimizing colour at each disparity, effectively generating the following overloaded cost function:

$$f(D(\mathbf{x})) = \min_I f(I, D(\mathbf{x})). \tag{6.7}$$

Yao & Cham [YC04] used the modes of the above distribution in their NVS framework. Fitzgibbon *et al.* [FWZ05] defined a different overloaded cost function, preferring to minimize out disparity, thus:

$$f(I_0^*(\mathbf{x})) = \min_{d_{\text{min}}<d<d_{\text{max}}} f(I_0^*(\mathbf{x}), d), \tag{6.8}$$

then found modes in the resulting $\mathbb{R}^3$ space (in the case of RGB colour). I follow this latter approach, for several reasons:

- Disparity is not involved in the regularization, as the priors involved are image priors—colour, by contrast, is. It is therefore more important for the labels to represent the distribution over colours well. Equally, if the same colour minimizes equation 6.5 at several values of $D(\mathbf{x})$ then choosing modes over disparity would generate redundancy in the label set.
- As the new photoconsistency kernel of equation 6.6 is concave, equation 6.5 can now have several minima at different colours for a given $D(\mathbf{x})$, the less photoconsistent of which might be more plausible in the context of the surrounding texture. It is therefore important to have all these colours represented, rather than one per disparity.
- Fitzgibbon *et al.* [FWZ05] showed that there are far fewer modes over colour than over disparity, for a convex kernel. Figure 6.1(b) demonstrates that this is also the case for the truncated quadratic kernel used here. Therefore, in order to minimize the number of labels generated it is preferable to use modes over colour.

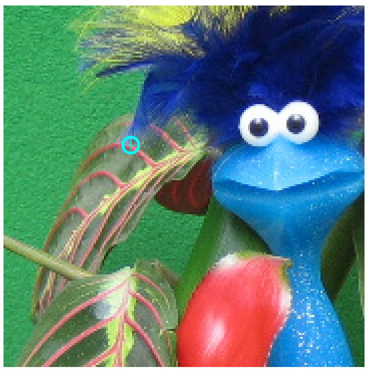
In practice the disparity range searched over, $(d_{\text{min}}, d_{\text{max}})$, is discretized into a set of disparities, $\mathcal{D}$, using the discretization described in §4.2.1.3.

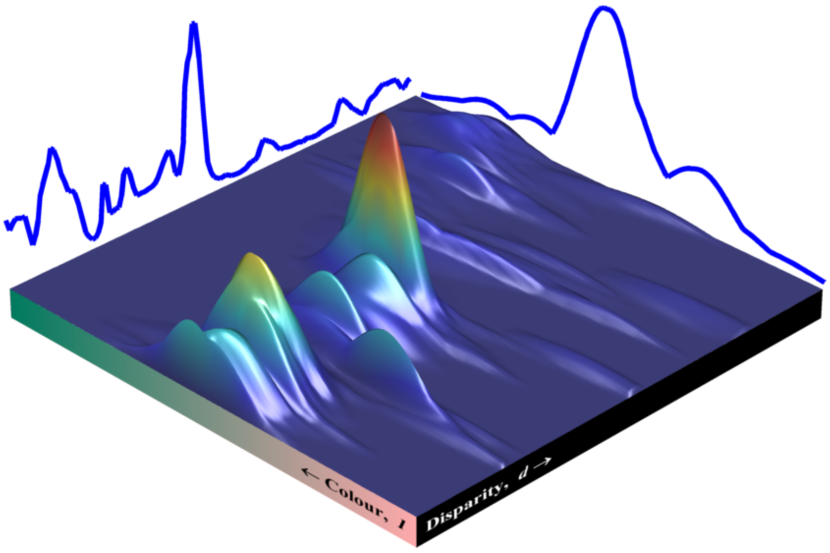
6.1.3 Finding colour modes

Fitzgibbon *et al.* [FWZ05] use gradient descent in RGB space to find the colour modes for each output pixel, starting from 20 random colours, as shown in figure 6.2(a). As the truncated quadratic kernel used here has zero gradient in the truncated region (equating to the region outside the outer-most iso-surface in figure 6.2), many of the starting points do not move, hence generate random colour modes. A better approach in this case is to start from 20 random input sample colours, as shown in figure 6.2(b), as these points are guaranteed to be inside the non-truncated region. However, this does not overcome the main drawbacks of this approach, which are its large computational burden[1] and ability to miss modes.

In this section I introduce two novel contributions which allow the modes of the distribution to be found both faster and completely reliably. The first is to develop a deterministic algorithm for extracting *all* the modes of $f(I_0^*(\mathbf{x}))$ given a suitable superset of colours—the modes of $\{f(I_0^*(\mathbf{x}), d)\}_d$—

[1]On today's hardware, finding modes using this method takes 0.01–0.1 seconds per pixel.


a



b

Figure 6.1: Data likelihood. (a) The Plant sequence ground truth image with a pixel highlighted (by the cyan crosshairs). (b) A visualization of the data likelihood of the highlighted pixel from (a), computed as $\exp(-f(I,d))$, over disparity and the principal component of colour, computed from the input samples. Also shown are the min-marginals (blue lines), obtained by minimizing out disparity and colour as per equations 6.7 and 6.8 respectively.

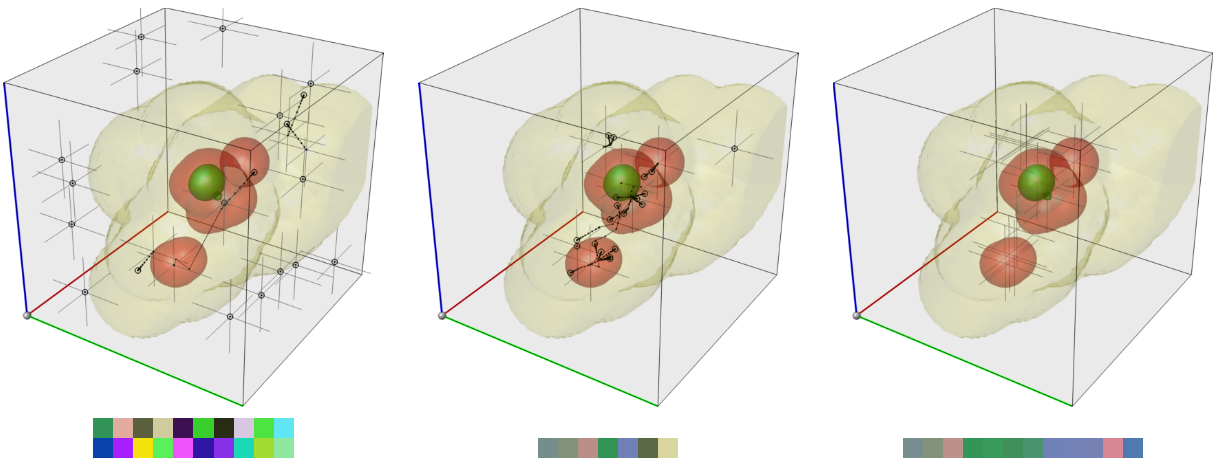

a b c

Figure 6.2: Finding colour modes. Iso-surfaces of $f(I_0^*(\mathbf{x}))$ in RGB space (*top*) for the pixel highlighted in figure 6.1(a), and the modes found (*bottom*), ordered in ascending order of cost, using the following approaches: (a) gradient descent from 20 random starting points, (b) gradient descent from 20 random input samples, and (c) using the method introduced in §6.1.3. The location of modes in RGB space are indicated by grey crosses, while the steps taken in the gradient descent approaches are indicated by the black lines, with starting points circled.

for *any* photoconsistency kernel, $\rho_{\mathrm{d}}(\cdot)$. The second is to provide a means of finding *every* mode of $\{f(I_0^*(\mathbf{x}), d)\}_d$ for the truncated quadratic kernel.

Writing I for $I_0^*(\mathbf{x})$, let I' be a mode of $f(I)$. The requirement of a mode is

$$f(I') < f(I' + \delta I) \tag{6.9}$$

for all sufficiently small $\delta I \in \mathbb{R}^3$. From equation 6.8 it is possible to define the disparity at which the mode I' can be found, thus

$$d' = \operatorname*{argmin}_{d \in \mathcal{D}} f(I', d). \tag{6.10}$$

Consider that if

$$f(I', d') \geq f(I' + \delta I, d'), \tag{6.11}$$

then equation 6.9 cannot be true. As a result, it can be seen that a colour mode over all depths has the following properties:

[P1] *It must necessarily be a colour mode of the disparity d'.*

[P2] *The disparity d' must also be the disparity at which $f(I', d)$ is lowest.*

A deterministic method of finding colour modes is therefore:

1. For each discretized disparity d_i, enumerate all colour minima of $f(I, d_i)$.
2. For each such minimum, denoted (I', d_i), reject it as a mode if $f(I', d_i) > \min_{d \in \mathcal{D}} f(I', d)$.

6.1.3.1 Enumerating colour modes at each disparity

The problem of enumerating all colour minima of $f(I, d)$ at a given disparity, d, depends on the form of the photoconsistency kernel, $\rho_{\mathrm{d}}(\cdot)$. For example, a quadratic kernel is trivially shown to have a single mode: the mean of input samples. This closed form computation means that modes over colour for this kernel can be computed quickly and reliably using the algorithm described above. Indeed, the reliability of this method in finding modes over colour depends only on the ability to find them first at each disparity. It is therefore generally optimal for all convex kernels, which produce a single minimum at each depth, as those modes can always be computed using a standard gradient-based optimizer.

The kernel used here is concave, and can therefore generate multiple minima at each disparity. However, given a set of inlying input samples (*i.e.* samples in the quadratic region of the kernel), the cost minimizing colour (a local minimum) can be computed as the mean of these samples. The test of a mode for this particular kernel is therefore that the mode's colour is the mean of the inlying input samples for that colour. This is to say that a mode, I', for a pixel $\mathbf{x}$ at disparity d, satisfies

$$I' = \frac{\sum_{i=1}^{N} I_i\left(\pi_i\left(\mathbf{x}, d\right)\right) \cdot [\|I_i\left(\pi_i\left(\mathbf{x}, d\right)\right) - I'\|^2 < \kappa]}{\sum_{i=1}^{N} [\|I_i\left(\pi_i\left(\mathbf{x}, d\right)\right) - I'\|^2 < \kappa]}. \tag{6.12}$$

In previous work [WRTF07] the modes were found using an iterative, mean shift algorithm, starting from the mean of every pair of input samples (*i.e.* N^2 colours). However, it is possible for this method to occasionally miss modes. To be sure of finding every mode, the approach used here is to test whether the mean of a given subset of input samples is a mode, for every possible combination of input samples, *i.e.* 2^N colours. While the complexity is now exponential in N, for the small number of input images used here (4–6), computational cost is similar to the previous method. In addition the approach is not iterative, and certain combinations of input samples are rejected prior to computing the mean and performing the test of equation 6.12, using the methods described below.

Colour modes which have only one inlier (i.e. $\sum_{k=1}^{n} [\|I_i\left(\pi_i\left(\mathbf{x}, d\right)\right) - I'\|^2 < \kappa] = 1$) exhibit no consistency between input images. While they may provide the correct colour, this only occurs when that colour is only ever visible in one view at most, which is rarely the case. Sample combinations with fewer than two samples are therefore ignored.

The diameter of the convex portion of the truncated quadratic kernel in colour space is $2\sqrt{\kappa}$. Any pair of input samples further than this distance apart, *i.e.* for which

$$\|I_i\left(\pi_i\left(\mathbf{x}, d\right)\right) - I_j\left(\pi_j\left(\mathbf{x}, d\right)\right)\|^2 > 2^2\kappa \tag{6.13}$$

cannot therefore both be inliers of the same mode. All pairs which satisfy the above criterion can be precomputed in $\mathcal{O}(N^2)$ time, and sample combinations which contain any of these sample pairs can be ignored.

6.1.4 Graph clique potentials

The above procedure is used to find the modes of $f(I_0^*(\mathbf{x}))$ for each pixel—the number of modes, and therefore labels, will vary from pixel to pixel. The colour of each mode is stored in the colour data structure, C, such that $C(\mathbf{x}, L(\mathbf{x}))$ returns the colour of the mode indicated by the label $L(\mathbf{x})$ for pixel $\mathbf{x}$, and the output image, I_0^*, is generated using equation 4.9.

Given the label colours, the graph clique potentials are cached for every combination of input labels prior to solving equation 6.1 using TRW-S. Each entry in the unary potential tables is computed as

$$\phi_{\mathbf{x}}(L(\mathbf{x})) = \lambda_{\mathrm{d}} \min_{d \in \mathcal{D}} \sum_{i=1}^{N} \rho_{\mathrm{d}}(I_i(\pi_i(\mathbf{x}, d)) - C(\mathbf{x}, L(\mathbf{x}))). \tag{6.14}$$

Each potential in the pairwise potential tables is computed as

$$\phi_{\mathbf{pq}}(L(\mathbf{p}), L(\mathbf{q})) = g_{\mathbf{pq}}(C(\mathbf{p}, L(\mathbf{p})), C(\mathbf{q}, L(\mathbf{q}))). \tag{6.15}$$

where $g_{\mathbf{pq}}(\cdot)$ is the cost function resulting from the pairwise image prior. The forms that this cost function takes are the subject of the next section.

6.2 Pairwise prior functionals

This section describes the various pairwise prior functionals that will be investigated in the following section. The first prior is a parametric, filter-based prior, while the following three priors are all non-parametric, example-based priors, each using different exemplar libraries.

6.2.1 Parametric prior

The parametric prior used here models the sparse nature of the response to a derivative filter in images, discussed in §3.3.1. I use the model proposed by Tappen *et al.* [TRF03], which gives each pairwise the clique the same cost function,

$$g(I, J) = \|I - J\|^{0.7}. \tag{6.16}$$

This prior, which is the only one to be tested that is not learned/constructed from the input image sequence, will be referred to as "Sparse".

6.2.2 Non-parametric priors

The non-parametric priors used are based on the distance-to-closest-exemplar models of [EL99, WL00, CB04, FWZ05], generating the following cost function:

$$g_{\mathbf{pq}}(I, J) = \min_{T \in \mathbb{T}_{\mathbf{pq}}} \|\overrightarrow{T} - \overrightarrow{\{I, J\}}\|^2, \tag{6.17}$$

where $\mathbb{T}_{\mathbf{pq}}$ is a patch library specific to the neighbourhood $\{\mathbf{p}, \mathbf{q}\}$. I use three types of patch library, as follows.

6.2.2.1 Global library

The first set of patch libraries are created from the set of all patches in the input sequence with the same shape as the output patch, as per [EL99, WL00, FWZ05]. Each of the four different neighbourhood shapes therefore has an associated patch library, denoted $\mathbb{T}_{[0,1]}$, *etc.* according to offset of the second pixel from the first. The texture library for each clique functional, $g_{\mathbf{pq}}(\cdot)$, is then assigned as follows.

$$\mathbb{T}_{\mathbf{pq}} = \begin{cases} \mathbb{T}_{[0,1]} & \text{if } \mathbf{p} - \mathbf{q} = [0, 1]^\top \\ \mathbb{T}_{[1,0]} & \text{if } \mathbf{p} - \mathbf{q} = [1, 0]^\top \\ \mathbb{T}_{[1,1]} & \text{if } \mathbf{p} - \mathbf{q} = [1, 1]^\top \\ \mathbb{T}_{[1,-1]} & \text{if } \mathbf{p} - \mathbf{q} = [1, -1]^\top \end{cases} \tag{6.18}$$

This prior will be referred to as “Global”.

6.2.2.2 Local library

The second set of patch libraries are created from the set of all patches from constrained regions of the input sequence which again have the same shape as the output patch. The regions are constrained to those input image pixels around the epipolar lines cast by the two output pixels in the neighbourhood, as illustrated in figure 6.3, in a similar fashion to the approach used in [CB04]. Specifically, for the neighbourhood $\{\mathbf{p}, \mathbf{q}\}$, the corresponding constrained region in the ith input image is given by the set of input image pixels which satisfy

$$\min_{d \in \mathcal{D}} \min\left(\|\mathbf{x} - \pi_i(\mathbf{p}, d)\|_\infty, \|\mathbf{x} - \pi_i(\mathbf{q}, d)\|_\infty\right) < 0.5 + \varphi, \tag{6.19}$$

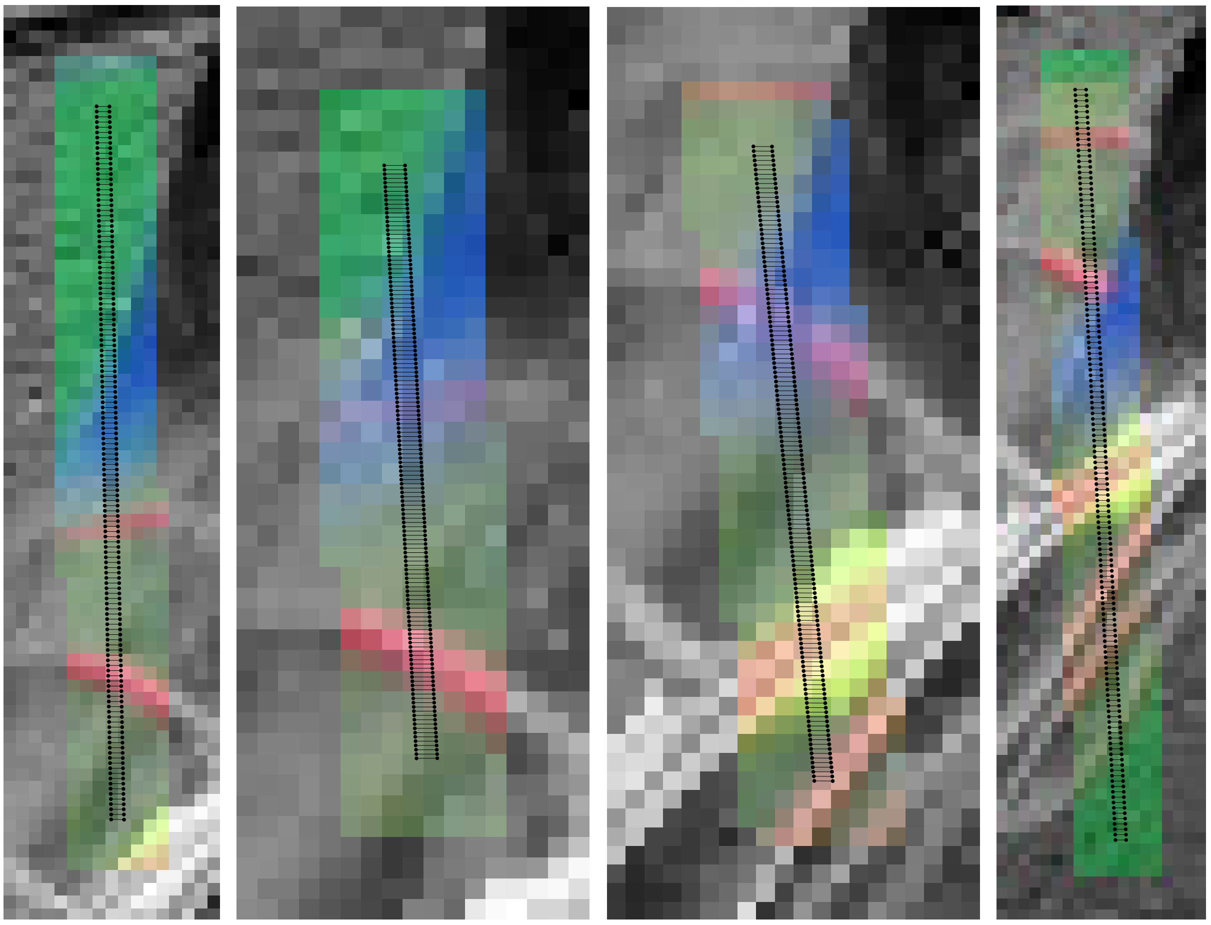


Figure 6.3: Local and sampled patch libraries. The patch libraries $\mathbb{T}_{\mathbf{x}\mathbf{x}+[1,0]^\top}$, where $\mathbf{x}$ is the pixel indicated in figure 6.1(a), for the Local and Sampled priors. The Local library, with $\varphi = 3$, consists of all 2×1 patches in the coloured (*i.e.* non-greyscale) regions of the input images. The Sampled library consists of all samples from the points indicated by conjoined black dots, which trace out the epipolar lines of the two pixels in the neighbourhood. The libraries are a concatenation of libraries from individual input images.

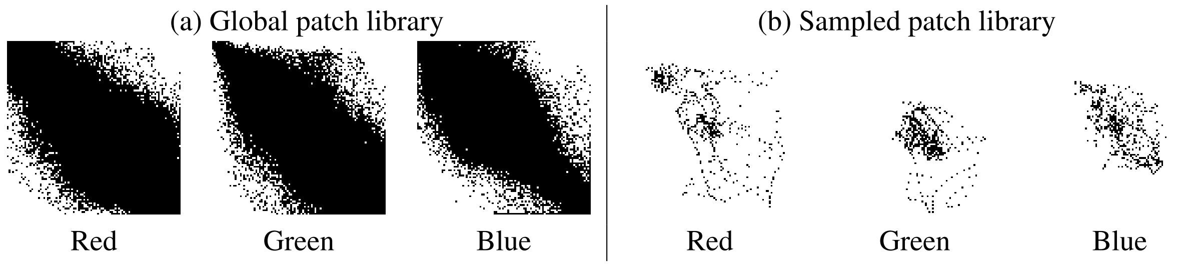


Figure 6.4: Patch library histograms. Representations of patch libraries $\mathbb{T}_{\mathbf{x}\mathbf{x}+[1,0]^\top}$, where $\mathbf{x}$ is the pixel indicated in figure 6.1(a), for the (a) Global and (b) Sampled priors. The libraries are sets of 6-d exemplars (in the case of RGB images) that are represented here by three 2-d histograms, one over each colour channel—the intensities of the two pixels in each exemplar are represented along the vertical and horizontal axes respectively. As E_{texture} is a function of the nearest exemplar, not the exemplar density, the histograms are binary, with black indicating there is at least one exemplar at that location, white otherwise.

where $\mathbf{x}$ is an input image pixel and φ is a scalar value controlling the dilation of the region. The prior will be tested with $\varphi = 0$ (meaning that only patches which intersect an epipolar line are included), 1 and 3; as φ becomes larger, the patch libraries become more similar to those of the Global prior. This prior will be referred to as "Local".

6.2.2.3 Sampled library

The final set of patch libraries are created from the following set of colour samples:

$$\mathbb{T}_{\mathbf{pq}} = \{\{I_i(\pi_i(\mathbf{p}, d)), I_i(\pi_i(\mathbf{q}, d))\}_{d \in \mathcal{D}}\}_{i=1}^{N} \tag{6.20}$$

Each sample point pair is a projection of a fronto-parallel output patch at some disparity into the input sequence. The samples mark out the epipolar lines of the two output pixels, as shown in figure 6.3. Unlike the previous two priors, this prior uses input image samples from non-integer locations, just like the samples used to compute the colour modes—indeed, the samples are in fact the same. The projection and sampling of output patches in input views to generate an exemplar library is a novel contribution of this work. The resulting prior will be referred to as "Sampled".

6.3 Experiments

In this section I detail the experiments performed to compare priors.

6.3.1 Sparse prior weight

A brief investigation into a suitable value for the objective function parameter λ_d (the data cost weight), was carried out by computing results over a range of values, some of which are shown in figure 6.5. What can be seen from this figure is that results for the Sampled prior (*bottom row*) are fairly stable over λ_d. This is due to the multi-modal nature of the distribution modelled by the texture library, which aims to give all patches a zero cost, such that the weight on this cost has no effect once this is achieved. The effect is clearly demonstrated in figure 6.7(c), where all but the few pixels around rendering errors have near zero cost. A value of $\lambda_\text{d} = 1$ was used for this prior and all other exemplar-based priors.

Results with the Sparse prior (*top row* of figure 6.5) are much more sensitive to noise because far fewer patches (only those with uniform colour) generate zero cost, as can be seen in figure 6.7(b). The

Values given per output pixel:	Prior computation time (ms)	Energy minimization (% > lower bound)	Monkey ϵ_{rms}	Monkey ϵ_{ge}	Plant ϵ_{rms}	Plant ϵ_{ge}
Sparse	0.32	4.6×10^{-4}	5.79	1.41	13.05	9.43
Global	3100	0.0040	5.49	1.24	13.09	9.30
Local, $\varphi = 0$	2.9	0.078	5.73	1.28	13.83	9.94
Local, $\varphi = 1$	3.9	0.051	5.47	1.14	13.49	9.53
Local, $\varphi = 3$	7	0.022	5.49	1.17	13.16	9.29
Sampled	0.41	0.040	5.18	0.92	11.8	8.41

Table 6.1: Quantitative results. A table of quantitative results for all pairwise priors.

result is that too low weights can over-smooth the image, while too high weights will leave artifacts uncorrected. A value of $\lambda_{\text{d}} = 1/4$ for the Sparse prior was found to strike a reasonable balance between these two extremes. These weights were used in the remaining experiments.

6.3.2 Quantitative results

The four priors were compared on six quantitative measures: the time taken to cache all the prior term costs, the energy of the solution found using TRW-S, given as a percentage increase over the lower bound also found, and r.m.s. and gross pixel errors for the Monkey and Plant sequences. Where appropriate, measures have been averaged over output pixels and across the four sequences. The results are shown in table 6.1, with image quality results summarized by the bar chart of figure 6.6.

6.3.2.1 Computational performance

In terms of time taken to cache the prior clique energy tables, the parametric Sparse prior is fastest, and the Global prior, which uses a very large exemplar library, is the slowest by between two and four orders of magnitude; the other exemplar-based methods are slightly slower than Sparse, and decrease in speed as the library size increases. TRW-S then takes around 0.55ms per pixel to optimize the resulting problem for all of the priors, which is to be expected as the problems are all the same size. In terms of minimization performance however, the Sparse prior energies generated are closest to their lower bounds, and an order of magnitude better than the Global prior which follows. The remaining priors are another order of magnitude worse than the Global prior, but all priors generate energies within a small fraction of a percent of their lower bounds, indicating that TRW-S is capable of finding good minima on this problem.

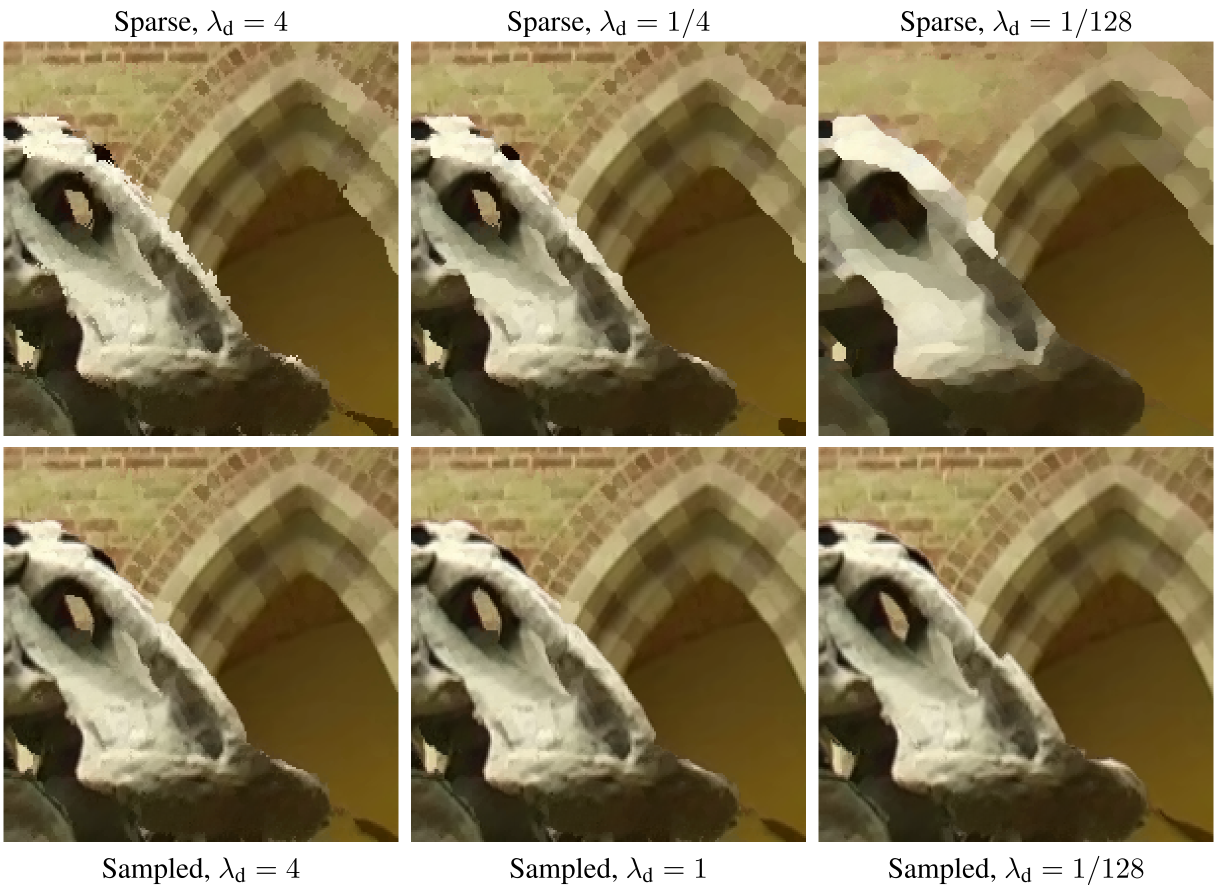

Sparse, $\lambda_d = 4$
Sparse, $\lambda_d = 1/4$
Sparse, $\lambda_d = 1/128$
Sampled, $\lambda_d = 4$
Sampled, $\lambda_d = 1$
Sampled, $\lambda_d = 1/128$

Figure 6.5: Prior weights. A range of values for λ_d were tested with the Sparse (*top row*) and Sampled (*bottom row*) priors and the results compared, in order to select a reasonable value.

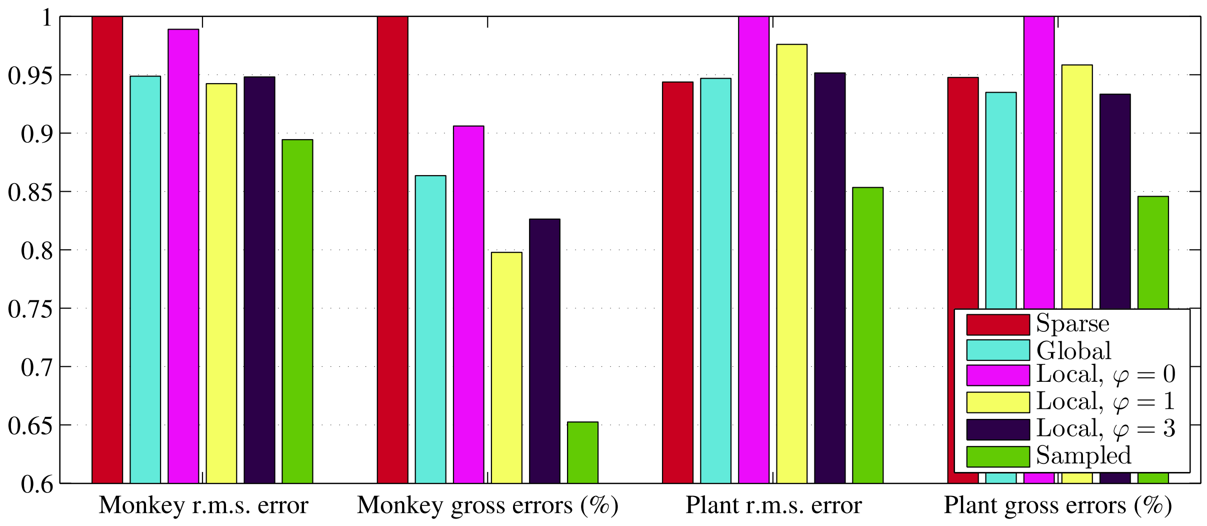


Figure 6.6: Normalized results. A visual representation of the quantitative image quality results from table 6.1, linearly normalized so that the highest value in each category is 1.

In terms of image quality, figure 6.6 shows that the Sampled prior performs considerably better than the other priors at all four measures. The relative performance of the Local prior with different values of φ is interesting. There is a trade-off to be had between having a library that is large enough to contain the correct patch, but small enough to distinguish it from most other, wrong patches. On the Plant sequence the results improve as φ increases, while on the Monkey sequence they peak at $\varphi = 1$. No real conclusions can be drawn about the relative performance of the other priors.

6.3.3 Qualitative results

I will now look at the qualitative results, focussing on some particular areas of reconstruction.

6.3.3.1 Large scale errors

The local optimization algorithms of previous chapters struggled to fix large scale errors in the ML images without the help of a multi-resolution strategy. Figure 6.7 shows results on a large scale error (seen in (a))—the cabinet frame in the Dino1 sequence. Both the Sparse and Global priors fail to correct the error, while the Local and Sampled priors succeed. As TRW-S is able to find good minima, one can be reasonably sure that the artifacts are a result of the priors not being able to differentiate between the correct and incorrect patches. The Sparse and Global priors do not have clique-specific prior functionals, so the functionals must accept a wide variety of patches, including patches which will be wrong in a given location, while the clique-specific Local and Sampled priors can afford to be more discriminative. This effect can be seen in the texture library representations of figure 6.4, which show that the Sampled prior library is much sparser, and will therefore apply a high cost to many more patches, than the Global prior library.

The snout of the Dino2 sequence, shown in figure 6.8, is another area of large scale error. Again, the clique-specific Local and Sampled priors perform much better, correctly reconstructing most of the bridge of the snout. It is also interesting to note that the Sparse prior (B) generates a much more piecewise-constant image, with fewer gradients, while the Global prior (A) generates an image with sharper edges.

The Plant and Monkey sequences (figures 6.10 and 6.11 respectively) contain errors which only the Sampled prior succeeds in fixing—the ‘ribs’ on the leaves of the former, and an area of fur in the upper right corner of the zooms of the latter. A potential reason for this is that the output images, which are generated from bilinearly interpolated samples of input images, tend to have smoother gradients in textured regions than the input images themselves. This effect can be seen by comparing the leaf and feathers in figure 6.9(c & d). The Sampled prior, whose patch libraries are also generated from bilinearly interpolated samples, will therefore match the correct patch well, while the Local prior libraries will contain the less smooth input image patches, hence match less well. The Local prior will therefore tend to discourage regions of high image gradient where a suitable, homogenous alternative (with reasonable data likelihood) exists.

An area that all the priors, including Sampled, fail to reconstruct correctly is the stalk to the left of the plant sequence, shown in zoom 2 of figure 6.10.

6.3.3.2 Occluded regions

A failure mode of all the image priors tested here is the reconstruction of texture in partially occluded regions. An example of this is seen under the Monkey's arm, in the zooms of figure 6.11—dark vertical lines are filled in with the surrounding, lighter colour. Another example is seen on the leaf below the blue feathers in Plant, in zoom 2 of figure 6.10—the pink ribs of the leaf are replaced by the surrounding green colour in the partially occluded region.

Even with the robust, truncated quadratic photoconsistency kernel to reject outliers caused by, amongst other things, occlusions (which has some effect, as discussed in §6.3.5) the correct colour has a cost sufficiently high for it to be replaced by another colour with a lower cost, which will generally come from outside the occluded region.

6.3.3.3 Textured regions

Another failure mode of this approach, which occurs with all priors, is the generation of artifacts in textured regions. The low contrast, stochastic textures on the Monkey's face (figure 6.11) and the baize background of Plant (figure 6.10) are smoothed into a homogeneous region, lacking the original detail, as shown by the zooms in figure 6.9. In the brickwork of the Dino2 sequence (zooms of figure 6.8), which is higher contrast, regular texture, some of the mortar between bricks is lost.

Looking at the respective implicit disparity maps shown in figure 6.12 reveals that there are high frequency changes in disparity in the regions of failure (most pronounced at the top left of the Plant sequence (b)). One can be certain that the surfaces being reconstructed are in actual fact much smoother than this (indeed the baize of Plant and brickwork of Dino2 are planar), a fact which could be leveraged to correct the errors through the additional use of a geometry smoothness prior.

The difficulty with incorporating both geometry and texture priors in the framework presented here is memory use. One must store a separate edge cost matrix, $\phi_{\mathbf{pq}}(\cdot)$ for each clique $\{\mathbf{p}, \mathbf{q}\}$—in order for these matrices to fit in memory each edge can have of the order of only 100 label combinations, *i.e.* an average of 10 labels per node. This provides the potential for a few colour modes at each pixel, but not enough for modes spaced densely over disparity (required for regularization of disparity). This means, due to the computational expense required by TRW-S, one can only regularize over either depth or colour, but currently not both.

6.3.4 Methods for finding modes

In §6.1.3 I introduced a new method for finding modes of data likelihood over colour. Figure 6.13(A & B) shows a comparison of results using modes generated with this method, and also the gradient-descent-based method of Fitzgibbon *et al.* [FWZ05]. Not only is the new approach 30 times faster (though the code is more heavily optimized), but, because it finds all modes reliably, it does not suffer from the pixelwise artifacts seen in (A), where the correct colour mode has been missed by the mode-finding algorithm.

6.3.5 Robust photoconsistency kernel

In this chapter a robust photoconsistency kernel is introduced. The difference that the robust kernel makes can be seen by comparing figure 6.13(B & C). Using the truncated quadratic kernel (C) generates slightly fewer artifacts in partially occluded regions than the quadratic kernel. This is to be expected, as the robust kernel truncates the effect of colour outliers generated by occlusions, not only allowing the correct colour to be a mode, but also to have a low cost. However, the majority of partially occluded regions are still incorrectly rendered, as discussed in §6.3.3.2.

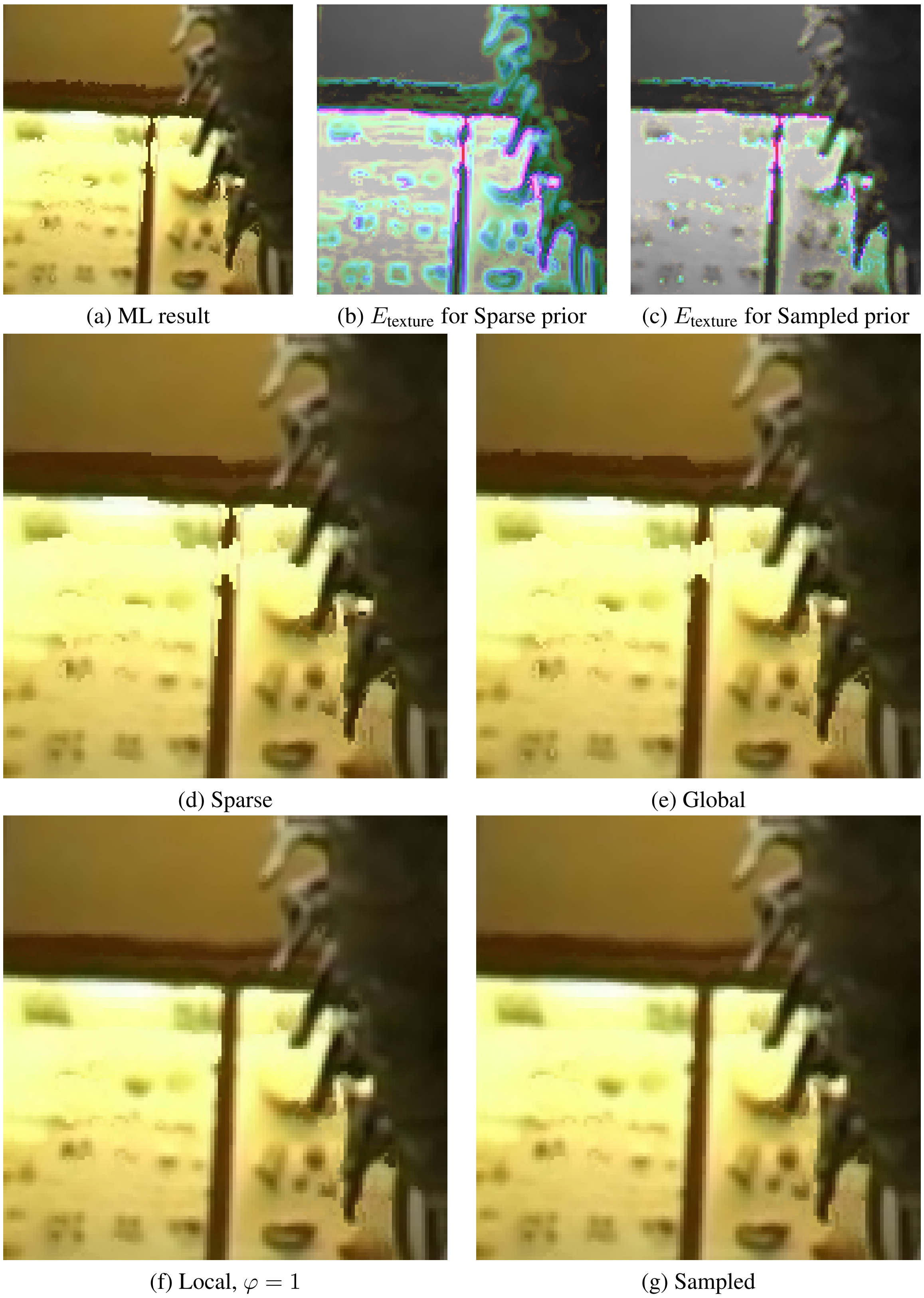
6.4 Conclusion

This chapter has shown that the global energy minimization algorithm, TRW-S, is able fix large areas of error that the local optimization algorithms of previous chapters failed to correct (in single resolution frameworks). In doing so, it has compared the performance of standard pairwise parametric and non-parametric image priors, and shown that the fact that powerful optimization techniques can, in practice, only be applied to two-pixel patches poses a problem, as such small cliques tend to lack the ability to distinguish between natural and unnatural texture, which is required for good regularization.

The discriminative power of example-based priors can be improved by restricting the training data for the prior to local regions within the sequence. This improves the results and confers a further, considerable speed advantage. I have shown how to construct a clique-specific patch dictionary that shows leading edge results for a pairwise image prior, by sampling input images at the projected locations of fronto-parallel patches from the output view. At first sight, such a library might simply appear to encode the prior that 2-pixel cliques are fronto-parallel in the new view. However, at occlusion boundaries the library contains examples of the transition across that boundary, permitting the correct, discontinuous (in disparity space) reconstruction with a low cost, as seen in figure 6.7(c).

In addition, I have introduced a fast, reliable algorithm for enumerating all modes of data likelihood over colour, even with the non-convex, truncated quadratic kernel.

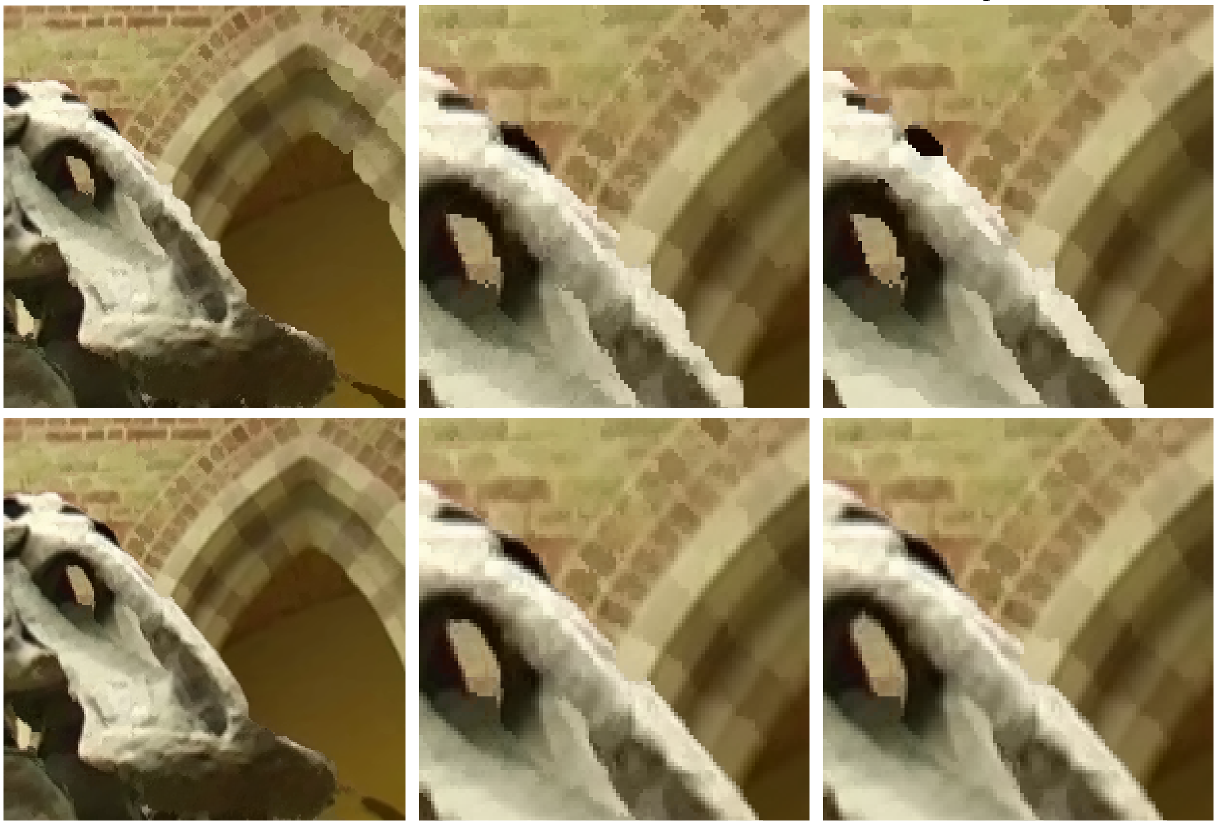
Failure modes of all the priors, including the Sampled prior, are the occasional generation of artifacts in occluded areas and on smooth, textured surfaces. It is believed that the incorporation of disparity regularization could fix these, a problem which will be tackled in the next chapter.


(a) ML result (b) E_{texture} for Sparse prior (c) E_{texture} for Sampled prior

(d) Sparse (e) Global

(f) Local, $\varphi = 1$ (g) Sampled

Figure 6.7: Dino1 results. (a) The ML result for the Dino1 sequence, and the same image with values of E_{texture} using the (b) Sparse and (c) Sampled priors superimposed—red is highest cost, greyscale is lowest; a log scale is used, and ranges are different between (b) & (c). (d)–(g) Output using the four pairwise priors on the Dino1 sequence.


(a) Global (A) Global zoom (B) Sparse zoom

(c) Local, $\varphi = 1$ (C) Local zoom (D) Sampled zoom

Figure 6.8: Dino2 results. Results on the Dino2 sequence for the remaining two priors not shown in figure 6.5, and a zoom for all four priors.

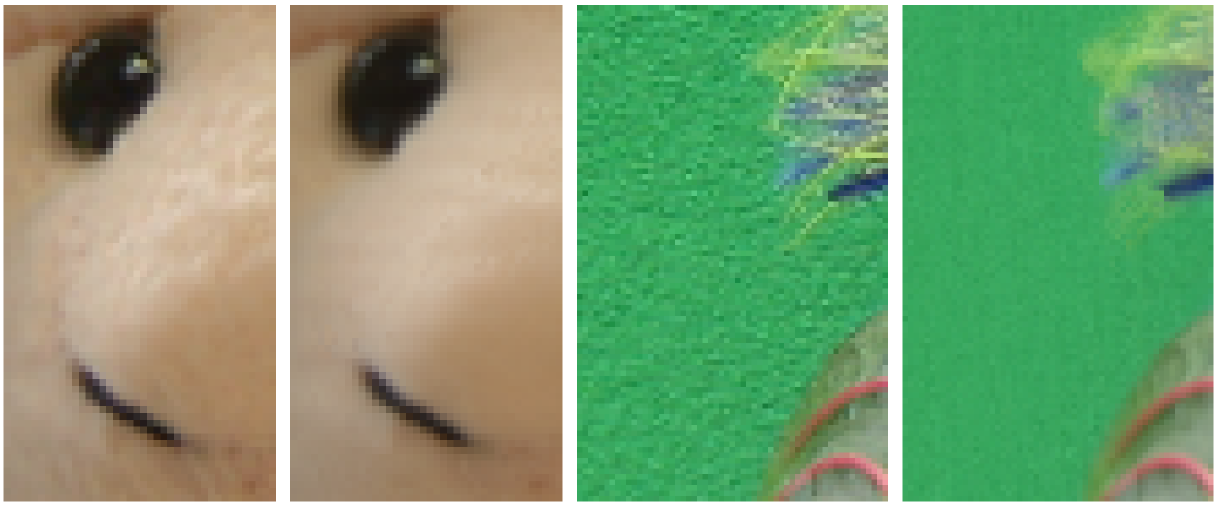

(a) Ground truth (b) Sampled output (c) Ground truth (d) Sampled output

Figure 6.9: Texture loss. Examples of low contrast, stochastic texture in the (a) Monkey and (c) Plant sequences, and their reconstructions ((b) & (d) respectively) using the Sampled prior.

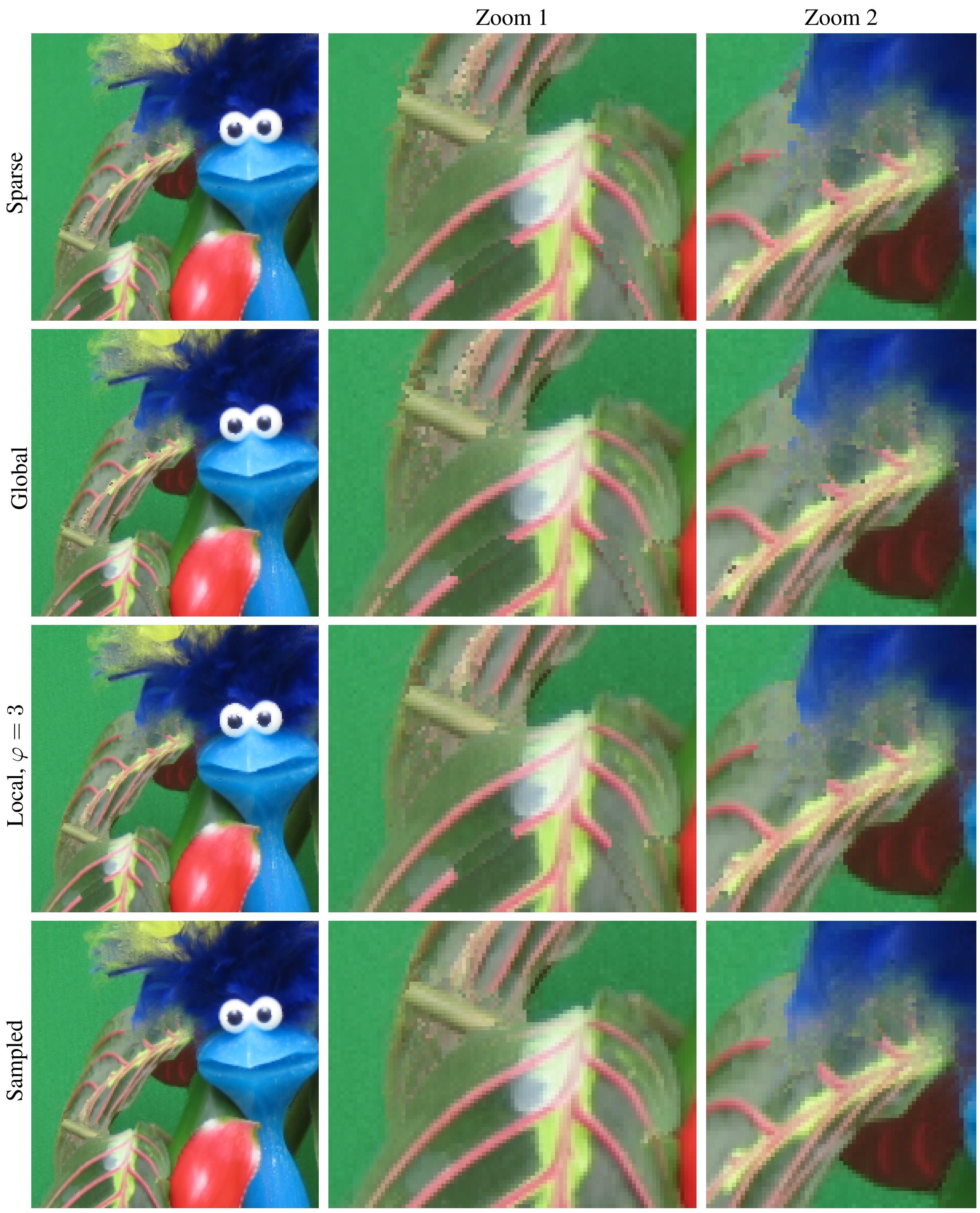


Figure 6.10: Plant results. Results of all four pairwise priors on the Plant sequence, with two zooms highlighting areas in which the priors perform differently.

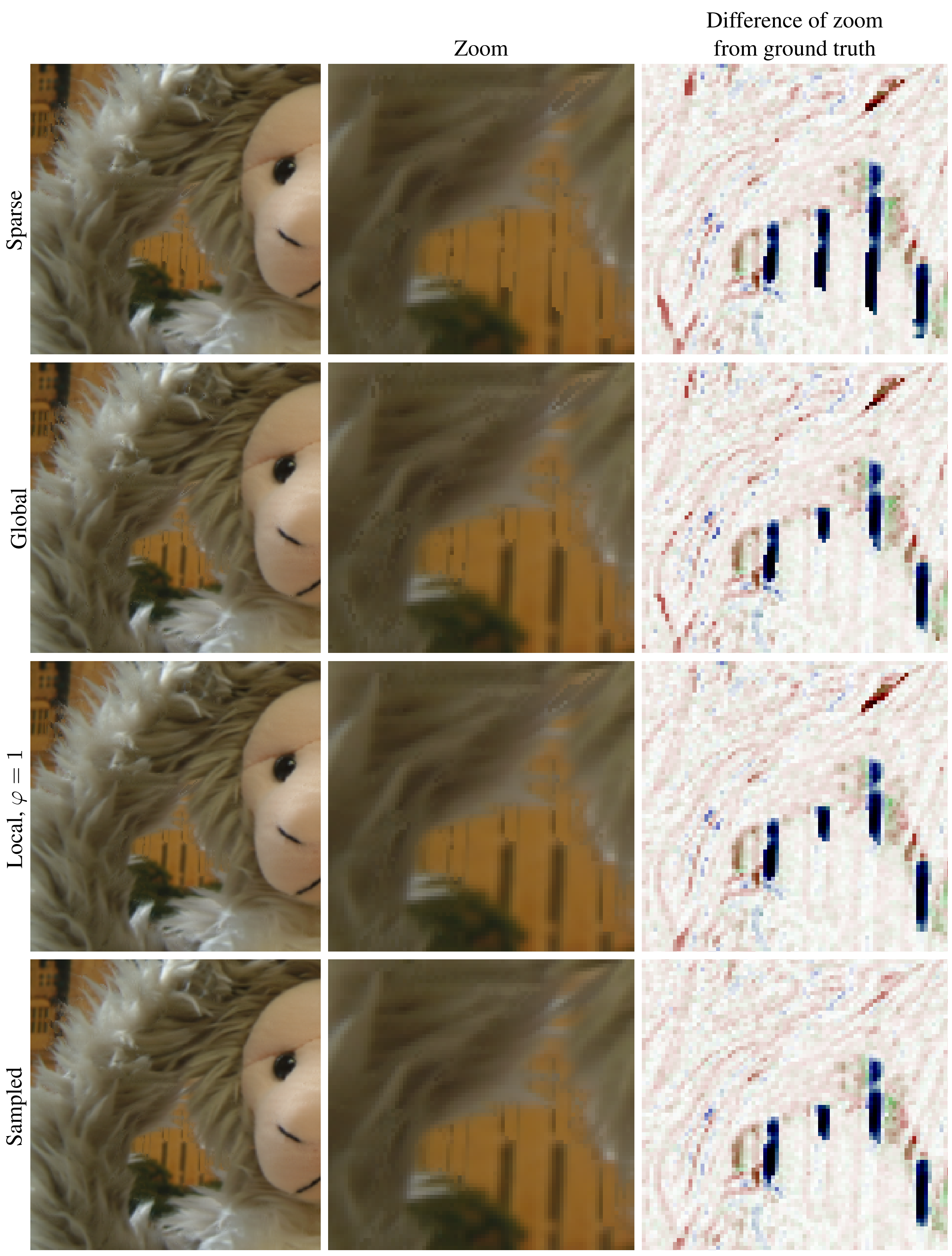


Figure 6.11: Monkey results. Results for all four pairwise priors on the Monkey sequence, with a zoom and its difference from ground truth.

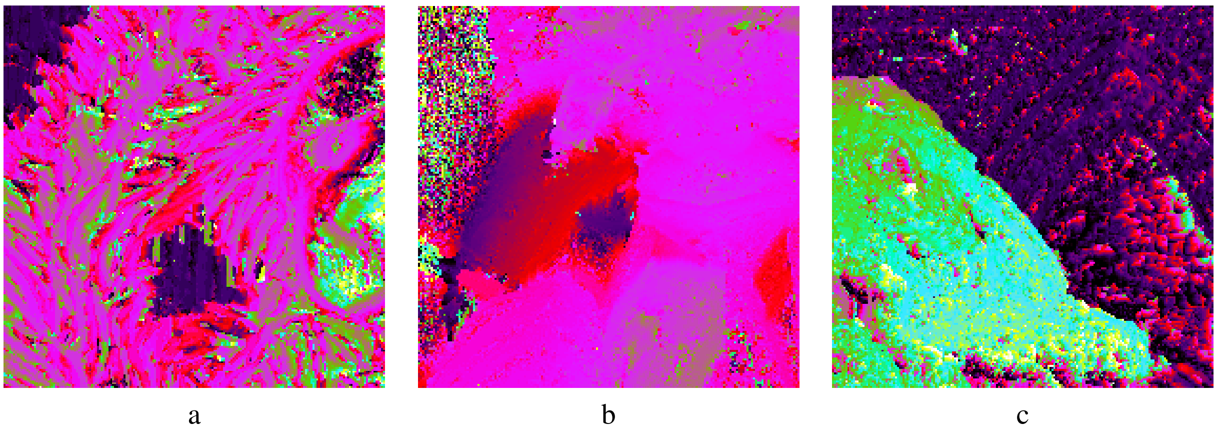


Figure 6.12: Implicit disparity. Implicit disparity maps, with each disparity value generated as $D(\mathbf{x}) = \operatorname{argmin}_{d \in \mathcal{D}} f(I_0^*(\mathbf{x}), d)$, for the (a) Monkey, (b) Plant and (c) Dino2 sequences.

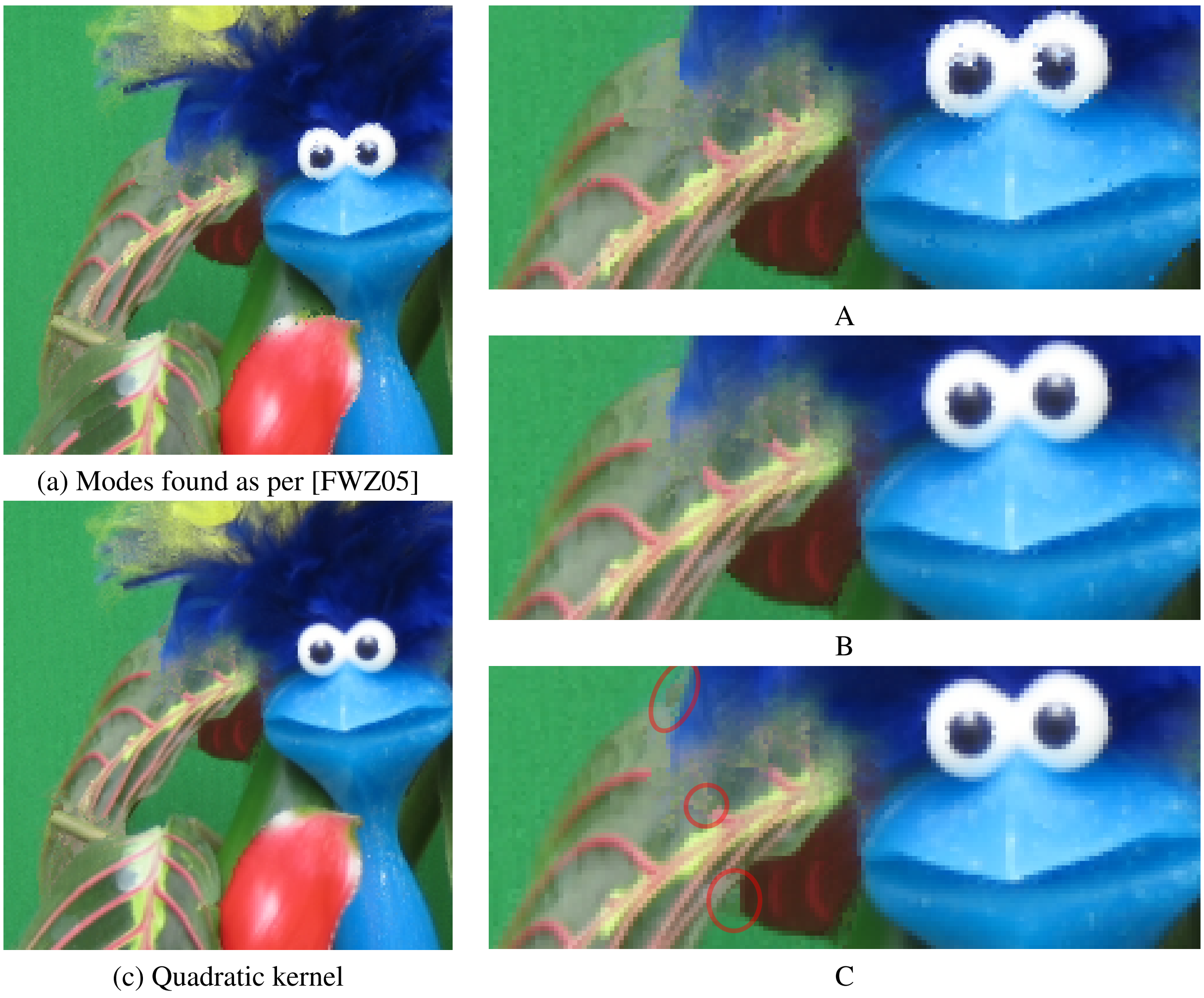


Figure 6.13: Modes and kernels. (a) Output using the Sampled prior on the Plant sequence, with modes found using gradient descent as per [FWZ05], initialized from random input samples, which took 48ms per pixel. (c) Output for the Sampled prior with a quadratic kernel, *i.e.* $\kappa = \infty$. (A) A zoom of (a). (B) A zoom of figure 6.10(d), with modes found using the deterministic approach introduced in §6.1.3, which took 1.6ms per pixel. (C) A zoom of (c), with some differences from (B) highlighted.

Chapter 7

Regularizing geometry

In the previous chapters, I have investigated the use of image priors to regularize the NVS problem. This chapter introduces regularization of the geometry of a scene to this framework, in the form of a prior on the smoothness of the implicit disparity of an output image. In addition, I use the explicit model of geometry to determine occluded pixels, and use this to reject the colour of occluded pixels as a factor in the colour of each output pixel, rather than relying on the robust data likelihood kernel of chapter 6.

Smoothness priors have been used widely by the stereo community, and as such there has been much research into forms of these priors, as well as the algorithms used to optimize the resulting objective functions. Geometrical visibility reasoning has also been used, though to a lesser degree, due to the difficulty in optimizing the resulting objective function. Some stereo algorithms have already been repurposed to the task of backward transfer NVS, as discussed in §3.2.3.

In this work I take a recently introduced stereo algorithm [WQ05], which employs a general geometrical occlusion model (*i.e.* no ordering constraint), and adapt it to the NVS domain, requiring that a number of nontrivial problems be addressed. The primary contributions are 1) simultaneously solving for depth and colour, with occlusion modelling, and 2) replacing the CRF with the efficient texture prior introduced in the previous chapter. This is the first NVS method to use a general geometrical occlusion model in a global optimization framework, and the first to combine this with a texture term.

The chapter proceeds as follows: in §7.1 I define the objective function to be optimized, which incorporates regularization of smoothness and texture, as well as the geometrical occlusion model; §7.2 then describes the optimization algorithm developed to solve this problem; the final sections present and evaluate the results, before concluding.

7.1 Objective function

As with the previous approaches, I cast this problem into an energy minimization framework, with the energy again defined at a high level by equation 2.12. In contrast to the previous chapter, however, I do not minimize disparity out of the energy, but rather keep an explicit record of the scene geometry, in the form of the disparity map, D, as per chapters 4 & 5. The disparity map is necessary for two reasons: it is used to determine which input image samples are occluded, in the data likelihood term, and it is also regularized by the prior term.

7.1.1 Data costs

As before, I assume that sampling noise is *i.i.d.* Gaussian noise. However, rather than assuming that there are no outliers due to occlusions or specularities, as in chapters 4 & 5, or using a robust kernel to reject outliers, as in chapter 6, I determine which input samples are occluded directly from the explicit disparity map, D. A boolean indicator variable, $V_i(\mathbf{x})$, indicates whether the output pixel $\mathbf{x}$, when projected into the i^{th} input image at disparity $D(\mathbf{x})$, is visible ($V_i(\mathbf{x}) = 1$) or not ($V_i(\mathbf{x}) = 0$).

The value of $V_i(\mathbf{x})$ is computed using the asymmetrical occlusion model of Wei & Quan [WQ05]—if there is another output pixel, $\mathbf{p}$, which projects to the same point[1] in I_i as pixel $\mathbf{x}$, and for which the projected depth is less than that of $\mathbf{x}$ then $V_i(\mathbf{x}) = 0$. The depth of output pixel $\mathbf{x}$ in the coordinate frame of I_i, which shall be written as $Z_i(\mathbf{x})$, is computed as

$$Z_i(\mathbf{x}) = \mathbf{P}_3^i \begin{bmatrix} \mathbf{x}/D(\mathbf{x}) \\ 1/D(\mathbf{x}) \\ 1 \end{bmatrix}, \tag{7.1}$$

where $\mathbf{P}_3^i$ is the third row of the projection matrix $\mathtt{P}_{0\to i}$, as defined in equation A.11. Therefore

$$V_i(\mathbf{x}) = 1 - \max_{\mathbf{p}\in\mathcal{X}_0} \left[\|\pi_i\left(\mathbf{x}, D(\mathbf{x})\right) - \pi_i\left(\mathbf{p}, D(\mathbf{p})\right)\|_\infty < 0.5\right] \cdot \left[Z_i(\mathbf{p}) < Z_i(\mathbf{x})\right]. \tag{7.2}$$

Occluded input samples should not be included in the evaluation of photoconsistency cost as they are not a noisy measure of the colour of the correct surface point. However, if there is no cost associated

[1] I define ‘same point’ to mean within half a pixel in both horizontal and vertical directions. This measure is an approximation, as a pixel’s projected footprint will vary according to its position and disparity. While a more accurate definition could be employed, this one was found to work suitably well.

with an occluded input sample then the objective energy will encourage occlusions where there are none, as this has a lower cost; indeed, occlusions will be encouraged wherever the photoconsistency cost for an input sample is higher than the occlusion cost. For this reason, occluded samples are given a penalty cost, ν, and the photoconsistency cost is truncated to a value of κ or less, where $\kappa < \nu$ in order to avoid encouraging occlusions. The data cost is therefore defined as

$$\begin{aligned} E_{\text{photo}}(\mathsf{I}_1, .., \mathsf{I}_N | \mathsf{I}_0^*, \mathsf{D}) &= \lambda_{\text{d}} \sum_{\mathbf{x} \in \mathcal{X}_0} \sum_{i=1}^{N} f\left(I_i\left(\pi_i\left(\mathbf{x}, D(\mathbf{x})\right)\right) - I_0^*(\mathbf{x}), V_i(\mathbf{x})\right), && (7.3) \\ f(\Delta I, V) &= V \cdot \rho_{\text{d}}(\Delta I) \; + \; (1 - V) \cdot \nu, && (7.4) \\ \rho_{\text{d}}(\Delta I) &= \min\left(\|\Delta I\|^2, \; \kappa\right). && (7.5) \end{aligned}$$

7.1.2 Prior term

Unlike the priors used in previous chapters, the one used in this chapter is a joint prior over both I_0^* and D. It is a combination of a smoothness prior found in stereo algorithms, as described in §3.2, and the image prior developed in the previous chapter. For reasons of optimizability (as with the previous chapter), the prior will be first order, with cliques defined over all 2×1 and 1×2 patches in the image, thus

$$E_{\text{prior}}(\mathsf{I}_0^*, \mathsf{D}) = \sum_{\mathbf{x} \in \mathcal{X}_0} \phi(\{\mathbf{x}, \mathbf{x} + [1, 0]^\top\}, \mathsf{D}, \mathsf{I}_0^*) + \phi(\{\mathbf{x}, \mathbf{x} + [0, 1]^\top\}, \mathsf{D}, \mathsf{I}_0^*), \tag{7.6}$$

ignoring boundary effects. Following CRF-based stereo frameworks, the clique energy takes the form

$$\phi(\{\mathbf{p}, \mathbf{q}\}, \mathsf{D}, \mathsf{I}_0^*) = W(\mathsf{I}_0^*, \{\mathbf{p}, \mathbf{q}\}) \cdot S(D(\mathbf{p}), D(\mathbf{q})). \tag{7.7}$$

The second term is the geometry regularization, or smoothness, term, which in this chapter is chosen to be a truncated linear kernel on the first derivative of disparity, thus:

$$S(D(\mathbf{p}), D(\mathbf{q})) = \min\left(|D(\mathbf{p}) - D(\mathbf{q})|, \delta_{\text{s}}\right), \tag{7.8}$$

where δ_{s} is a discontinuity preserving threshold. The first term of equation 7.7 modulates the smoothness term according to some function of the reference image, and is equivalent to the CRF terms of some stereo methods, *e.g.* [BVZ01, KZG03, SZS03, SP07]. This generally discourages disparity discontinuities from crossing areas of constant colour by aligning them with high image gradients. When I_0^* is fixed, as in the case of stereo, this works well, but when I_0^* is strongly dependent on D, as in the case of NVS, the disparity discontinuities themselves generally generate strong image gradients in I_0^*, thus creating a circular effect which reinforces the location of a discontinuity even if it is actually wrong. However, the example-based image prior, "Sampled", from the previous chapter can differentiate between plausible and implausible strong image gradients, discouraging discontinuities only where they generate texture not seen in the input sequence. This prior is therefore used as the modulating term here, thus:

$$W(\mathsf{I}_0^*, \{\mathbf{p}, \mathbf{q}\}) = 1 + \lambda_{\mathrm{t}} \min\left(\min_{T \in \mathbb{T}_{\mathbf{pq}}} \|\overrightarrow{T} - \overrightarrow{I_0^*(\{\mathbf{p}, \mathbf{q}\})}\|^2,\ \delta_{\mathrm{t}} \right), \tag{7.9}$$

where $\mathbb{T}_{\mathbf{pq}}$ is a patch library specific to the neighbourhood $\{\mathbf{p}, \mathbf{q}\}$, generated as described in the previous chapter, and λ_{t} and δ_{t} are two model parameters.

7.1.3 Computing colour

NVS differs from stereo in that one is optimizing over both colour and disparity, as opposed to just disparity, generating a continuous 4-d search space per pixel (disparity plus 3 colour channels). As with previous chapters, to aid the optimization this space is discretized into a set of labels. As disparity plays a key role in both the prior and data terms I use the discretization introduced in §4.2.1.3, generating a set of labels spaced equally over disparity and inferring colour directly from the disparity; this also enables the use of optimization frameworks commonly used in stereo. However, in contrast to those chapters, the colour of pixel $\mathbf{x}$ is computed, again assuming *i.i.d.* Gaussian noise, as the mean of *visible* (as opposed to all) input image samples, thus:

$$I_0^*(\mathbf{x}) = \frac{\sum_{i=1}^{N} V_i(\mathbf{x}) I_i\left(\pi_i\left(\mathbf{x}, D(\mathbf{x})\right)\right)}{\sum_{i=1}^{N} V_i(\mathbf{x})}. \tag{7.10}$$

The value of $I_0^*(\mathbf{x})$ computed above may not necessarily be that which minimizes equation 7.3, given $D(\mathbf{x})$, especially as E_{photo} is already robust to outliers through the truncation term, κ. However, if one assumes that all visible samples are a good match (*i.e.* inliers, as they should be for the correct solution), then equation 7.10 will give the colour that minimizes the E_{photo} term; conversely, if some visible samples are outliers (*e.g.* actually occluded) then this colour computation will increase E_{photo} more than when using the optimal colour. Therefore, equation 7.10 increases the importance of the visibility constraint.

Making colour a function of disparity allows the complete objective energy to be written in terms of

$$
\begin{aligned}
Z_i(\mathbf{x}) &:= \mathbf{P}_3^i \begin{bmatrix} \mathbf{x}/D(\mathbf{x}) \\ 1/D(\mathbf{x}) \\ 1 \end{bmatrix} \\
V_i(\mathbf{x}) &:= 1 - \max_{\mathbf{p}\in\mathcal{X}_0} \left[\|\pi_i(\mathbf{x}, D(\mathbf{x})) - \pi_i(\mathbf{p}, D(\mathbf{p}))\|_\infty < 0.5\right] \cdot \left[Z_i(\mathbf{p}) < Z_i(\mathbf{x})\right] \\
I_0^*(\mathbf{x}) &:= \frac{\sum_{i=1}^N V_i(\mathbf{x}) I_i(\pi_i(\mathbf{x}, D(\mathbf{x})))}{\sum_{i=1}^N V_i(\mathbf{x})} \\
\rho_{\text{d}}(\Delta I) &:= \min\left(\|\Delta I\|^2,\ \kappa\right) \\
f(\Delta I, V) &:= V \cdot \rho_{\text{d}}(\Delta I) \ + \ (1 - V) \cdot \nu \\
E_{\text{photo}}(\mathsf{I}_1, .., \mathsf{I}_N|\mathsf{D}) &:= \lambda_{\text{d}} \sum_{\mathbf{x}\in\mathcal{X}_0} \sum_{i=1}^N f\left(I_i(\pi_i(\mathbf{x}, D(\mathbf{x}))) - I_0^*(\mathbf{x}), V_i(\mathbf{x})\right) \\
W(\mathsf{I}_0^*, \{\mathbf{p}, \mathbf{q}\}) &:= 1 \ + \ \lambda_{\text{t}} \min\left(\min_{T\in\mathbb{T}_{\mathbf{pq}}} \|\overrightarrow{T} - \overrightarrow{I_0^*(\{\mathbf{p}, \mathbf{q}\})}\|^2,\ \delta_{\text{t}}\right) \\
S(D(\mathbf{p}), D(\mathbf{q})) &:= \min\left(|D(\mathbf{p}) - D(\mathbf{q})|, \delta_{\text{s}}\right) \\
\phi(\{\mathbf{p}, \mathbf{q}\}, \mathsf{D}, \mathsf{I}_0^*) &:= W(\mathsf{I}_0^*, \{\mathbf{p}, \mathbf{q}\}) \cdot S(D(\mathbf{p}), D(\mathbf{q})) \\
E_{\text{smooth}}(\mathsf{D}) &:= \sum_{\mathbf{x}\in\mathcal{X}_0} \phi(\{\mathbf{x}, \mathbf{x} + [1, 0]^\top\}, \mathsf{D}, \mathsf{I}_0^*) + \phi(\{\mathbf{x}, \mathbf{x} + [0, 1]^\top\}, \mathsf{D}, \mathsf{I}_0^*) \\
E(\mathsf{D}|\mathsf{I}_1, .., \mathsf{I}_N) &:= \underbrace{E_{\text{photo}}(\mathsf{I}_1, .., \mathsf{I}_N|\mathsf{D})}_{\text{data likelihood}} \ + \ \underbrace{E_{\text{smooth}}(\mathsf{D})}_{\text{smoothness cost}}
\end{aligned}
$$

Figure 7.1: Energy function. The energy $E(\mathsf{I}_0^*, \mathsf{D}|\mathsf{I}_1, .., \mathsf{I}_N)$ can be minimized as a function of the new-view disparity map, D—I_0^* and the latent visibility variables can written as a function of D and the input variables. The term E_{prior} has been changed to E_{smooth} to indicate that the prior regularizes smoothness, albeit weighted by a texture term.

D only, as shown in figure 7.1. However, the difference with this new energy is that colour is no longer a precomputed property of the label, but is rather a function of the entire labelling.

7.2 Optimization

The previous chapters have shown that powerful optimization is key to extracting maximum benefit from a given prior. The choice of optimizer in this chapter is governed by the desire to incorporate a geometrical occlusion model into the objective energy. As Kolmogorov & Rother [KR06a] empirically show, graph cuts outperforms other optimizers, such as TRW and BP, on the highly-connected graphs generated by a geometrical occlusion model. Indeed, this is precisely the optimizer used in the stereo approach [WQ05] on which this work is primarily based. For this reason, the optimization strategy developed in this chapter uses QPBO (an extension of graph cuts) in a fusion moves approach.

The fusion moves [BVZ01, WRTF06, WS06, LRB07] approach to multilabel optimization is to reduce it to an iterative sequence of binary problems, each of which combines, or fuses, a proposal solution with the current solution. The binary optimization must not increase the energy of the current solution, thus guaranteeing that the energy monotonically decreases with each iteration. In the case that the binary optimization is optimal [BVZ01, WS06], the final solution (after convergence) is assured of being less than one fusion move from the global optimum. In the stereo framework, and indeed with the approach presented here, the proposal label represents a fronto-parallel plane, D^{p}, which is fused with the current disparity map, D_t, to generate a new disparity map D_{t+1}. This is achieved by taking each pixel in D_{t+1} from either D_t or D^{p}, as controlled by a binary indicator image B with elements $B(\mathbf{x})$:

$$\mathsf{D}_{t+1} = \mathsf{B} \cdot \mathsf{D}_t + (1 - \mathsf{B}) \cdot \mathsf{D}^{\mathsf{p}}, \tag{7.11}$$

where dot indicates elementwise multiplication. Each iteration therefore becomes a question of solving the following binary optimization problem:

$$\mathsf{B} = \underset{\mathsf{B}}{\operatorname{argmin}}\, E(\mathsf{B} \cdot \mathsf{D}_t + (1 - \mathsf{B}) \cdot \mathsf{D}^{\mathsf{p}} \mid \mathbf{I}_1, .., \mathbf{I}_N). \tag{7.12}$$

7.2.1 Graph construction

This section details the graph cuts construction used to solve the binary optimization problem defined by equation 7.12.

7.2.1.1 Incorporating visibility

Several graph-cuts-based stereo algorithms [KZ02, WQ05, BG07] have included geometrical visibility reasoning. However, none of these algorithms has included nodes that explicitly represent the visibility of each reference image pixel in each of the other views. In the case of NVS, the values of these variables are necessary for computing the colour of each output pixel, so it is preferable to have visibility nodes in the graph. Indeed, having visibility nodes actually simplifies the graph construction of [WQ05], as figure 7.2 shows, removing the need for higher order cliques (and the resulting approximations) in the construction. This section describes the construction introduced here, first in the context of a stereo problem.

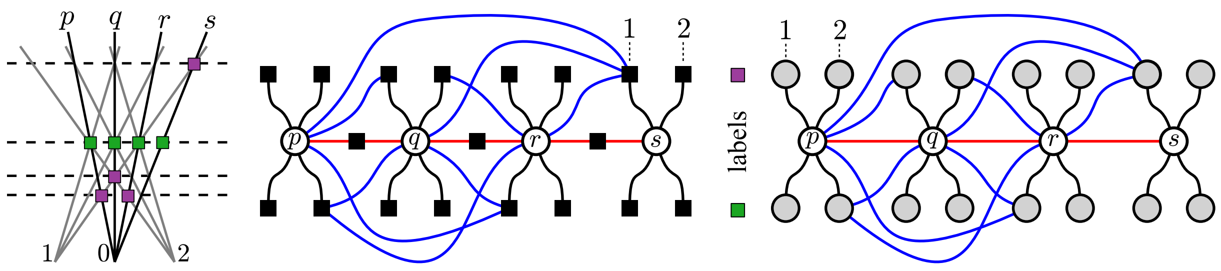
When solving the binary optimization problem of equation 7.12, one can precompute all the possible occlusions between all the pixels. For a given pixel, $\mathbf{x}$, let $\mathcal{L}_{ab}^{\mathbf{x}}$ be the list of pixels which occlude pixel $\mathbf{x}$ at the disparity indicated by $B(\mathbf{x}) = a$ when their own disparities are indicated by $B(\mathcal{L}_{ab}^{\mathbf{x}}) = b$. The data cost for pixel $\mathbf{x}$, given by equation 7.4, is therefore only a function of the values of B for the variables $\{\mathbf{x}, \mathcal{L}_{00}^{\mathbf{x}}, \mathcal{L}_{01}^{\mathbf{x}}, \mathcal{L}_{10}^{\mathbf{x}}, \mathcal{L}_{11}^{\mathbf{x}}\}$; unfortunately this forms a clique of arbitrary size. Wei & Quan [WQ05] notice that, by separating the data cost into two cliques, one for $B(\mathbf{x}) = 1$ and one for $B(\mathbf{x}) = 0$, the cliques become smaller, though still arbitrary in size, containing only the variables $\{\mathbf{x}, \mathcal{L}_{00}^{\mathbf{x}}, \mathcal{L}_{01}^{\mathbf{x}}\}$ and $\{\mathbf{x}, \mathcal{L}_{10}^{\mathbf{x}}, \mathcal{L}_{11}^{\mathbf{x}}\}$ respectively. Figure 7.2(b) gives an example of a four node clique generated in this way, though [WQ05] actually make approximations to avoid cliques larger than size three. The cost for one of these cliques is given by

$$\phi_{\mathbf{x}i}^{1}(B(\{\mathbf{x}, \mathcal{L}_{10}^{\mathbf{x}}, \mathcal{L}_{11}^{\mathbf{x}}\})) = \begin{cases} 0 & \text{if } B(\mathbf{x}) = 0 \\ \nu & \text{else if any } B(\mathcal{L}_{10}^{\mathbf{x}}) = 0 \\ & \text{or any } B(\mathcal{L}_{11}^{\mathbf{x}}) = 1 \\ \rho_{\text{d}}\left(I_i\left(\pi_i\left(\mathbf{x}, D_t(\mathbf{x})\right)\right) - I_0^*(\mathbf{x})\right) & \text{otherwise} \end{cases} \tag{7.13}$$

What is apparent from the equation above is that there are only three possible costs, indicating that a pairwise clique (which can model four costs) should be sufficient to model them. This can be achieved by introducing a visibility node, $V_i^a(\mathbf{x})$, representing the visibility of pixel $\mathbf{x}$ at the disparity given by $B(\mathbf{x}) = a$ projected into the i^{th} input view, and replacing the middle condition with "else if $V_i^a(\mathbf{x}) = 0$", so that the costs above can be modelled by the black edges in figure 7.2(c).

The visibility node will automatically have value 1 (visible), as this offers the lowest cost. If a pixel in $\{\mathcal{L}_{a0}^{\mathbf{x}}, \mathcal{L}_{a1}^{\mathbf{x}}\}$ occludes it, let's say pixel $\mathbf{p}$ in $\mathcal{L}_{ab}^{\mathbf{x}}$, then there must be an edge connecting nodes $\mathbf{p}$ and $V_i^a(\mathbf{x})$ which has a high ($> \nu$) cost if $V_i^a(\mathbf{x}) = 1$, $B(\mathbf{p}) = b$ and zero cost for other permutations. To avoid this high cost, $V_i^a(\mathbf{x}) = 0$ when $\mathbf{p}$ (or any other pixel in $\{\mathcal{L}_{a0}^{\mathbf{x}}, \mathcal{L}_{a1}^{\mathbf{x}}\}$) occludes the corresponding input sample, ensuring the occlusion cost, ν, is paid in the data cost edge instead. These visibility costs, represented by the blue lines in figure 7.2(c), are therefore never paid, but enforce the occlusion constraint described in §7.1.1. This construction is very similar to the large clique Potts model construction of [KKT07].

This graph construction can generate non-submodular edges. If the pixel $\mathbf{p}$ (from above) is in $\mathcal{L}_{a1}^{\mathbf{x}}$,



(a) A simple stereo problem (b) Construction of [WQ05] (c) Construction introduced here

Figure 7.2: Stereo graph construction. (a) A simple, 2-d, binary label, stereo problem in which the reference view (labelled 0) has four pixels (p–s), generating the four rays emanating from its optical centre, and there are two other input views (1 & 2) with three pixels each. The current disparity map for the reference image, D_t, is indicated by purple squares, while the proposal disparity map being fused, D^p, which is fronto-parallel, is indicated by green squares. (b) A factor graph representation of the graph that Wei & Quan [WQ05] would construct for the problem in (a), but for their approximations. The nodes (circles) represent the binary label variables of the reference image pixels. Red lines connect nodes to geometry regularizing factors, while black and blue lines, which represent data and occlusion interactions respectively, connect nodes to data cost factors, one for each reference image pixel at each disparity (top row for purple labels; bottom row, green) projected into each of the two other views (labelled 1 & 2 for one pair of factors). (c) The construction introduced here for the problem in (a), in which the binary visibility variable for each data cost is explicitly represented by a node (grey circles). Lines represent pairwise factors (cliques), the colours indicating the purpose of each clique as before.

then only the state $V_i^a(\mathbf{x}) = 1$, $B(\mathbf{p}) = 1$ has a high cost, breaking the submodularity constraint of equation 3.2. If the values associated with either of the two labels are swapped (*e.g.* $V_i^a(\mathbf{x}) = 1$ is made to mean occluded) to make the edge submodular, then edges generated by pixels in $\mathcal{L}_{a1}^{\mathbf{x}}$ immediately become non-submodular. The construction of [WQ05] suffers from the same problem; they approximate the energy to remove the non-submodularities. The approach used here is to leave the energy unchanged, but employ QPBO instead of graph cuts to deal with the non-submodularity.

7.2.1.2 Incorporating variable colour

While the construction above works well for stereo problems, the NVS problem is somewhat more complex—both the data costs of equation 7.3 and the smoothness costs of equation 7.6 are a function of colour, which itself is a function of visibility. One approach to dealing with this, similar to that used by Strecha *et al.* [SFVG04], is to use a two-step approach to the binary optimization of equation 7.12, first optimizing disparity (and visibility) with the current colour, then updating colour using equation 7.10. While similar to EM, the colour computation step can actually increase the energy, and the approach suffers from the local optimality of EM. A preferable approach is to incorporate the variability of colour directly into the QPBO graph construction, allowing a more optimal solution to be found at each iteration. However, in taking this approach, all the good work in removing higher order cliques from the stereo construction is negated by the reappearance of higher order cliques in the NVS construction, as shown in figure 7.3(a)—data costs generate cliques of size $1 + N$, while the smoothness prior generates cliques of size $2 + 4N$.

The solution developed here is to use a hybrid of the two-step and single graph approaches—where possible, the variability of colour is modelled in the graph, and where the cliques are too large, the colour of each pixel at each disparity is fixed to the value computed using visibilities generated in the previous iteration (computed from D_t). Colours for all pixels are then updated at the end of the iteration, based on the new visibilities (computed from D_{t+1}). A "too large" clique is deemed to be one that cannot generally (without placing constraints on the clique energies) be converted to a set of pairwise edges. Freedman *et al.* [FD05] suggest that this is the case for cliques above size three, but Kolmogorov & Zabih [KZ04] give a decomposition which holds for any triple clique. They claim only that the decomposition works for submodular triple cliques[2], so I show in figure 7.4 that the decomposition holds for all cliques, albeit generating non-submodular pairwise cliques in some cases, but, as QPBO is being used in the optimization, this is not of primary concern.

Limiting cliques to size three means that all data costs can be modelled exactly for $N = 2$. However, if a visibility node has no occlusion constraint edges attached to it then its value cannot vary and it can be removed from the graph, reducing the size of certain data cost cliques and hence allowing many to be modelled exactly for $N > 2$, as shown in figure 7.3(b).

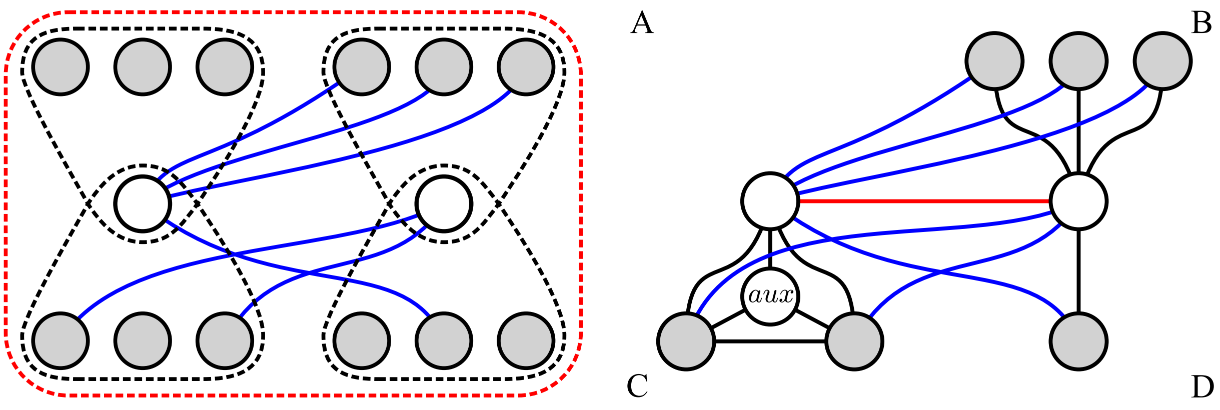
Let $\phi^a_{\mathbf{x}ij}$ be the clique functional representing the data cost for $B(\mathbf{x}) = a$, where $V^a_i(\mathbf{x})$ and $V^a_j(\mathbf{x})$ both have the potential to vary. The entries of the energy table are given by

$$\phi^a_{\mathbf{x}ij}(B(\mathbf{x}), V^a_i(\mathbf{x}), V^a_j(\mathbf{x})) = \begin{cases} 0 & \text{if } B(\mathbf{x}) = a \\ \sum_{k=1}^{N} f(I_k(\pi_k(\mathbf{x}, D_t(\mathbf{x}))) - I^*_0(\mathbf{x}), V^a_k(\mathbf{x})) & \text{otherwise} \end{cases} \tag{7.14}$$

where $I^*_0(\mathbf{x})$ is computed separately for each combination of visibilities using equation 7.10. The pairwise cliques resulting from a single visibility having the potential to vary is constructed similarly.

It should be noted that the submodularity of these data cost cliques is not guaranteed, and also that it is possible for an entry in the table representing one or two occlusions to generate a lower energy than

[2] A submodular triple clique is one which can be decomposed into a set submodular pairwise cliques.



(a) Exact construction (b) Approximate soluble solution

Figure 7.3: NVS graph construction. Graph constructions similar to those in figure 7.2, but for a two pixel new view with $N = 3$. (a) The higher order clique construction required to exactly solve equation 7.12. Dashed lines represent higher order cliques which contain all the nodes encircled. The smoothness term (red clique) depends on both the colour and disparity of each pixel in the neighbourhood, so is a function of all visibility and pixel disparity nodes (*i.e.* size $2 + 4N$). The data costs (black cliques) depend on the disparity of a particular node and the colour at the disparity being modelled, which is a function of all visibility nodes of that pixel at that disparity (*i.e.* size $1 + N$). (b) The soluble approximation to (a), containing only pairwise cliques. Surface smoothness costs use a fixed approximation of pixel colour in equation 7.9, becoming a pairwise clique as in stereo. All visibility nodes with no occlusion interactions can be removed as the image samples associated with those nodes will always be visible, reducing some of the data cost cliques in size. Cliques of size 1, 2 and 3 are modelled exactly using the graph structures in corners A, D and C respectively. In particular, the triple clique energy is decomposed into 6 pairwise terms as described in [KZ04], which also generates an additional, latent node, aux. Cliques of size 4 (corner B) or larger use a fixed approximation of pixel colour, generating pairwise edges as in stereo.

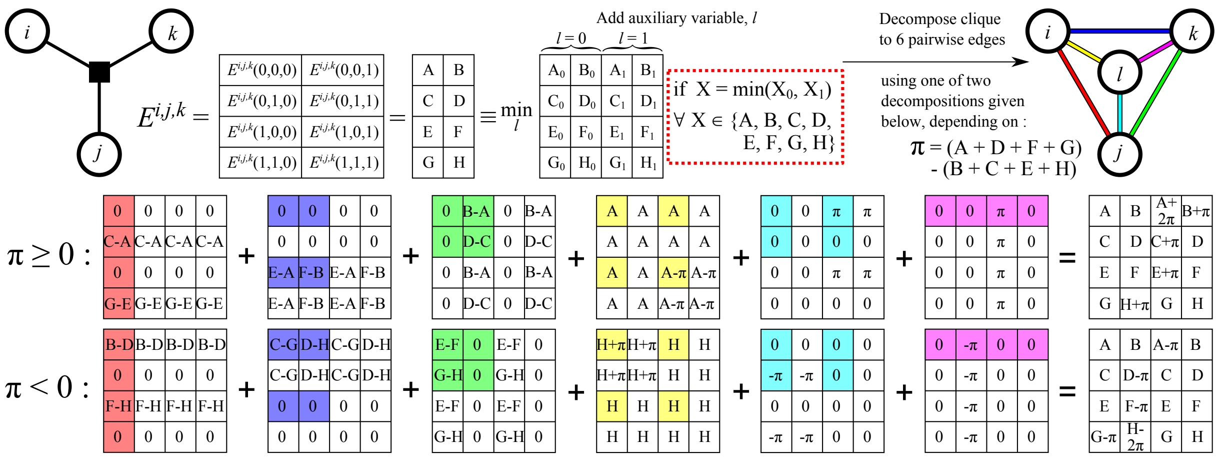

Add auxiliary variable, l
Decompose clique to 6 pairwise edges
using one of two decompositions given below, depending on :
π = (A + D + F + G) - (B + C + E + H)
if X = min(X₀, X₁) ∀ X ∈ {A, B, C, D, E, F, G, H}
π ≥ 0 :
π < 0 :

Figure 7.4: Triple clique decomposition. Kolmogorov & Zabih [KZ04] show how a submodular triple clique can be decomposed into 6 pairwise edges, with the addition of an extra, latent graph node. This figure shows that their decomposition holds for non-submodular triple cliques also, by showing that the resulting 4 variable energy tables for the two values of π satisfy the required constraint (in the red dashed box) *independent* of the form of the input triple clique energy table.

the entry representing no occlusions. In the latter case, it is possible for a visibility node to be set to 0 when the associated image sample is not genuinely occluded, leading to the objective energy not being correctly modelled. In the case that a visibility node, $V_i^a(\mathbf{x})$, has only one interacting pixel, say $\mathbf{p}$, the data cost clique can be connected directly to the node $B(\mathbf{p})$ instead of $V_i^a(\mathbf{x})$, thus avoiding the problem. However, in the case that a visibility node has two or more interacting pixels the issue remains.

If, after the removal of unnecessary visibility nodes, some data costs generate cliques larger than size three then the colour of I_0^* for the relevant pixels is fixed at its current value (based on visibilities computed from D_t) and the data costs modelled using pairwise edges as per the stereo problem, with energies given by equation 8.10. These cliques are submodular, and the attached visibility nodes cannot be set to 0 without a genuine occlusion occurring.

Smoothness cliques are assumed to always be larger than size three, so the current colour approximation is automatically used in equation 7.9, generating a single pairwise edge for each prior neighbourhood. However, the smoothness modulating term of equation 7.9 has no guarantees on submodularity, therefore these smoothness edges are not guaranteed to be submodular either.

7.2.2 Label fixing

The sections above describe the QPBO graph constructed to solve the binary optimization problem of equation 7.12. It was noted that occlusion constraints, data cost triple cliques and even smoothness constraints may generate non-submodular edges, as a result of which the binary labelling, B, output by QPBO may contain some *unlabelled* nodes. These unlabelled nodes must be set to 0 or 1 in a postprocessing step in order to generate D_{t+1} using equation 7.11, and furthermore, in order to ensure convergence of the algorithm, the labels should be chosen to ensure that $E(\mathsf{D}_{t+1}|\mathsf{I}_1,..,\mathsf{I}_N) \leq E(\mathsf{D}_t|\mathsf{I}_1,..,\mathsf{I}_N)$. This section discusses the ways in which this label fixing can be done. Note that, while there are many more nodes (*e.g.* visibility nodes) involved in the QPBO optimization, the aim is only to find the values of B, *i.e.* the labels for the nodes corresponding to pixel disparities, so these are the only nodes whose labels are fixed. When computing the energy of a labelling, the values of the visibility variables can be computed directly from the disparity using equation 7.2.

There are several approaches to fixing labels that have already been proposed in the literature.

QPBO-F *Fix to current* [WRTF07]: fix unlabelled nodes to 1, the current best labelling.

QPBO-L *Lowest energy label* [LRB07]: fix unlabelled nodes collectively to whichever of 0 or 1 gives the lowest energy.

QPBOP *Probe*: probe the graph, as described in [BHT06, RKLS07], in order to find the labels of more nodes, that form part of an optimal solution.

QPBOI-F *Fix to current and improve*: fix unlabelled nodes to 1, and transform this labelling using QPBOI [RKLS07].

I introduce two new approaches to label fixing which are based on the optimal splice technique of §5.2.2.2. That technique split the two labellings into independent regions, and independently selected the label, 0 or 1, which gave the lowest energy for each region. The unlabelled nodes of B can similarly be split into independent regions, growing each region from a seed unlabelled node by adding all nodes that share a clique with the seed node, then repeating the process for all new unlabelled nodes in the region, and so on. Given an ordered list of cliques containing ordered node indices this process can be achieved in $\mathcal{O}(|\mathsf{B}|)$ (*i.e.* linear) time. A looser constraint than regions being independent is regions being *strongly connected*. Nodes in two different strongly connected regions (SCRs) can share a clique, but the dependence between the two regions can only be unidirectional; in practice (in this application) SCRs are almost always independent. This is relevant because the SCRs can be computed in $\mathcal{O}(|\mathsf{B}|)$ time [BJ89] *without* the ordering preprocess, making them an efficient approximation to independent regions. The two new approaches to fixing labels are therefore:

QPBO-R *Lowest cost label per region*: split unlabelled nodes into SCRs, as per [BJ89]. For each SCR, independently select the labelling, 0 or 1, which gives the lowest total energy for cliques connected to that region.

QPBOI-R *Improve lowest cost label per region*: Label nodes as per QPBO-R, then use QPBOI to transform this labelling.

All the described methods ensure convergence, because they are all guaranteed[3] to have an equal or lower energy than the output of QPBO-F, which is itself guaranteed not to increase the energy as a result of the autarky property described in §3.1.2.

[3]In fact the approximation of independent regions with SCRs used here can theoretically lead to an increase in energy, but only a small one, and this has proved extremely rare in practice.

7.2.3 Implementation details

The disparity proposals, D^{p}, for each fusion move are fronto-parallel planes, as per [WQ05]. Doing so not only keeps the smoothness costs (ignoring the modulation term) submodular [BJ01], but also reduces the number of non-submodular visibility constraints [WQ05]. The disparities of these fronto-parallel planes are discretized as described in §4.2.1.3. Each cycle of the optimization algorithm fuses each of the N_{L} proposals to the current solution once, in order, from front to back ($d_{\max}$ to $d_{\min}$). The reason for the ordering is that it improves the quality of the fixed colour approximation introduced in §7.2.1.2 by allowing occluding pixels to be fixed first so that the visibility variables are more accurate for the occluded pixels which follow.

The algorithm runs through three complete cycles over disparity only, rather than stopping at convergence. The reason for this is that convergence is not guaranteed as updating colour can increase the energy. Three cycles are chosen simply because most details are fixed at this point, and to continue would provide diminishing returns. In addition, the first cycle through disparity does not use any visibility reasoning in computing E_{photo}, *i.e.* $V_i(\mathbf{x})$ is assumed to be 1 in equation 7.3, though it is used to update colour. The visibility reasoning adds considerable complexity to the graphs solved in equation 7.12, increasing the number of unlabelled nodes returned by QPBO, leading to poor minima being reached. Initially removing the visibility reasoning allows a reasonable intermediate solution to be generated, from which to start optimization of the true objective function.

The various parameters of the energy function given in figure 7.1 are made to be functions of the sequence-dependent variables N, c (number of colour channels) and d (the difference in disparity between consecutive disparity labels), with a view to making them invariant to these values. The constant values in these functions, which are fixed for all experiments, were chosen after a coarse grid search over parameter space and qualitative inspection of the results, giving the parameter values in table 7.1.

I use Kolmogorov's [Kol07] implementations of QPBO, QPBOP and QPBOI, which use a max-flow algorithm described in [BK04]. Both QPBOP and QPBOI methods make use of tree-recycling [KT05] for a fast implementation; the number of graph solves is at most linear in the number of unlabelled nodes for QPBOI, but exponential for QPBOP, though it should be noted that QPBOP labels nodes optimally, rather than approximately, as with QPBOI.

Parameter	κ	ν	δ_{s}	λ_{d}	δ_{t}	λ_{t}
Value	$c(12.5N/(N-1))^2$	$\kappa+1$	$1.9d$	$10\delta_{\text{s}}/\kappa N$	$5000c$	$6/\delta_{\text{t}}$

Table 7.1: Parameter settings. Values of the constant parameters in the objective function, where c is the number of colour channels in the input sequence, and d is the constant disparity spacing between the discrete proposal depths, which varies between input sequences.

		QPBO-F	QPBO-L	QPBOI-F	QPBOP	QPBO-R	QPBOI-R
Monkey	Time (s)	391	363	373	523	364	374
	Energy	6.3	2.9	3.1	0	1.3	0.4
Plant	Time (s)	1407	1368	1288	5115	1326	1236
	Energy	5.0	5.5	2.0	0	4.8	2.9
Dino1	Time (s)	232	233	234	1464	232	246
	Energy	2.0	1.3	0.4	0	0.9	0.076
Dino2	Time (s)	562	569	571	n/a	576	601
	Energy	1.4	1.7	0	n/a	0.80	1.0

Table 7.2: Label fixing methods. Quantitative results for the label fixing methods described in §7.2.2 on the four test sequences. The methods were evaluated on the time taken to generate an output image, and the energy of the final solution, given as a percentage increase above the lowest energy for each particular sequence. Results for QPBOP on the Dino2 sequence are not available as the computation takes too long.

7.3 Experiments

This section examines the impact of the contributions made in this chapter.

7.3.1 Label fixing methods

Each of the label fixing methods described in §7.2.2 was used to render each of the four test sequences. Table 7.2 gives the quantitative performance of these algorithms, in terms of the time taken[4] and the energy achieved. From the results it is clear that QPBOP takes the longest time by some considerable margin—one of the runs did not even finish—but also generates the lowest energy. This is to be expected, as it finds the globally optimal solution of each binary problem, but requires a number of re-solves of the graph exponential in the number of unlabelled nodes. However, it should be noted that generating a lower energy in each binary optimization does not guarantee a lower final energy; a lower energy intermediate solution can lead to a higher energy local minimum of the multi-label problem. For example, for Dino2, QPBO-F generates a lower energy than QPBO-L, despite the latter being guaranteed to find an equal or lower energy than the former on a given binary problem.

[4]The time includes that time taken to sample input images, construct the binary graphs and solve them using QPBO, as well as the post-processing label fixing step.

Apart from QPBOP, the other algorithms take approximately the same time to run. In terms of energy minimizing performance, QPBO-F and QPBO-L perform worst on two sequences each, while QPBOI-F and QPBOI-R perform best on two sequences each. However, QPBOI-R wins by a significant margin on the Monkey sequence, and the "Improve" step of each binary optimization is guaranteed to start from an equal or lower energy user-defined labelling than QPBOI-F. Given the trade-off between speed and efficacy, QPBOI-R is chosen as the optimizer for all results shown.

7.3.2 Qualitative performance

Let us now look at the qualitative results generated using this framework, focussing on some particular areas of reconstruction.

7.3.2.1 Large scale errors

Making use of a global optimizer, this framework is able to correct the "usual suspect" large scale errors of the test sequences. Both the 'ribs' of the leaves, and the stalk are both correctly rendered in the Plant sequence, as shown in figure 7.6. Figure 7.8 presents the output for the Dino1 and Dino2 sequences, showing the cabinet frame of Dino1 (a), and also the bridge of the Edmontosaurus's snout (c) (zoomed in figure 7.13(C)) are both correctly rendered also, for the most part—the snout does contain some nicks.

7.3.2.2 Occluded regions

In partially occluded regions such as below the feathers in figure 7.6(Zoom 2) and under the monkey's arm in figure 7.5(Zoom 1), the occluded texture is reconstructed faithfully.

7.3.2.3 Fine details

A failure mode of this framework is the difficulty it has in aligning discontinuity boundaries in the output disparity maps with the true disparity boundary around fine structural features. In this situation artifacts, in the form of obviously incorrect edges, are generated, which only occur when using geometry regularization. Examples of this include the monkey's fur, as shown in figure 7.5(Zooms 1 & 2), and a vertebra in Dino1, as shown figure 7.9(a & b). The issue is caused by the smoothness prior's preference for shorter discontinuity boundaries, which the texture term of equation 7.9 was incorporated into the energy to overcome, so it may be that more optimal parameters could reduce the effect.

7.3.2.4 Textured regions

Regularizing geometry allows areas of texture that were poorly reconstructed by the image priors of previous chapters to be correctly reconstructed. By assigning all the brickwork of Dino2 the same disparity, the mortar is rendered correctly, as shown in figure 7.9(c & d), while the high frequency detail of a portion of the baize background of the Plant sequence is revealed by correctly assigning disparity to the region, as shown in figure 7.13. Some detail of the fur on the monkey's face (figure 7.5) is also revealed.

7.3.3 Incorporating visibility

In this chapter the decision was made to include a geometrical occlusion model in order to penalize occlusions. The effect of this design choice is illustrated in figure 7.10, which shows the results on the Plant sequence with and without the occlusion model, *i.e.* $V_i(\mathbf{x}) = 1$ in equation 7.3 in the latter case, though colour is still computed using the true value of $V_i(\mathbf{x})$. Artifacts are highlighted in the result generated without visibility reasoning (*top left*), which do not appear when visibility reasoning is included. An inspection of the disparities, D, for the two results indicates that the artifacts result from errors in the disparity map. These disparity errors generate more occlusions (*bottom row*)—by discouraging occlusions, the visibility model reduces this kind of artifact.

However, in certain situations the wrong disparity will be chosen in order to avoid occlusions when using visibility reasoning. An example of this is the reconstruction of the baize background of the Plant sequence; it is correctly reconstructed, for the most part, without visibility reasoning, but with visibility reasoning the cost of occlusions that would be generated by the stalk to the lower left of the image causes a large section of the background to be given the wrong disparity, and hence smooth over the low contrast texture of the baize.

7.3.4 Incorporating variable colour

A contribution of the work presented here has been to develop a means of incorporating the variability of pixel's colour, depending, as it does, not only on its own disparity, but also the disparity of other pixels which may occlude it, into the binary graph solved in each fusion iteration, as described in §7.2.1.2. To ascertain its impact I compare results with and without variable colour included in the graph, *i.e.* data costs are based on the current $\mathbf{I}_0^*$ in the latter case.

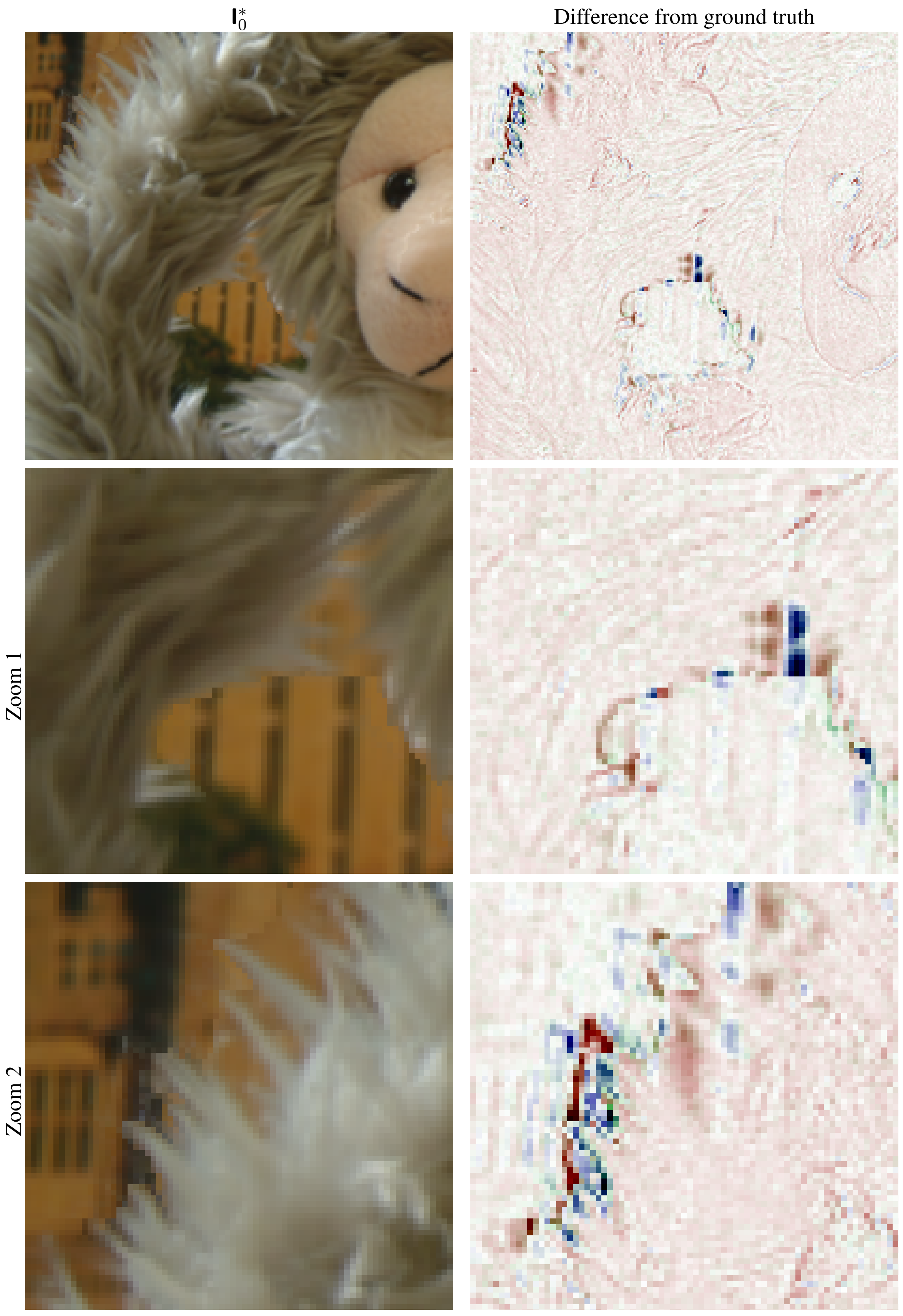


Figure 7.5: Monkey results. Output image for the Monkey sequence, with zooms to highlight areas of interest. The image has an r.m.s. error of 4.14, and 0.35% gross errors.

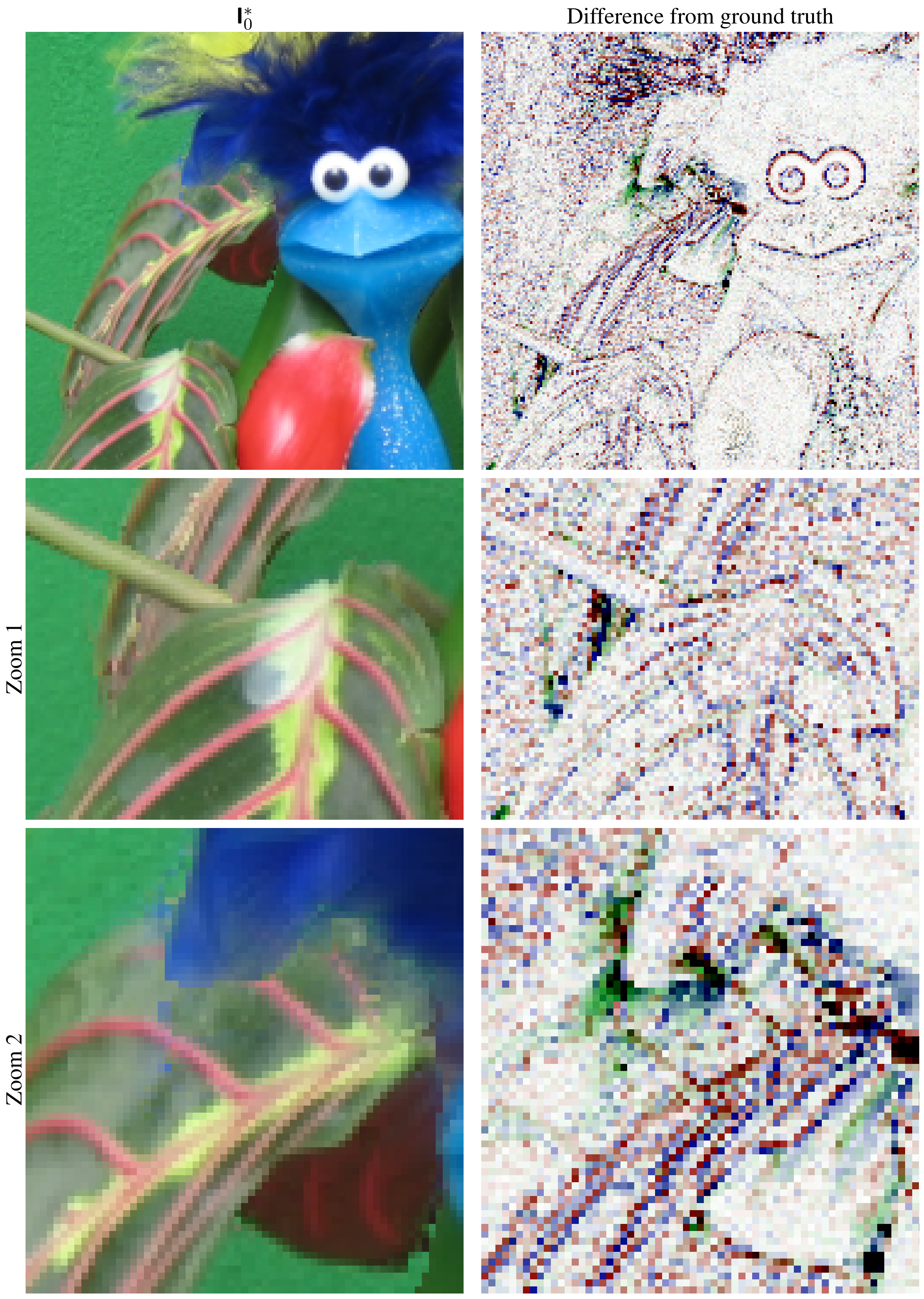


Figure 7.6: Plant results. Image and disparity output for the Plant sequence, with zooms to highlight areas of interest. The image has an r.m.s. error of 9.77, and 6.47% gross errors.

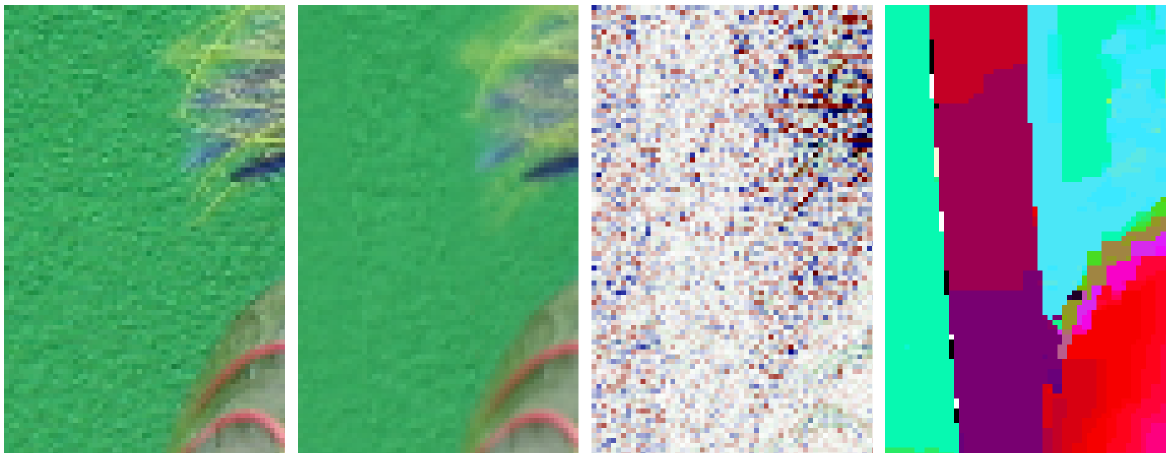

(a) Ground truth (b) I_0^* (c) Diff. (a) & (b) (d) D

Figure 7.7: Texture reconstruction. A zoom of a section of baize in the Plant sequence, showing (a) the ground truth texture, (b) the reconstructed texture and (c) its difference from the ground truth, and finally (d) the reconstructed disparity of the zoomed region.

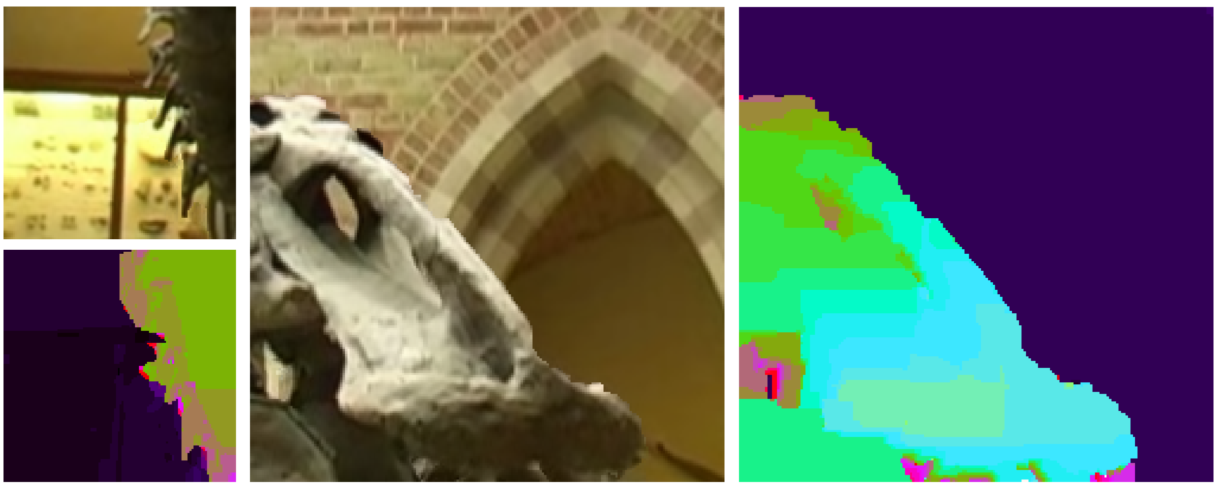


Figure 7.8: Edmontosaurus results. Output images, I_0^* (a & c) and disparities, D (b & d) for the Dino1 and Dino2 sequences respectively.

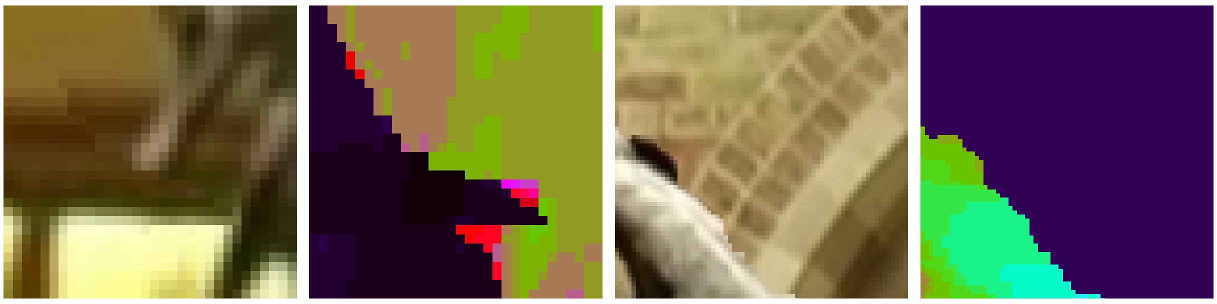


Figure 7.9: Edmontosaurus zooms. Zooms of Dino1 I_0^* (a) and D (b), and Dino2 I_0^* (c) and D (d).

Figure 7.11 demonstrates the qualitative effect of incorporating variable colour on the Monkey sequence with both two and four input images ($N = 2$ & $N = 4$ respectively). In the case of $N = 2$, using the fixed colour approach (*top left*) generates many artifacts, while incorporating variable colour (*top right*) generates a much improved image. However, by the time $N = 4$ (*bottom row*), the additional complexity of the variable colour construction has a much less significant impact on image quality.

Figure 7.12 shows the quantitative effect of incorporating variable colour, in the form of graphs of energy and no. unlabelled pixels over time (in terms of iteration number), for the Monkey sequence images shown in figure 7.11. The first loop through the discrete set of disparities does not incorporate visibility constraints, therefore also does not incorporate variable colour, so the first N_{L} iterations[5] generate the same result for both fixed and variable colour constructions. Interestingly, even though the graphs contain some non-submodular edges at this stage, they do not generate any unlabelled nodes (figure 7.12(b & d)). However, the energy can increase between iterations, as seen in figure 7.12(a & c), as $\mathbf{I}_0^*$ is updated after each fusion.

After N_{L} iterations, E_{photo} is computed with visibility reasoning, causing the total energy to increase. From this point on, for $N = 2$ (figure 7.12(a)) the energy decreases at a much faster rate for the variable colour construction, despite it generating many more unlabelled nodes (figure 7.12(b)), while for $N = 4$ the energy and no. unlabelled nodes (figure 7.12(c & d)) are much more similar for the two constructions, with the variable colour construction generating a slightly lower energy.

As the fixed colour approach updates disparity and colour alternately, it is prone to fall into local minima where the current disparity matches the current colour, but the proposed disparity does not. The variable colour approach, by updating disparity and colour simultaneously, does not get stuck in these minima. It is known in stereo that there are fewer local minima of photoconsistency over disparity the more input images there are [OK93], which explains why this impact of the variable colour construction is more prominent for lower N.

7.3.5 Incorporating texture regularization

Equation 7.9 incorporates the pairwise image prior introduced in the previous chapter, in order to modulate E_{smooth}. Figure 7.13 demonstrates the effect of this design choice by comparing results generated with this modulating term (C), with those generated using no modulation (a, A), and the CRF weight (b,

[5]The number of discrete disparities can change with N, as the maximum distance between samples in each new input image is different for a fixed N_{L}.

B) described in [KZ02]. Without modulation, discontinuity edges arrange themselves in order to minimize photoconsistency, with no concern as to the visual quality of the edge itself. This generates very obvious and unnaturally shaped edges, as seen in figure 7.13(A). Incorporating a modulation term commonly used in stereo does not improve matters, because it causes discontinuities to align themselves with the very edges caused by the incorrect discontinuity in the first place. However, the “Sampled” image prior is able to distinguish between likely and unlikely edges, allowing the reconstruction to correctly set the discontinuity boundary at the edge of the Edmontosaurus’ snout, as shown in figure 7.13(C) and figure 7.8(d).

7.4 Conclusion

This chapter has confirmed the common suggestion that graph-cut stereo methods can be applied to the task of new-view synthesis. While straightforward in principle, this repurposing presents a number of technical difficulties, the solutions to which are the main contributions of this chapter:

- It has shown how the asymmetrical occlusion model of [WQ05] can be constructed without approximations, albeit generating non-submodular cliques.
- It has shown that the triple clique decomposition of [KZ04] is valid for non-submodular cliques.
- It has shown how colour and disparity can be optimized simultaneously for NVS within the fusion move framework.
- It has shown that the “Sampled” pairwise texture prior can be used to regularize disparity discontinuity boundaries in place of the CRF terms used in stereo, which are not available in NVS.
- It has introduced two new label fixing approaches for post-processing the output of QPBO.

The results demonstrate that an explicit disparity model with global, geometric occlusion reasoning can correctly reconstruct texture in both unoccluded and partially occluded regions, and validate the design choices and contributions made in this chapter.

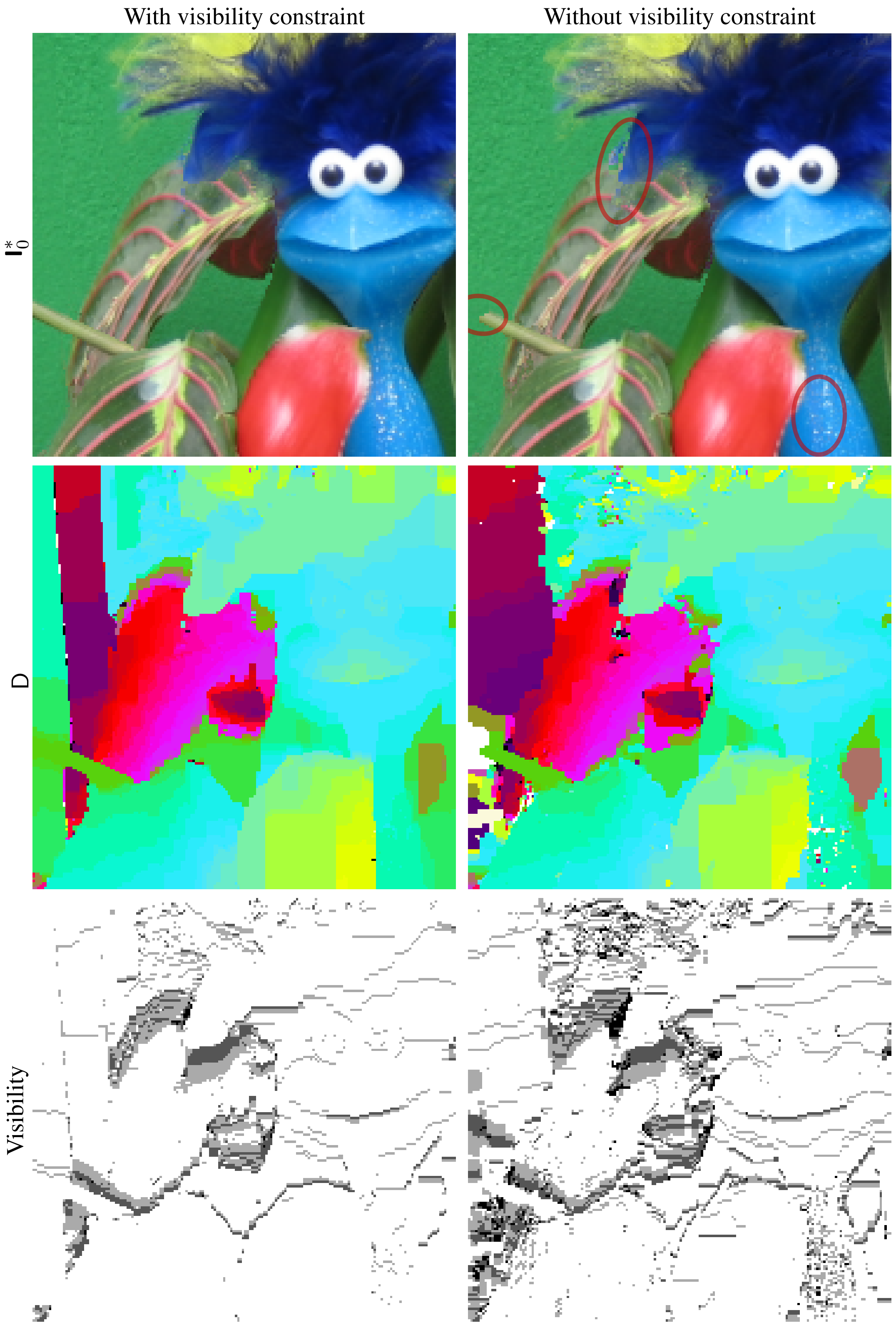


Figure 7.10: Effect of the visibility constraint. *First column*: The output image, $\mathbf{I}_0^*$, disparity map, D, and a map of the number of visible input samples per pixel (darker means fewer visible samples) for the Plant sequence, with the visibility constraint. *Second column*: Equivalent results without the visibility constraint, *i.e.* all samples are assumed to be visible when computing E_{photo}. Artifacts are highlighted in the output image (top right). Note that the colour update of equation 7.10 is still computed using visibilities.

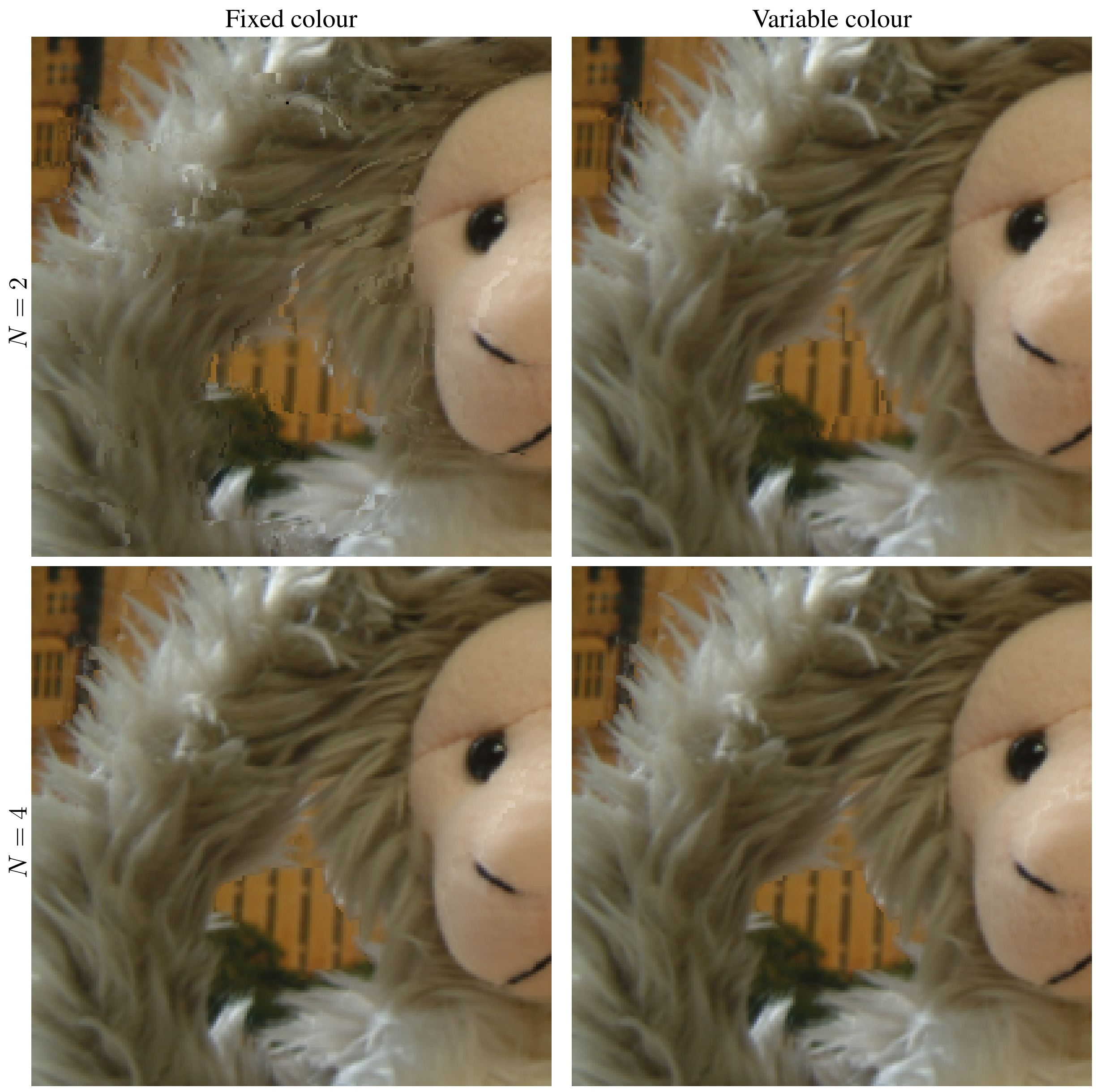
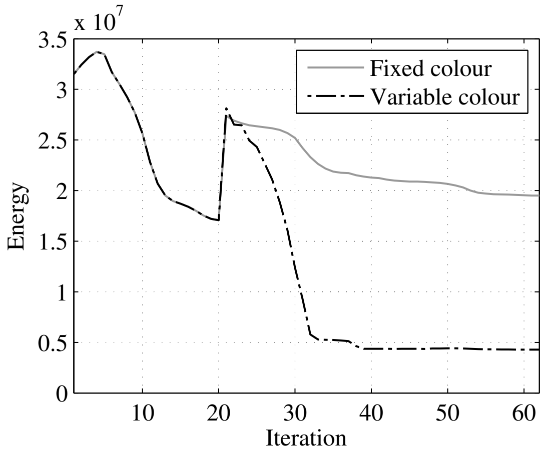
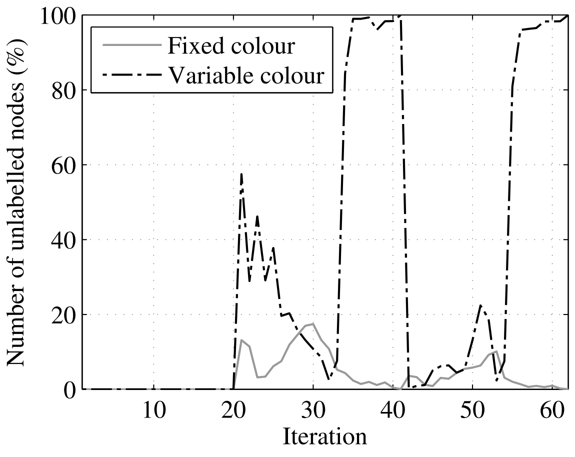


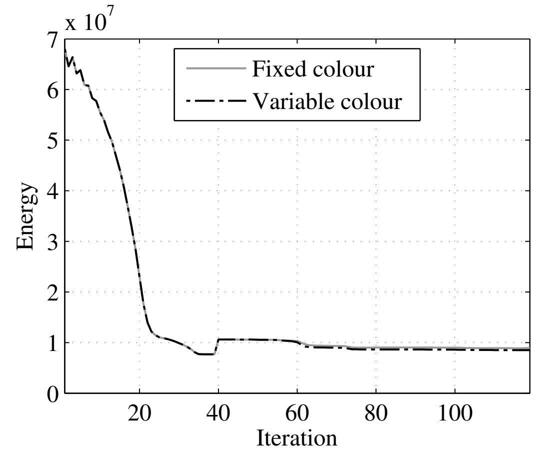
Figure 7.11: Incorporating variable colour. The columns show the effects of using fixed colour and variable colour (as described in §7.2.1.2) on $\mathbf{I}_0^*$ for the Monkey sequence. Rows show results for $N = 2$ (*top*) and $N = 4$ (*bottom*).



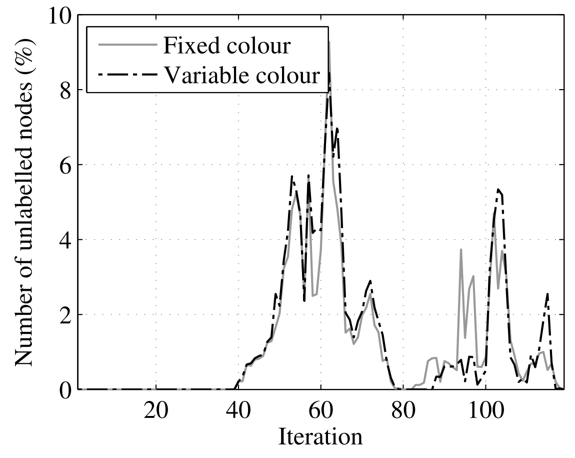
(a) Energy *vs*. iteration, $N = 2$



(b) No. unlabelled *vs*. iteration, $N = 2$

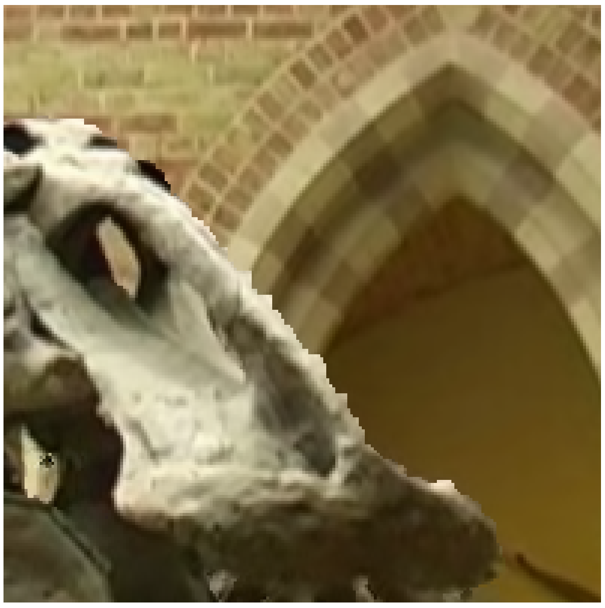


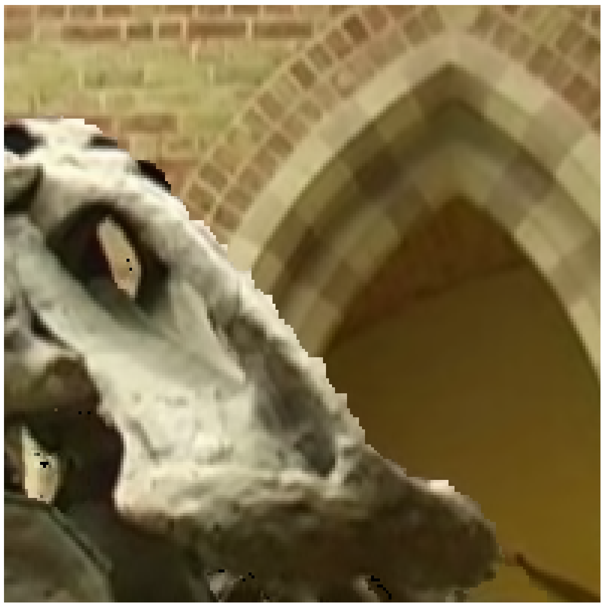
(c) Energy *vs*. iteration, $N = 4$

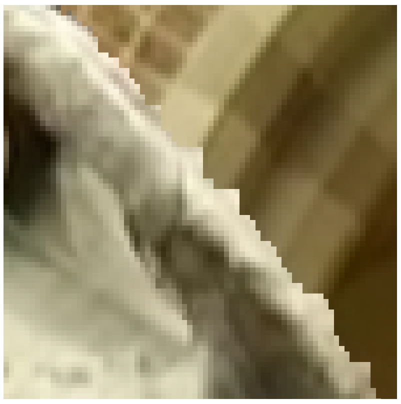


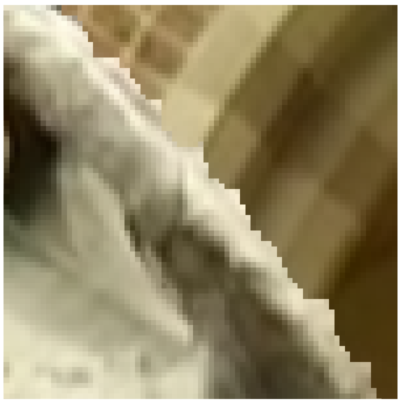
(d) No. unlabelled *vs*. iteration, $N = 4$

Figure 7.12: Quantitative effect of incorporating variable colour. Graphs (a) & (b) show the effects of using fixed colour and variable colour (as described in §7.2.1.2) on energy and no. unlabelled pixels per iteration respectively, for the Monkey sequence with $N = 2$ ($N_{\mathrm{L}} = 20$). Graphs (c) & (d) show the equivalent results for $N = 4$ ($N_{\mathrm{L}} = 39$). The fixed and variable colour methods are identical for the first N_{L} iterations, as visibility reasoning, and therefore variable colour, is not used in the optimizations of these iterations, as discussed in §7.2.3.


(a) $\delta_t = 0$


(b) CRF weight of [KZ02]


A


B

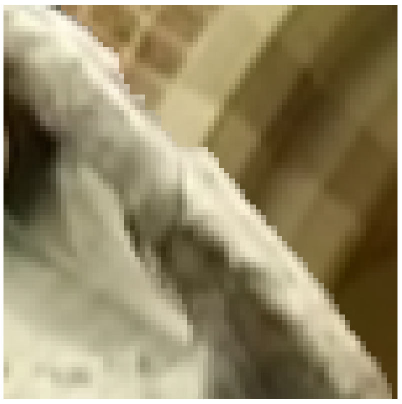

C

Figure 7.13: Effect of texture term. This figure shows the effect of changing the modulation of the smoothness term, given by equation 7.9, on $\mathbf{I}_0^*$ of the Dino2 sequence. (a) The output using $\delta_t = 0$, *i.e.* no modulation. (b) The output replacing equation 7.9 with the CRF weight from [KZ02]. (A)–(C) Zooms from (a), (b) and figure 7.8(b) respectively.

Chapter 8

Second order smoothness priors

This chapter looks at priors for forward transfer methods of new view synthesis (NVS)—those methods which infer the geometry of the *input* views, and use this geometry to project the input views directly into the output view. The problem posed here is therefore one of inferring the geometry of known views, *i.e.* the dense stereo problem, and the priors assessed will be those that regularize geometry. Projecting the depth-mapped input images into new views is also demonstrated.

The previous chapter investigated regularizing geometry in a graph-cuts-based framework with visibility reasoning, for a new view. In doing so it introduced several key tools than can be of use in the original stereo problem also, namely the construction of triple cliques and an occlusion model consisting only of pairwise cliques. This chapter uses this same framework for the stereo problem, but extends the smoothness prior to a second order prior, which permits planar surfaces with zero cost. Such priors were initially popular in the stereo literature, but quickly lost favour to the computationally more tractable first order prior. The main contributions of this chapter are to show that such second order priors are not only tractable, but generate better results.

The chapter is organized as follows: the objective function is defined in the first section, the optimization framework is described in the following section, then follows an experiments section to evaluate the disparity maps generated, followed by a section on using the disparity maps for forward transfer NVS, and finally the conclusion.

8.1 Objective function

In previous chapters the scene model has been $\mathcal{S} = \{\mathsf{I}_0^*, \mathsf{D}\}$, and priors have regularized either I_0^* or both I_0^* and D together. The stereo problem is different from NVS because one is given a *reference view*, I_0, which is a noisy version of the output image, I_0^*. In the previous chapter it was shown that incorporating a variable I_0^* into a graph-cuts-based stereo framework creates several complications, resulting from the need to have a joint prior, $p(\mathsf{I}_0^*, \mathsf{D})$. The vast majority of stereo methods ([SFVG04] is a rare exception) assume that $\mathsf{I}_0^* = \mathsf{I}_0$ in order to avoid this problem, allowing the prior to take the form $p(\mathsf{D}|\mathsf{I}_0)$, creating a CRF that regularizes only geometry. In this chapter I make the same approximating assumption, making the scene model $\mathcal{S} = \{\mathsf{I}_0, \mathsf{D}\}$, and the objective energy a function of disparity only, thus

$$E(\mathsf{D}|\mathsf{I}_0, .., \mathsf{I}_N) = E_{\text{photo}}(\mathsf{I}_1, .., \mathsf{I}_N|\mathsf{I}_0, \mathsf{D}) + E_{\text{smooth}}(\mathsf{D}|\mathsf{I}_0). \tag{8.1}$$

I will now define the constituent parts of this energy.

8.1.1 Data likelihood

The data likelihood term used here, including the occlusion model, is identical to that of the previous chapter, save for the form of photoconsistency kernel given in equation 7.5, which is changed to

$$\rho_{\text{d}}(\Delta I) = -\log\left(1 + \exp\left(-\frac{\|\Delta I\|^2}{\sigma_{\text{d}}}\right)\right), \tag{8.2}$$

where σ_{d} is a noise parameter. This kernel is also a robust measure of photoconsistency (with a maximum value of zero), but is based on a contaminated Gaussian [Sze99].

8.1.2 Surface smoothness

The surface smoothness prior takes much the same form of the previous chapter, being a sum over a set local neighbourhoods, $\mathbb{N}$, of a smoothness cost, ρ_{s}, modulated by a CRF term, W, thus

$$E_{\text{smooth}}(\mathsf{D}|\mathsf{I}_0) = \sum_{\mathcal{N}\in\mathbb{N}} W(\mathsf{I}_0, \mathcal{N}) \cdot \rho_{\text{s}}(S(\mathcal{N}, \mathsf{D})). \tag{8.3}$$

The CRF term modulates the smoothness term according to some function of the reference image, conditioning smoothness on I_0—this term is discussed further below. The function $S : \mathbb{R}^{|\mathcal{N}|} \mapsto \mathbb{R}$ is generally a derivative of disparity. The commonly used first derivative is given by

$$S(\{\mathbf{p},\mathbf{q}\},\mathsf{D}) = D(\mathbf{p}) - D(\mathbf{q}), \tag{8.4}$$

$\mathbb{N}$ being the set of all 2×1 and 1×2 patches in the image. This derivative permits fronto-parallel surfaces without penalty.

The assumption that surfaces in the scene are generally fronto-parallel is clearly false when one considers that a rotation of viewpoint immediately changes the assumption. Instead it is preferable to permit all *planar* surfaces without penalty, which can be achieved by using the second derivative of disparity (see appendix A). The full second derivative consists of the derivatives d_{xx}, d_{xy} and d_{yy}. While numerical computation of the derivatives d_{xx} and d_{yy} leads to triple cliques, d_{xy} leads to a quadruple clique and is therefore ignored, with the effect that a slightly larger class of surfaces are unpenalized (see appendix A). The second order prior used in this chapter is therefore defined as

$$S(\{\mathbf{p},\mathbf{q},\mathbf{r}\},\mathsf{D}) = D(\mathbf{p}) - 2D(\mathbf{q}) + D(\mathbf{r}), \tag{8.5}$$

where the neighbourhoods, $\mathcal{N} = \{\mathbf{p},\mathbf{q},\mathbf{r}\}$, are from the set of all 3×1 and 1×3 patches in the reference image.

The kernel placed on the derivative response is generally given by

$$\rho_{\mathrm{s}}(s) = \sigma_{\mathrm{s}}\left(\min\left(\frac{|s|}{\sigma_{\mathrm{s}}},1\right)\right)^{\gamma} \tag{8.6}$$

where $\gamma = 1$ or 2 and σ_{s} is a discontinuity preserving threshold, creating the truncated linear and truncated quadratic kernels respectively.

8.1.3 CRF weights

The CRF weights $W(\mathsf{I}_0,\mathcal{N})$ are set to encourage disparity edges to align with edges in the reference image, I_0. I use the method of Sun *et al.* [SZS03], and strengthen the smoothness constraint if the pixels in $\mathcal{N}$ are part of the same segment of an over-segmentation of I_0^*, encouraging discontinuities to align with segment boundaries. Note that this contrasts with the segment-based stereo methods [TSK01, BG04, HC04, KSK06, YWY$^+$06, BG07], which *force* discontinuities to align with segment boundaries.

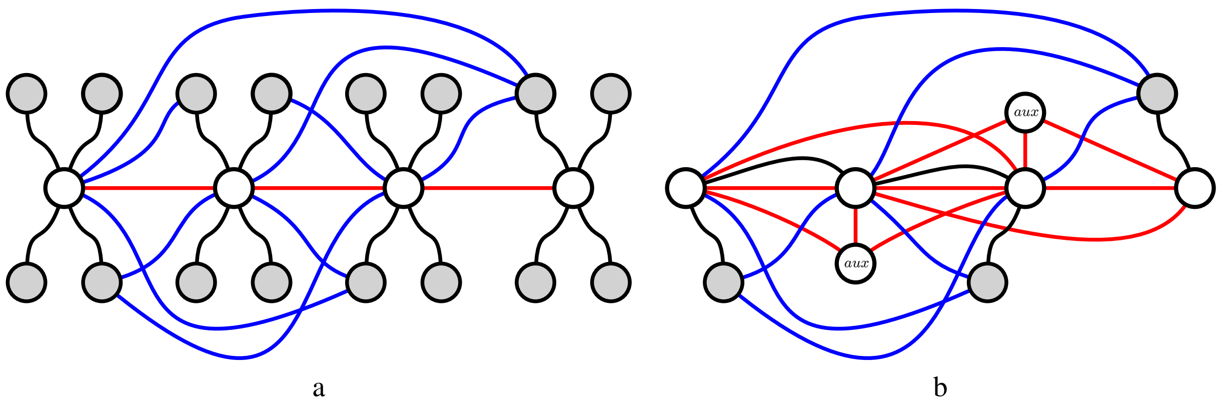


Figure 8.1: Graph construction. (a) The pairwise graph for the problem in figure 7.2(a), with the visibility constraint construction (blue lines) introduced in the previous chapter, and a first order smoothness prior (red lines). (b) The pairwise graph for the same problem, using a second order smoothness prior which generates six edges and an extra node (labelled aux) per clique. Note, however, that one smoothness edge (that edge between the two central nodes) is shared between neighbouring cliques. The visibility construction has also been simplified by removing visibility nodes (grey) which have fewer than two interacting pixels.

Here mean-shift [CM02, $h_s = 4$ and $h_r = 5$] is used to segment the reference image and one of two weights is assigned to each neighbourhood, depending on whether or not it overlaps a segmentation boundary. Precisely, if L is the map which assigns to each pixel its segmentation label, then

$$W(\mathbf{I}_0, \mathcal{N}) = \begin{cases} \lambda_\mathrm{h} & \text{if } L(\mathbf{p}) = L(\mathbf{q}) \; \forall \; \mathbf{p}, \mathbf{q} \in \mathcal{N} \\ \lambda_\mathrm{l} & \text{otherwise.} \end{cases} \tag{8.7}$$

8.2 Optimization

The above defines $E(\mathsf{D}|\mathbf{I}_0, .., \mathbf{I}_N)$ as a function of a real-valued disparity image, D. This section describes how I solve the following optimization problem:

$$\mathsf{D} = \underset{\mathsf{D}}{\operatorname{argmin}} \, E(\mathsf{D}|\mathbf{I}_0, .., \mathbf{I}_N) \tag{8.8}$$

In order to optimize the energy over the real-valued space, I follow the fusion move approach used in the previous chapter, reducing it to a sequence of binary problems which fuse a proposal disparity map, D^p, to the current estimate of the disparity, D_t, in an optimization problem given by equation 7.12.

8.2.1 Graph construction

The graph used to solve the binary optimization problem of equation 7.12 makes use of two contributions from the previous chapter—the pairwise geometrical occlusion model, and the triple clique decomposition for non-submodular cliques. Figure 8.1(a) recalls the construction introduced in that chapter for the simple stereo problem of figure 7.2(a), with the novel visibility nodes. As with the NVS problem, visibility nodes with fewer than two interactions can be removed, with data costs being added either to the unary cost of the relevant disparity node in the case of no interactions, or to an edge connecting said disparity node to that of the interacting pixel in the case of one interaction. Figure 8.1(b) shows this simplification, and also shows the pairwise decomposition (using the method of [KZ04], as per the previous chapter) of the triple clique smoothness term of a second order prior.

There are three types of energy functional used in the graph: data cost, visibility constraint and smoothness cliques. A data cost clique, $\phi^{a}_{\mathbf{x}i}$, models the energy of equation 7.4 as follows

$$\phi^{a}_{\mathbf{x}i}(B(\mathbf{x}), V^{a}_{i}(\mathbf{x})) = \begin{cases} 0 & \text{if } B(\mathbf{x}) \neq a \\ \nu & \text{else if } V^{a}_{i}(\mathbf{x}) = 0 \\ \rho_{\mathrm{d}}\left(I_i\left(\pi_i\left(\mathbf{x}, d\right)\right) - I_0(\mathbf{x})\right) & \text{otherwise} \end{cases} \tag{8.9}$$

$$d = (1 - B(\mathbf{x})) \cdot D^{\mathrm{p}}(\mathbf{x}) + B(\mathbf{x}) \cdot D_t(\mathbf{x}). \tag{8.10}$$

If $\nu > \rho_{\mathrm{d}}(\cdot)$ (which it is here) then the value of $V^{a}_{i}(\mathbf{x})$ will default to the lower energy state of 1. In order to flip its state to 0 and ensure the occlusion cost is paid in the case of an occlusion, a visibility constraint edge, $\phi^{ba}_{\mathbf{px}i}$, for a pixel $\mathbf{p}$ which occludes $V^{a}_{i}(\mathbf{x})$ when $B(\mathbf{p}) = b$ must have the following costs:

$$\phi^{ba}_{\mathbf{px}i}(B(\mathbf{p}), V^{a}_{i}(\mathbf{x})) = \begin{cases} \Omega & \text{if } B(\mathbf{p}) = b \text{ and } V^{a}_{i}(\mathbf{x}) = 1 \\ 0 & \text{otherwise} \end{cases} \tag{8.11}$$

$$\Omega > \nu - \rho_{\mathrm{d}}\left(I_i\left(\pi_i\left(\mathbf{x}, d\right)\right) - I_0(\mathbf{x})\right). \tag{8.12}$$

Note that the cost Ω is never actually paid. The entries of smoothness clique energy tables, which model equation 8.3, are computed (for a second order prior, by way of example) as

$$\phi_{\mathbf{pqr}}(B(\mathbf{p}) = p, B(\mathbf{q}) = q, B(\mathbf{r}) = r) = \rho_{\mathrm{s}}\begin{pmatrix} p \cdot D_t(\mathbf{p}) + (1-p) \cdot D^{\mathrm{p}}(\mathbf{p}) - \\ 2q \cdot D_t(\mathbf{q}) - 2(1-q) \cdot D^{\mathrm{p}}(\mathbf{q}) \\ + r \cdot D_t(\mathbf{r}) + (1-r) \cdot D^{\mathrm{p}}(\mathbf{r}) \end{pmatrix} \tag{8.13}$$

Given that the proposal disparity maps, D^{p}, are general, no guarantees on the submodularity of this term can be made, regardless of the order of the prior.

As the graph constructed is not guaranteed to be submodular the optimizer used here is again QPBO, and the various label fixing strategies described in §7.2.2 are used and compared in §8.3.2.

8.2.2 Proposal generation

In α-expansion stereo methods [BVZ01, KZ02], as with the previous chapter, the proposal disparity at each step is a fronto-parallel plane. As shown in [BVZ01], repeated fusion of these proposals leads to a strong local optimum in the case of a first order prior. In the case of a second order prior, the nature of these proposal disparity maps has a much larger effect on the generated disparity map, as shown empirically in §8.3. As a result, I use a variety of schemes for generating the j^{th} proposal disparity map $\mathsf{D}^{\mathsf{p}}_j$, as follows:

SameUni Draw d_j from a uniform distribution, and set $D^{\mathsf{p}}_j(\mathbf{x}) = d_j$ for all $\mathbf{x}$, *i.e.* a fronto-parallel plane at a random disparity.

SegPln Uses the ad hoc approach of segmentation-based methods [KSK06, YWY$^+$06] to generate a set of piecewise-planar proposals, which are then cycled through continuously. In this implementation, demonstrated in figure 8.2, the first stage of proposal generation involves a local window matching process [SS02] to generate an approximate (very noisy) disparity map, by averaging the cost given by equation 8.2 over a pixel's 5×5 neighbourhood, summed over input images, at a range of disparities, and selecting the disparity which produces the lowest cost.

I then use two different image segmentation algorithms, one colour-based [CM02], and one texture-based [FH04b], and 14 sets of parameters in total, to generate segmentations of $\mathbf{I}_0$, ranging from highly under-segmented to highly over-segmented. For each segment in each segmentation, LO-RANSAC [CMO04] is used to find the plane that produces the greatest number of inlying correspondences from the first stage (given a suitable distance threshold), and set all the pixels in the segment to lie on that plane.

Smooth $D^{\mathsf{p}}_j(\mathbf{x}) = (D_j(\mathbf{x}+\Delta)+D_j(\mathbf{x}-\Delta))/2$, where $\Delta = [0,1]$ when j is odd, and $\Delta = [1,0]$ when j is even.

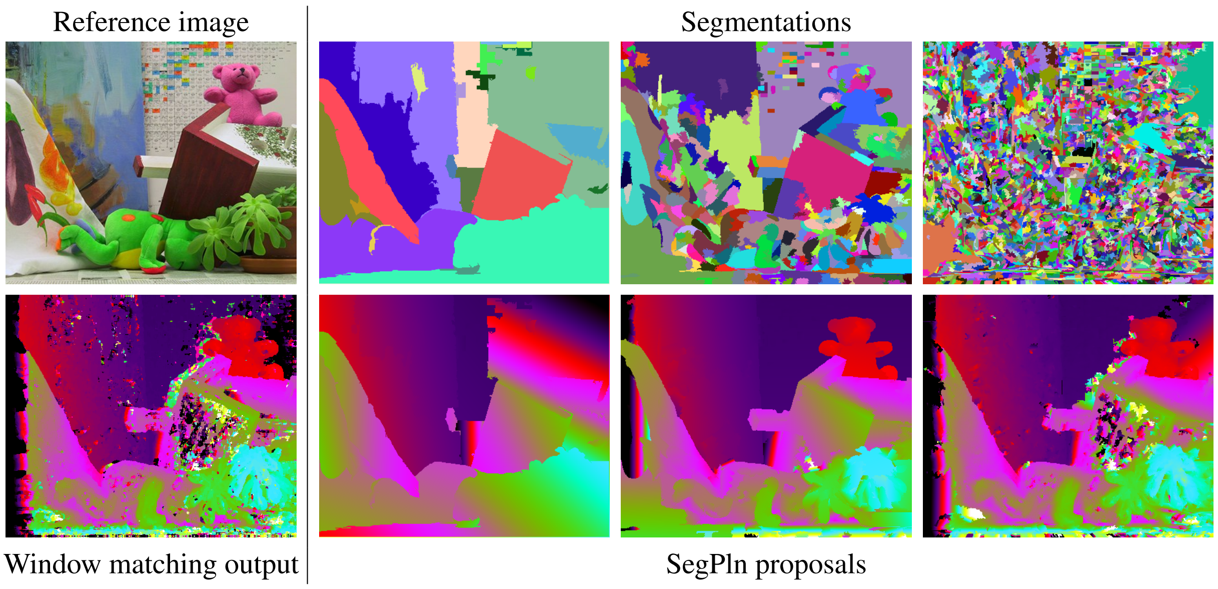


Figure 8.2: SegPln proposal generation. *Top row:* $\mathbf{I}_0$, and 3 of its 14 segmentations. *Bottom row:* approximate disparity map from window matching, and 3 SegPln proposals generated by fitting planes to each segment in the above segmentations.

These proposal methods represent the different approaches used by the main types of stereo algorithms: the fronto-parallel proposals of SameUni are essentially those used at each iteration of an α-expansion-based stereo algorithm (except drawn from a continuous, rather than discrete, space); SegPln proposals are those used by segment-based algorithms; Smooth proposals, generated by a smoothing operation on the current disparity map, can be viewed as a proxy for local methods such as gradient descent. With QPBO-based fusion, one gains the benefits of all these algorithms—indeed, any stereo algorithm available—without affecting the global optimum. For example, the SegPln proposals, the main workhorse of our algorithm, are produced with a range of algorithms and parameter settings; in general one can expect these disparity maps to be correct in some parts of the image, and for some parameter settings, but that no settings can be found for which any algorithm works best. By fusing the proposals in a well-defined energy minimization framework, the parameter sensitivity of these methods is turned into an advantage: the algorithm selects the best parts from each proposal, at the pixel (as opposed to segment) level.

8.2.3 Implementation details

Some further implementation notes will allow the reader to more accurately replicate this method.

The range of disparities searched over for a particular image sequence are normalized to $[0, 1]$ prior to the evaluation of E_{smooth}, in order to make the objective function invariant to image baseline, camera calibration and depth of field. The initial depth map, D_0, is set to $D_0(\mathbf{x}) = \text{rand}[0, 1]$ for each $\mathbf{x}$ independently. Optimization is halted when the average decrease in energy over the last 20 iterations drops below 0.01% of the current energy.

As the Smooth proposal only performs well when applying it to an approximately correct disparity map, the proposal set is prefixed with two such disparity maps, generated from the fusion of the other two proposal sets, followed by four iterations of smoothing, and this set is then repeated every six iterations.

The same parameter settings were used for all examples: $\nu = 0.01$, $\sigma_{\text{d}} = 30C$, $\lambda_{\text{l}} = 9N$, $\lambda_{\text{h}} = 108N$, $\sigma_{\text{s}} = 0.02$, where C is the number of colour channels per input image. These settings were obtained by visual evaluation of results on a small number of Middlebury images (although it must be emphasized that they were not chosen with any reference to the Middlebury evaluation score) over a range of parameter settings. The order of the prior was found not to change the relative performance of parameter sets significantly.

8.3 Experiments

In this section I describe the experiments carried out to evaluate the efficacy of QPBO in optimizing the non-submodular energy, the trade-offs of each of the QPBO labelling methods, the effect of using different disparity proposals, and comparing the method presented, with its second-order prior, to the same method with a first-order prior, and other, competing approaches to stereo through the Middlebury evaluation framework [SS08]. Since no ground truth disparity is available for the sequences used up to this point (for NVS), the experiments are carried out on some additional, Middlebury datasets [SS08], shown in figure 8.3, for which ground truth disparity is available, as well as the Monkey and Plant sequences. Each Middlebury sequence consists of two rectified views.

The optimization method used in each experiment is characterized by the order of the prior ("1op" for first-order prior, *etc.*), the smoothness kernel ("linear" for $\gamma = 1$, "quadratic" for $\gamma = 2$), the set of proposals, and the fusion strategy, *e.g.* "2op, linear, SameUni, QPBOI-R", or "1op, quadratic, SegPln, QPBOP".

Figure 8.3: Middlebury sequences. The reference images for the three Middlebury sequences Venus, Teddy and Cloth3, obtained from [SS08].

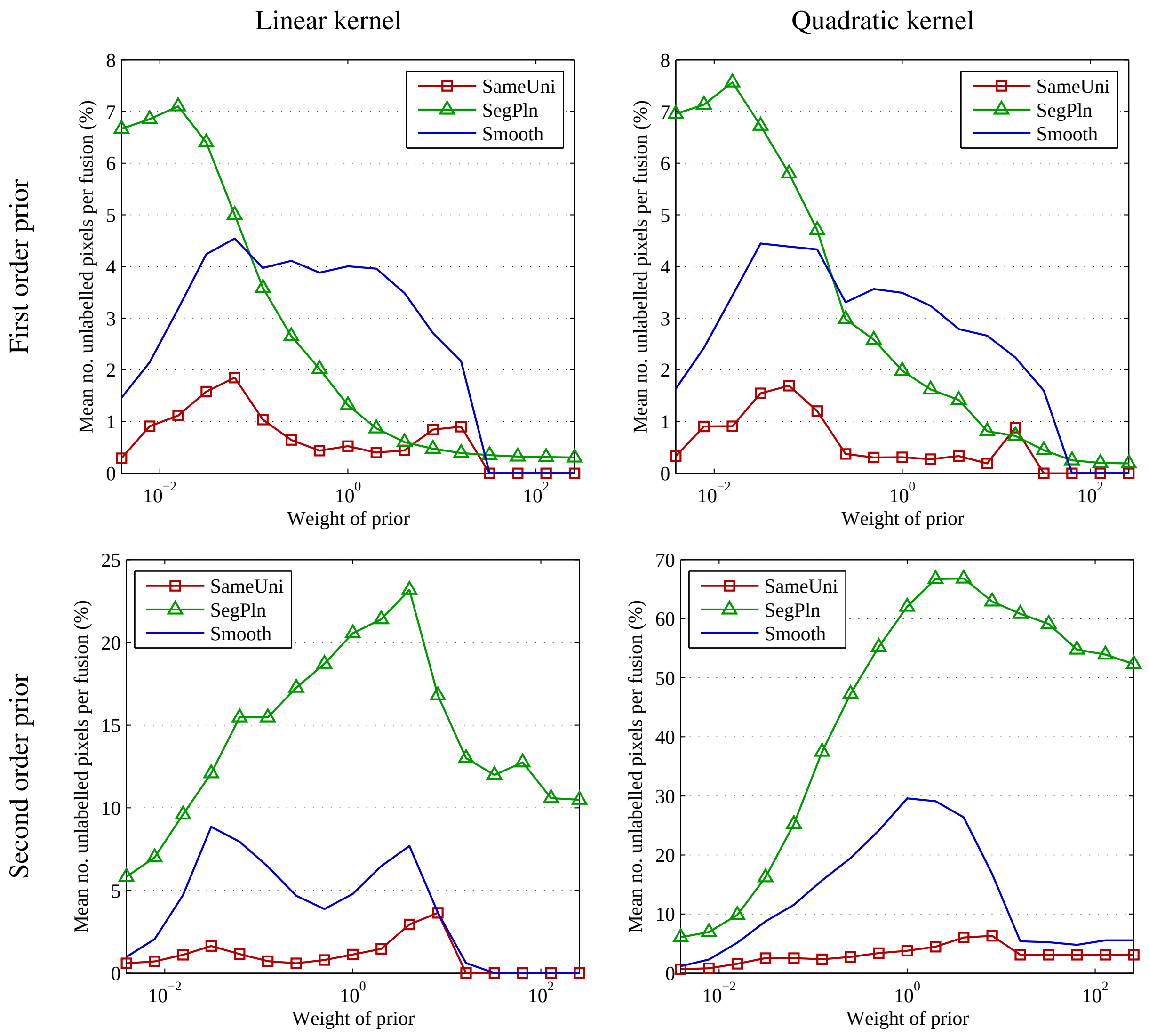


Figure 8.4: Unlabelled nodes. The average number of nodes unlabelled by QPBO across a range of variables.

8.3.1 Number of unlabelled nodes

The aim of the first experiment was to determine whether optimization of the non-submodular pairwise binary graph described in §8.2.1 (an NP-hard problem) was feasible using QPBO. The proportion of pixels that are labelled by QPBO has a direct impact on the quality of the solution found—trivially, if no nodes are labelled then (using QPBO-F) the final solution will be the same as the initial solution. It is therefore important to have as many nodes labelled as possible.

I used QPBO-F in these experiments, but varied the proposal schemes, order of prior and prior kernel to see what effect these had on the number of unlabelled nodes. I additionally varied the weight of the prior term, E_{smooth}, with respect to the data term, E_{photo}. The experiments were carried out on both the Middlebury Teddy and Cones sequences [SS08], and results were averaged across both sequences and the binary optimizations within each category.

Figure 8.4 shows the results of these experiments. When using the first order prior (*top row*) the results are very similar for truncated linear and quadratic kernels, with <0.5% of nodes unlabelled with SameUni proposals, around 2% with SegPln proposals and around 4% with Smooth proposals, with a prior weight of 1 (*i.e.* the default weight). What is interesting, even surprising to note is the way each of the proposal schemes affect the number of unlabelled nodes differently as the weight changes. Results with a second order prior (*bottom row*) are significantly different. While the level of unlabelled nodes for SameUni proposals remains in the order of 1%, SegPln proposals now generate the most unlabelled nodes of the three proposal schemes, and significantly more than with a first order prior—over 20% with a truncated linear kernel, and almost 70% with a truncated quadratic kernel. Additionally, variations across prior weights are different again from those of the first order prior.

8.3.2 Comparison of label fixing strategies

With a relatively high number of unlabelled nodes when using a second order prior, it is clearly important to try to fix them as effectively as possible. In the following experiments I again tested the six post-QPBO labelling strategies described in §7.2.2, in the context of this problem. I used 2op, linear, and SegPln settings as these give a level of unlabelled nodes that is high, but not prohibitively so (in the case of the more costly strategies).

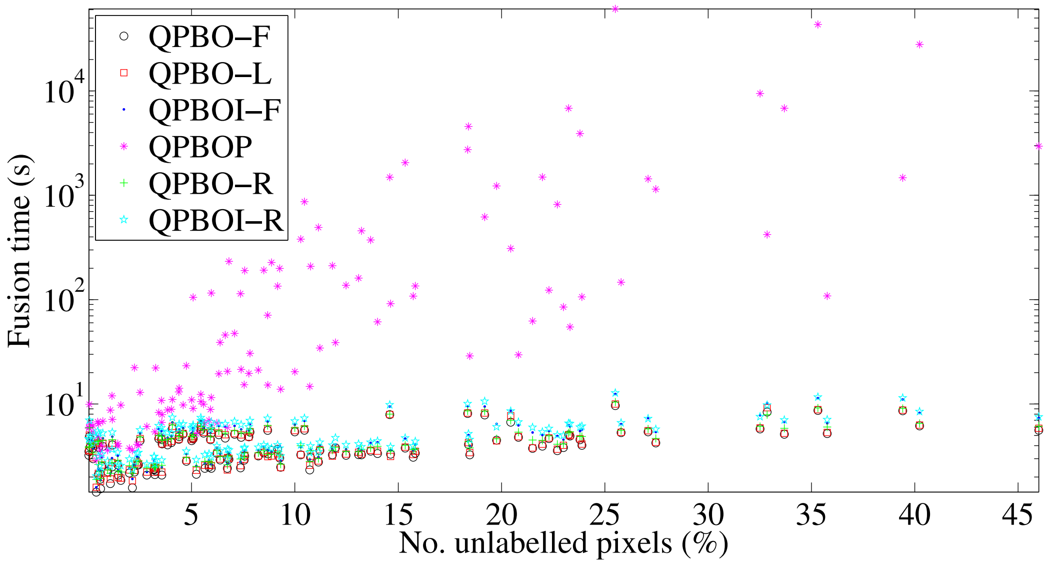
The first experiment involved trying all of the fusion strategies at each iteration, on exactly the same binary optimization (the optimal labelling given by QPBOP was used to update D), and this was carried out on the Middlebury Teddy, Cones and Cloth3 sequences and the results concatenated. The speed of the fusion strategy is important. Figure 8.5 shows (*top*) that QPBOP rapidly becomes several orders of magnitude slower as the number of unlabelled pixels rises, while other methods show a more modest increase over the same range; of these there is only a fractional difference in speed, though order of fastest to slowest is consistently QPBO-F, QPBO-L, QPBO-R, QPBOI-F, QPBOI-R. Also important is the energy reduction performance of each strategy—QPBOP, which gives an optimal solution, performs best, while QPBO-F, with the simplest labelling strategy, is guaranteed to perform worst. Figure 8.5 also shows (*bottom*) how the other strategies perform relative to these two, by normalizing the energy reduction between 0, representing the performance of QPBO-F, and 1, representing the performance of QPBOP. The normalized energy reduction of the four remaining strategies were discretized into 20 equally sized bins, which (except the bin for 0–0.05) are shown in the stacked bar graph of figure 8.5. The graph indicates that QPBOI-R achieves the largest energy reduction after QPBOP, based on it having the largest mass towards the right of the graph.

The second experiment tested the performance of each strategy over an entire iterative optimization, by running them individually until convergence on the same set of proposals. Table 8.1 shows the quantitative results of this experiment on the Teddy sequence. In terms of performance, QPBOI-R registers the lowest energy after QPBOP, in line with the previous experiment. In terms of time per fusion, QPBOP is the slowest method by two orders of magnitude, but converges in the smallest number of iterations and with the lowest average number of unlabelled nodes. The QPBOI methods are slower than the remaining methods, due to their costly graph resolving; however, QPBOI-F is more than twice as slow as QPBOI-R, which was not predicted by the results of the previous experiment. This is most likely due to its larger number of unlabelled nodes per fusion, which has a linear effect on the time taken to fix unlabelled nodes—there is a trend for the methods which fix nodes better (*i.e.* generate lower energies) to also generate fewer unlabelled nodes in successive iterations. QPBO-R is competitive with the other non-QPBOI methods in terms of speed, while outputting a lower energy.

Considering the trade-off between time and efficacy, QPBOI-R was deemed the most suitable method for this problem, and used in all further experiments save those involving a 2op quadratic prior—the potentially high number of unlabelled pixels involved in the latter optimizations can make the QPBOI method prohibitively expensive also, so QPBO-R was used in this case instead.

	QPBO-F	QPBO-L	QPBOI-F	QPBOP	QPBO-R	QPBOI-R
Energy (% > QPBOP)	1.13	0.666	0.374	0	0.571	0.279
Fusion time (avg. secs.)	8.33	8.24	50.9	1680	9.77	17.3
No. iterations	42	44	45	37	42	42
Unlabelled (avg. %)	27.8	12.8	16.1	11.0	12.8	13.1

Table 8.1: Results of the various fusion strategies applied to the Teddy sequence using "2op, linear, SegPln, QPBOI-R".



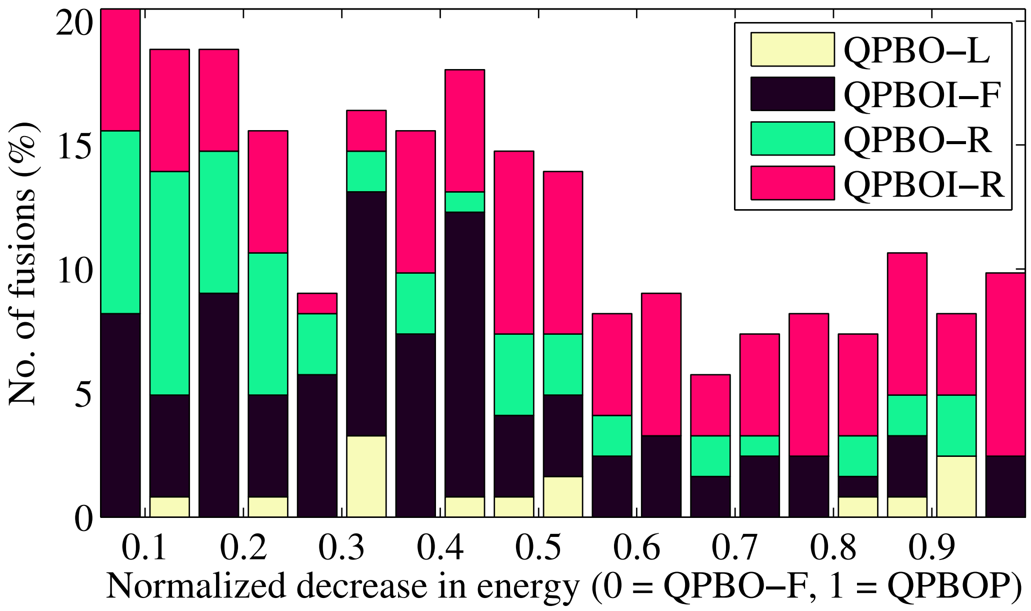


Figure 8.5: Label fixing. *Top*: Fusion times for individual binary optimizations with each of the six label fixing methods of §7.2.2, plotted against number of unlabelled nodes. *Bottom*: A stacked histogram of energy reduction performance for the four label fixing methods QPBO-L, QPBOI-F, QPBO-R and QPBO-R, normalized between the performances of QPBO-F (0) and QPBOP (1).

8.3.3 Proposals

In §8.2.2 three proposal schemes were introduced. Figure 8.6 demonstrates the effect of using these proposals on the Venus sequence, under the various smoothness priors. The number in the bottom left corner of each image is the number of fusion iterations used to generate that image (when the convergence criterion was met). From these it can be seen that many more SameUni proposals (drawn from an infinite set) are required before convergence, compared with the other approaches, making this a slower approach. Also clear is that the output from the SameUni proposals is always piecewise-fronto-parallel, regardless of the prior used, in spite of the fact that the lower energy final output of the Smooth proposals (which incorporates the other two outputs) are only piecewise-fronto-parallel with the first order linear prior, as one would expect. Since the disparity converges on a local minimum w.r.t. the fusion moves, the moves themselves must create a convergence basin which includes only piecewise-fronto-parallel solutions. Indeed, this can be seen to be the case when one considers the simple problem of figure 8.7—given a fronto-parallel current solution and planar optimal solution, any intermediate solutions increase the E_{smooth} cost, thereby creating an energy hump which cannot be overcome by any single fusion. This suggests that the SameUni proposals are only suitable for use with a first order linear prior.

The output from SegPln proposals, with their planar segments which contain many small changes in disparity instead of a few large ones, is forced to be as fronto-parallel as possible by the first order linear prior, generating an output that is far from accurate. However, these proposals are favoured by both the truncated quadratic first order prior and the two forms of second order prior, generating plausible results that are incorporated into the output of the Smooth proposals also.

As well as combining the outputs from the SameUni and SegPln proposals, the Smooth proposals also allow disparity gradient discontinuities to become smoother. This effect can be seen in figure 8.8, in particular by comparing the output from the Smooth proposals (e) to pre-smoothed disparity (d), *i.e.* fusion of SegPln and SameUni outputs, for the linear second order prior—gradient discontinuities (*e.g.* top right & bottom centre of the surface) become more curved. What this shows is that the linear second order prior, like the quadratic first and second order priors, does allow curved surfaces where the data supports this, unlike the linear first order prior.

The success of both the SegPln and Smooth proposal schemes, in spite of occasionally high numbers of unlabelled nodes, suggests that the optimization framework put forward here will work well with any arbitrary proposals, and not just the three schemes put forward here as examples.

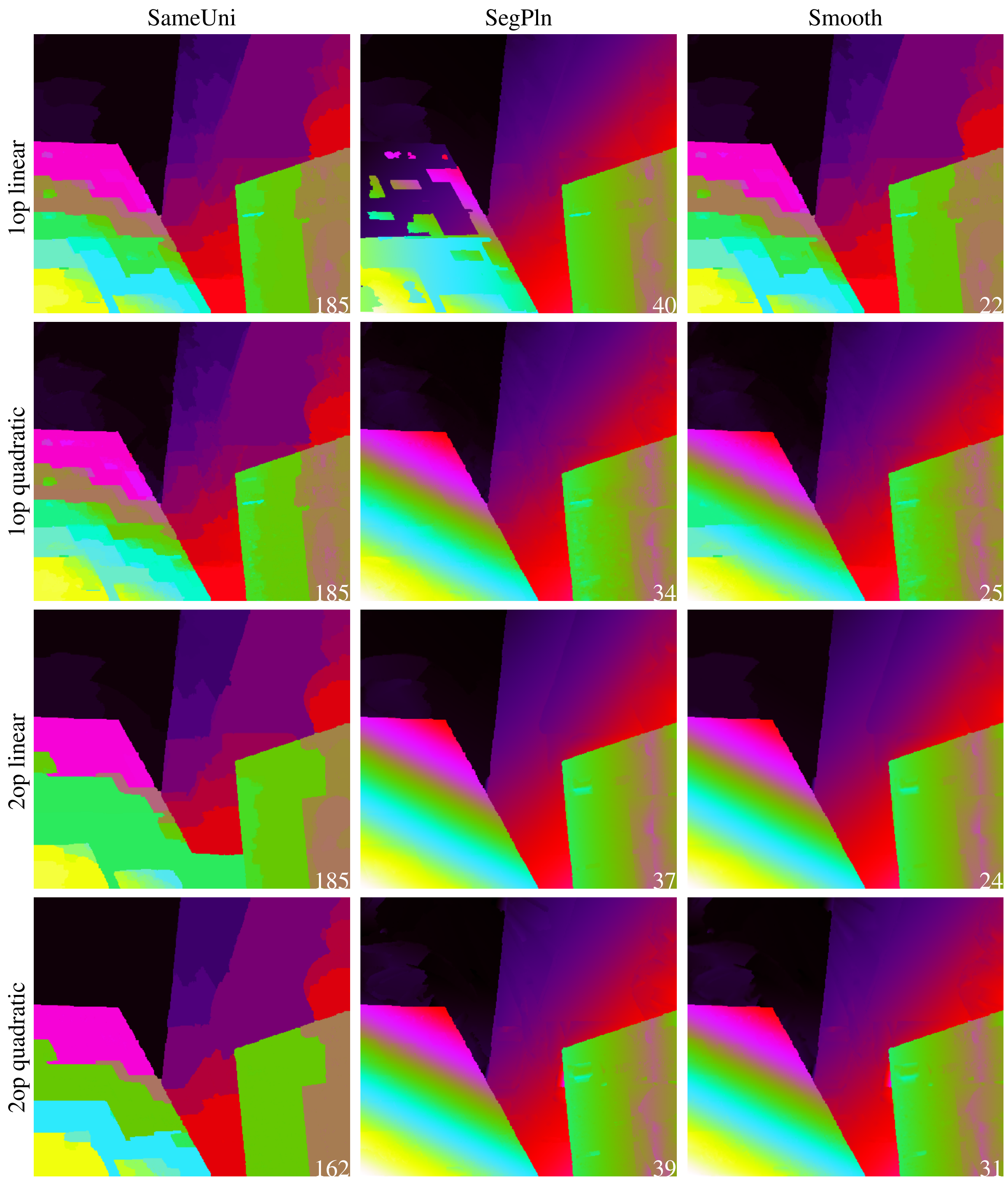


Figure 8.6: Venus results. Effect of proposals and the form of prior on D for the Venus sequence. The number of fusions until convergence is shown in the bottom right corner of each image.

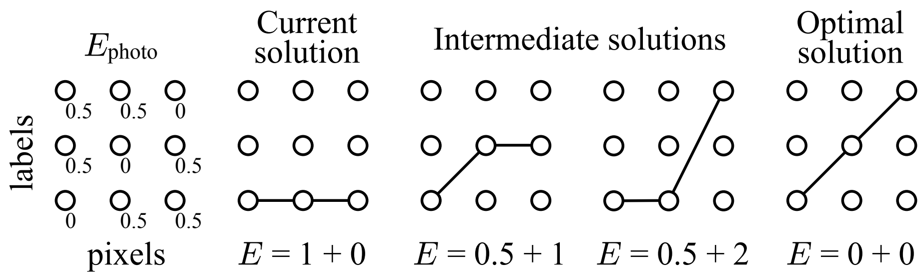


Figure 8.7: Energy hump. The current disparity estimate cannot move to the optimal disparity solution with fronto-parallel proposals without going through one of the two intermediate solutions shown, but these have a higher energy than the current solution. The current solution is therefore a local minimum w.r.t. that form of proposal.

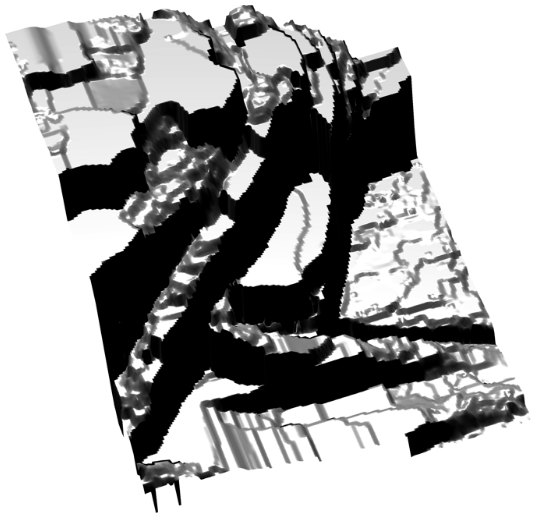
8.3.4 Comparison of priors

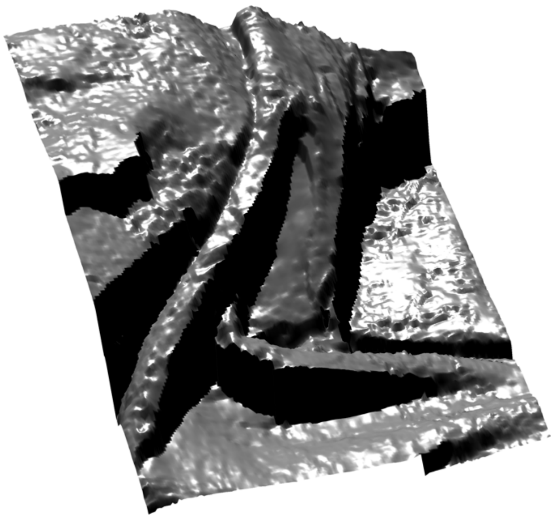
To evaluate the performance of second order priors it is important to compare their results with those of first order priors, all generated using the same stereo framework presented here. Results with all four forms of prior—1op linear, 1op quadratic, 2op linear and 2op quadratic—are given for three Middlebury sequences: Venus (figure 8.6), Cloth3 (figure 8.8) and Teddy (figure 8.10).

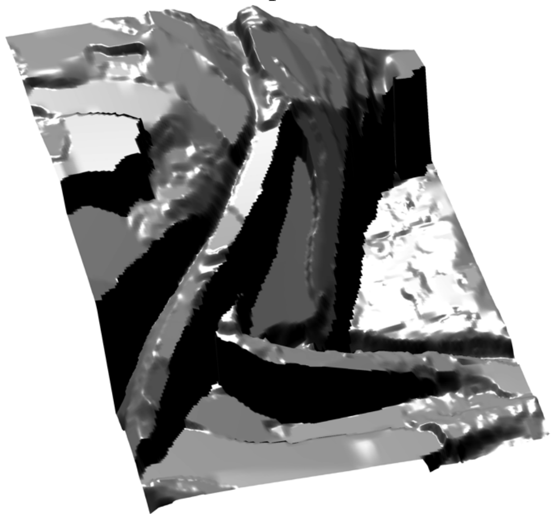
The first thing to notice is that all results using a 1op linear prior are highly piecewise-fronto-parallel. This results from the facts that the prior permits only fronto-parallel surfaces with zero cost, and that the concave kernel prefers a large jump in disparity over many small ones, generating a highly unnatural reconstruction.

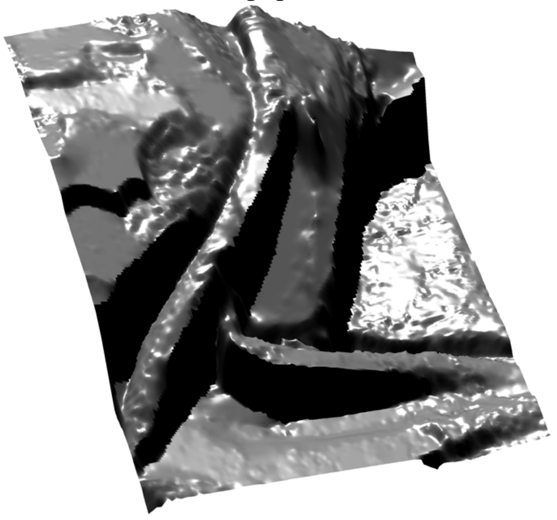
The convex centre of the quadratic kernel overcomes this piecewise nature, preferring several smaller jumps in disparity over a larger one. In the case of the first order prior this allows the generation of both planar (figure 8.6) and curved surfaces (figure 8.8(b)). However, it should be noted that these non-fronto-parallel surfaces must pay a smoothness cost, and this cost must be outweighed by the savings in data cost of the surface over a fronto-parallel one. In the cases where the surfaces are not textured enough or the surface gradient is too great the reconstruction inevitably reverts to piecewise fronto-parallel planes, as shown in figures 8.9 and 8.11 respectively.

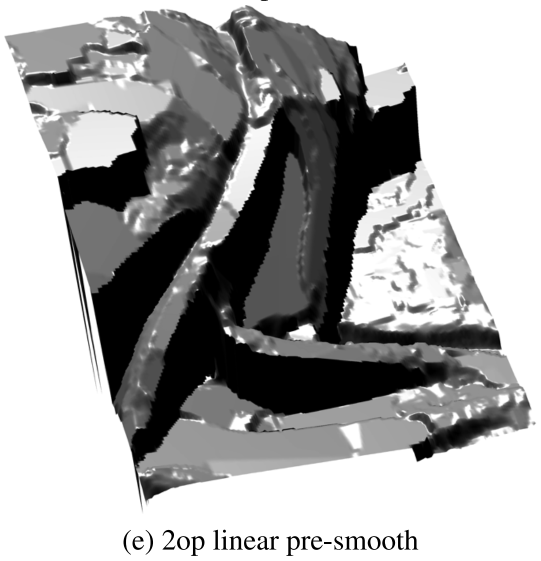
The second order prior can equally generate curved surfaces where the data costs favour this, but additionally allows planar surfaces to be reconstructed in low texture and steep gradient regions, as seen in figures 8.9 and 8.11, improving the results in these regions. The choice of kernel has a lesser impact on second order priors (see figure 8.8), but the truncated linear kernel generates a more piecewise-planar output than the quadratic kernel, which allows for slightly rougher surfaces at a fine scale. What is


(a) 1op linear


(b) 1op quadratic


(c) 2op linear


(d) 2op quadratic


(e) 2op linear pre-smooth

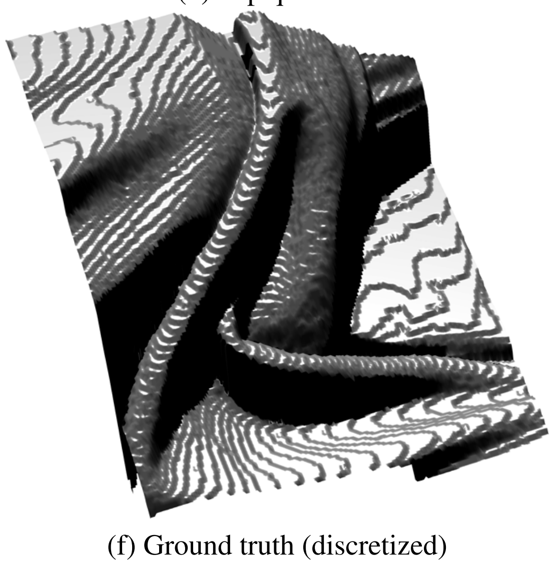

(f) Ground truth (discretized)

Figure 8.8: Cloth3 results. Results for a region of the Middlebury Cloth3 sequence, displayed as a shaded 3-d disparity surface.

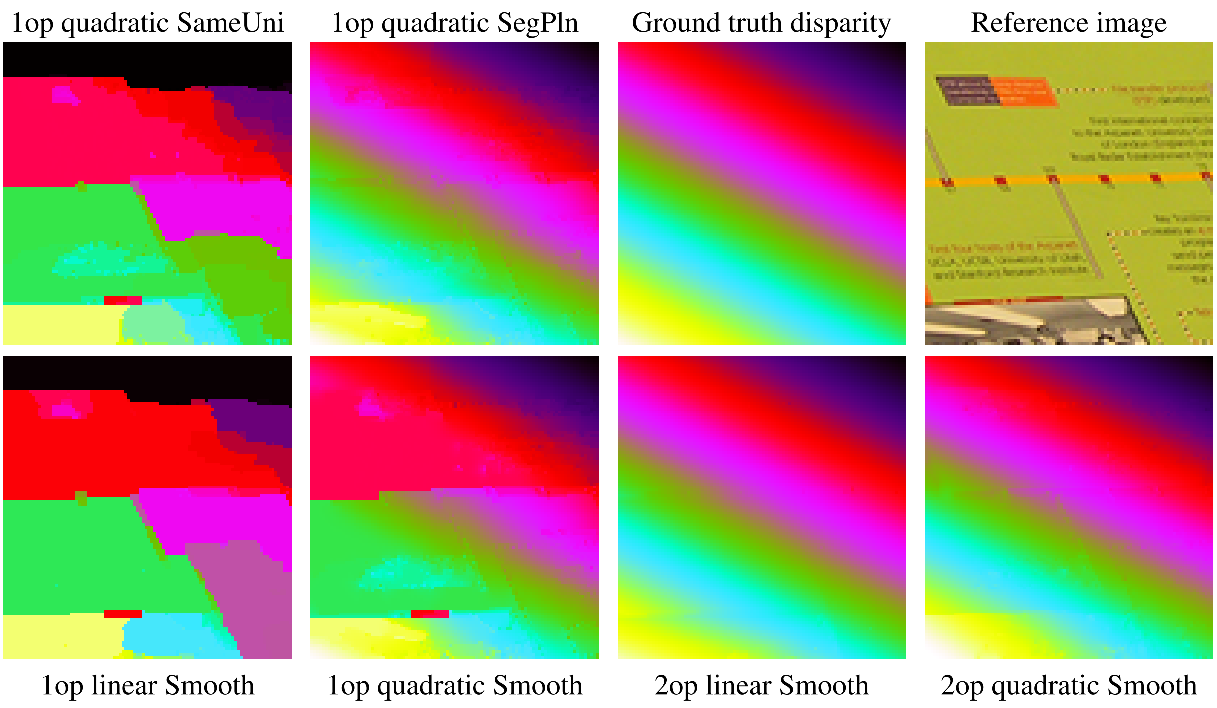


Figure 8.9: Venus zooms. Zooms of the Venus sequence results, showing a textureless, slanted area that the first order priors fail to reconstruct.

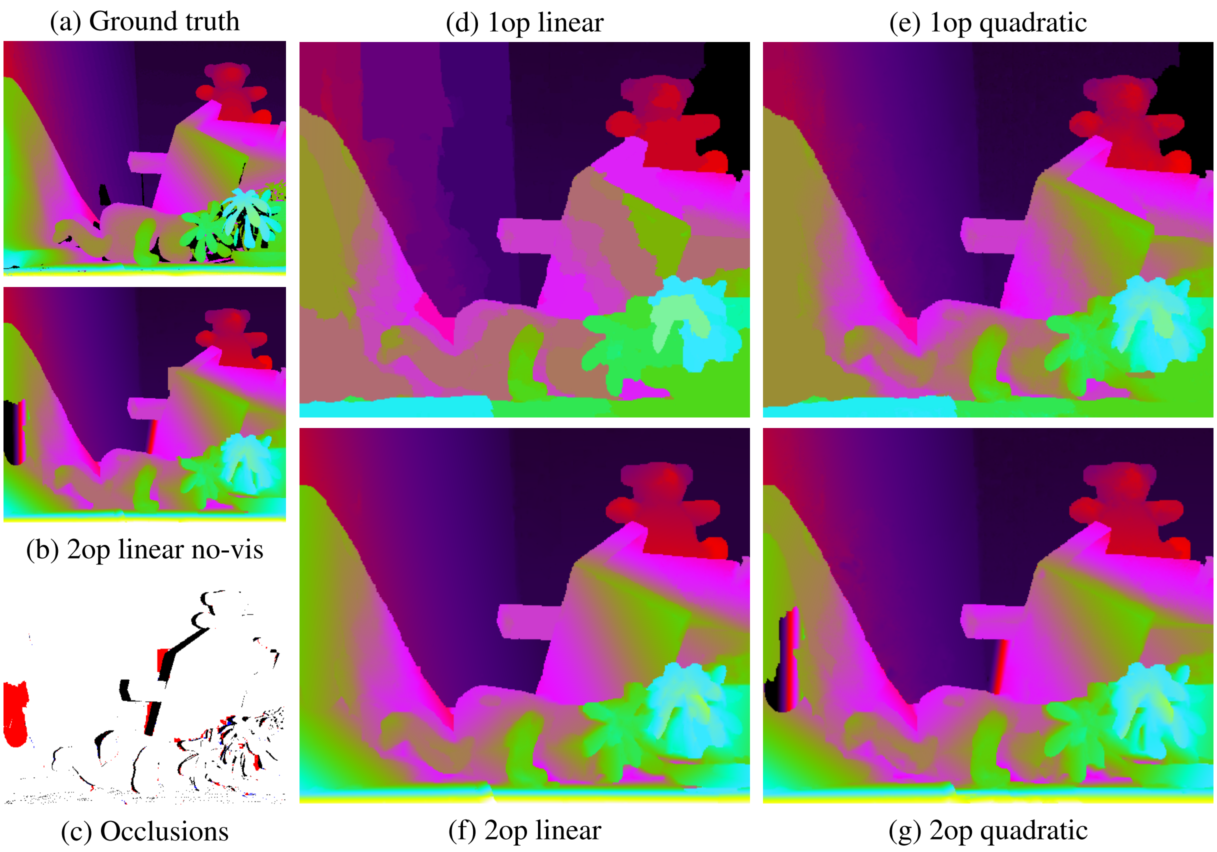


Figure 8.10: Teddy results. Output disparities for the Teddy sequence.

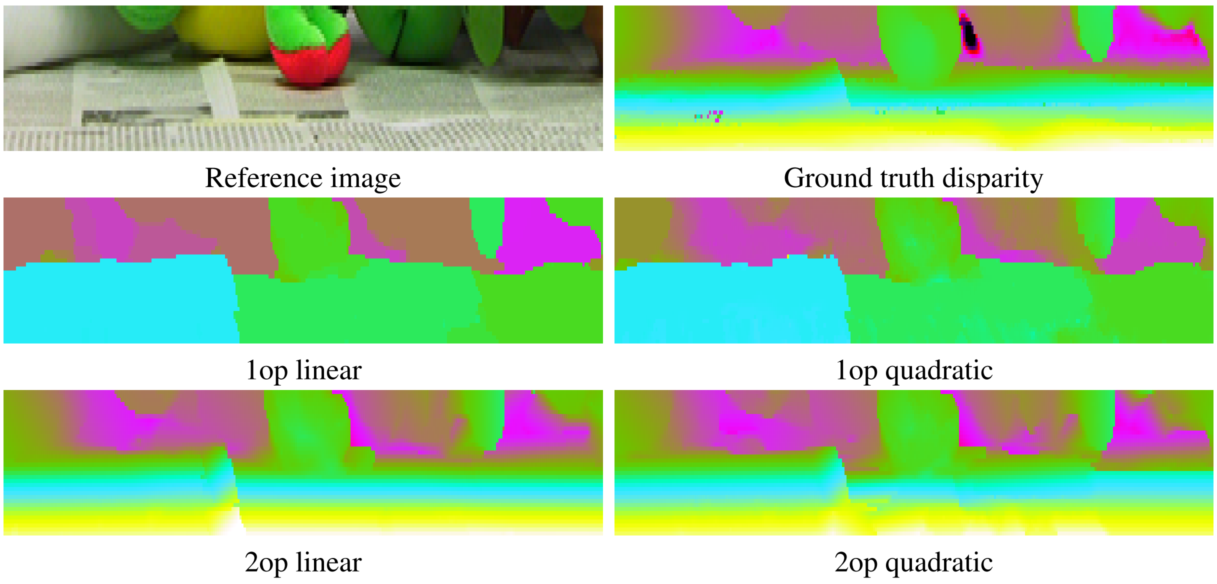


Figure 8.11: Teddy zooms. Zooms (stretched vertically) of the Teddy sequence results showing a highly slanted, textured region which the first order priors fail to reconstruct.

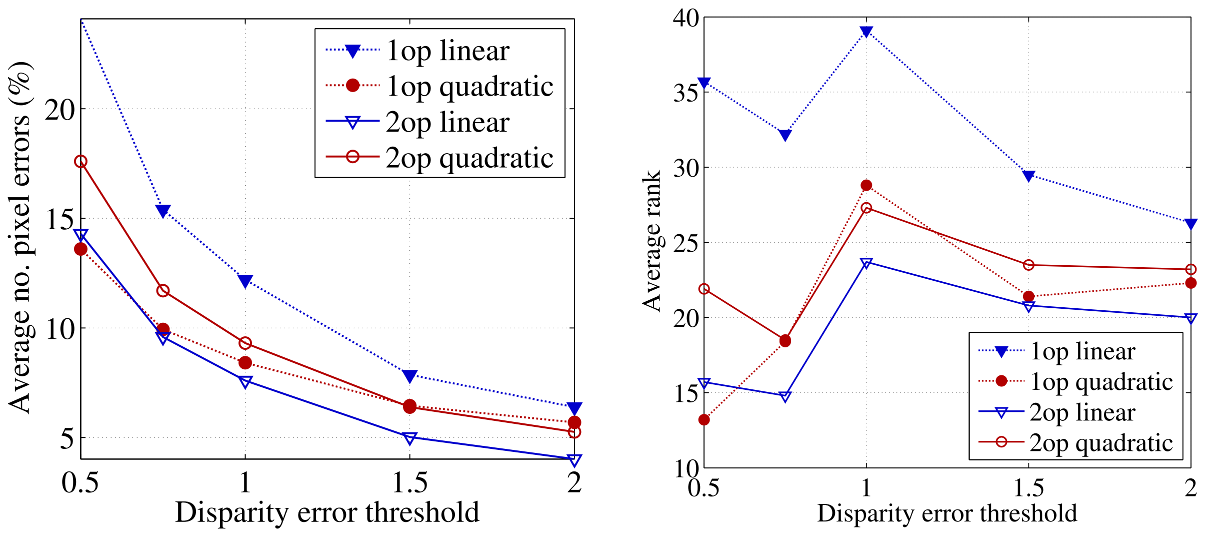


Figure 8.12: Middlebury scores. Scores for different priors in the Middlebury stereo evaluation framework.

noticeable with a truncated quadratic second order prior are that large scale artifacts can occur, *e.g.* to the left of figure 8.10(g), due to problems in the optimization caused by the high number of unlabelled pixels with this prior.

Figure 8.12 shows the quantitative results from the Middlebury evaluation framework for all combinations of smoothness prior order and kernel. Error rates at various error thresholds (*left*) show that the 1op linear prior consistently performs worst, while the 2op linear prior performs best at all error thresholds save the lowest, at which 1op quadratic performs best—the performance of this latter prior drops behind at higher thresholds due to the greater number of gross errors caused by the fronto-parallel preference of the prior. The 2op quadratic prior is always a fixed distance behind the 1op linear prior, a result of the extra gross errors caused by the optimization. Performance compared with other algorithms (*right*) shows exactly the same in terms of the relative performance of the priors, but it also shows that, while all priors perform worst at an error threshold of 1 (the default error threshold, which many algorithms are tuned to perform well at), the performance of all priors but 1op linear improves more at lower error thresholds than at higher ones, indicating their better sub-pixel accuracy in comparison to other methods.

8.3.5 Visibility

Figure 8.10(c) highlights the benefits of a visibility constraint. It shows the visibility map for $\mathbf{I}_1$ of Teddy—pixels deemed occluded according to the following disparity maps are painted (covering the previous colour) in the following order: 2op linear prior without visibility constraint (b), red; 2op linear prior with visibility constraint (f), blue; ground truth (a), black. Comparing numbers of red and blue pixels one can see that the visibility constraint reduces the number of falsely occluded pixels—it essentially encourages uniqueness of correspondences between input images. As unique correspondence is a constraint on real-world scenes, incorporating such a constraint in a stereo framework produces better results.

8.3.6 Multiple and arbitrary views

The formulation of our objective function allows for any number of input images to be used, and for those images to have arbitrary viewpoints. This makes it suitable for generating disparity maps for NVS, where there are generally multiple but arbitrarily placed input images are available. Results using the 2op linear prior are shown for two such sequences, Monkey and Plant, are shown in figure 8.13.

Figure 8.13(b) shows the output disparity for the image in (a), using only three input images. Having this explicit geometry reconstruction allows the scene to be rendered from angles well outside the range of the input images, as shown in (f). This can leave large holes where the scene was occluded in the input view. In particular, the rendering in (f) shows that smooth, non-fronto-parallel surfaces such as the leaves and even the nose of the toy have been faithfully reconstructed. There was little or no qualitative improvement between $N = 2$ (shown) and $N > 2$, something I believe can be attributed to the fact that three views are sufficient (in this case) to ensure that each pixel of $\mathbf{I}_0$ is visible in at least one other view. However, should more views be required, figure 8.13(c), shows that, in practice, the time per fusion iteration rises approximately linearly with N.

In contrast to the Plant sequence, the output disparity for the Monkey sequence, shown in figure 8.13(e) is highly fronto-parallel. This does not mean that a fronto-parallel solution is the lowest energy solution possible—rather, it suggests that the other proposal schemes failed to generate more likely proposals. Indeed, the highly textured nature of the monkey's fur is not at all suited to the SegPln scheme, which generates piecewise-planar solutions. It is possible that more advanced proposal schemes, especially those that generate disparity maps using local optimization approaches (*e.g.* [SFVG04]), hence find local optima which can then be fused, would generate better results. The current results do show that fronto-parallel solutions do generate noticeable artifacts when rendered in a new view (*e.g.* the ledges between different planes, visible to the right of the monkey's head), underscoring the need for a second-order prior.

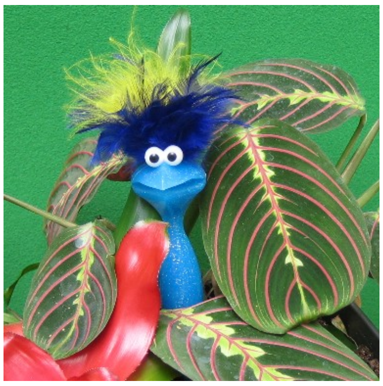
8.4 NVS from stereo

Once disparity maps have been computed for every input image in a sequence, they can be used to render new views of the scene. This section presents a simple approach for doing this, with a view to demonstrating the quality of rendering achieved using a forward transfer approach with second order smoothness priors. The rendering process is described below, followed by a presentation of the results for the four test sequences.

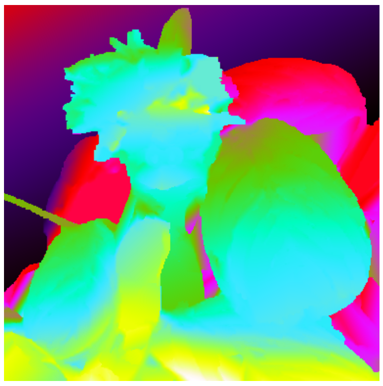
8.4.1 Implementation details

A disparity map for each input image is computed using the framework described in §8.2, using the 2op linear prior. The number of input images used (excluding the reference image) is the same as that used for rendering output views, *i.e.* for the Monkey and Plant sequences $N = 4$, and for the Edmontosaurus sequence $N = 6$.

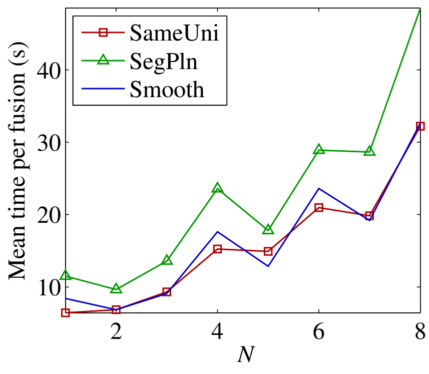
The disparity maps define a regular surface mesh in 3-d space, which is triangulated and texture mapped using the associated input image. Triangles that contain vertices that differ in disparity by more than 10% of the total disparity range of the scene are discarded, thus avoiding connecting two surfaces



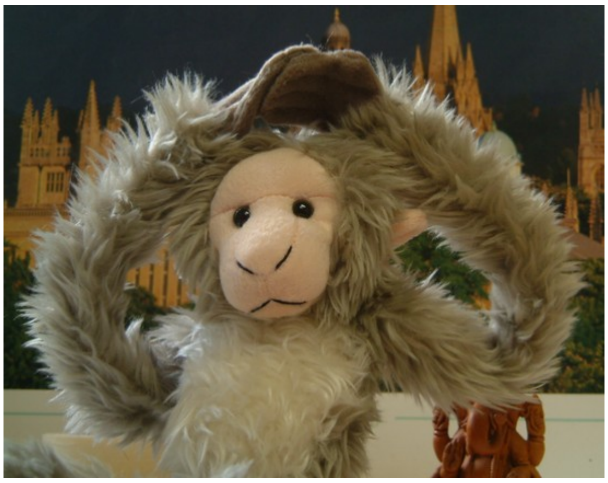
(a) Plant reference image



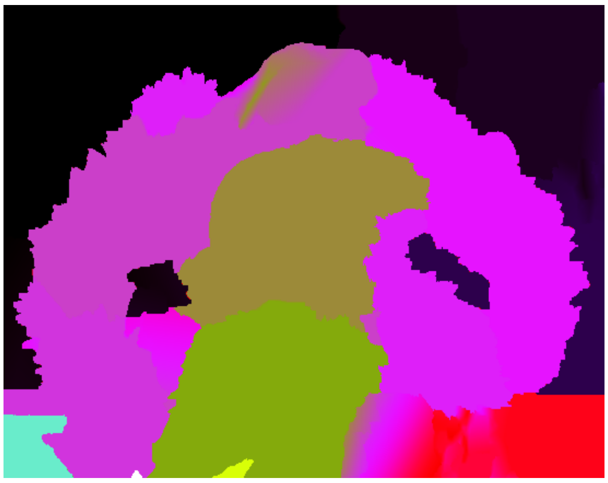
(b) Plant disparity, $N = 2$



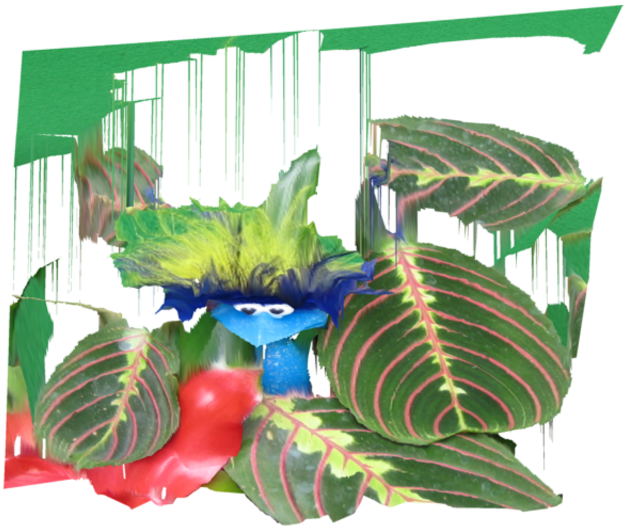
(c) Fusion times *vs.* N



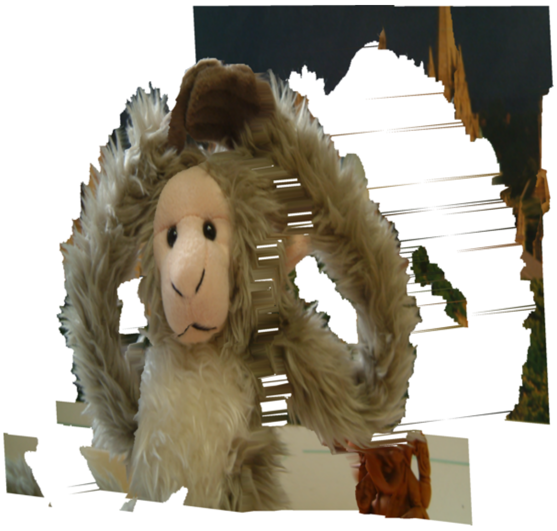
(d) Monkey reference image



(e) Monkey disparity, $N = 4$



(f) Plant pop-out



(g) Monkey pop-out

Figure 8.13: Multiple, arbitrary views. (a) The reference image, $\mathbf{I}_0$ for the Plant sequence, which has arbitrary input views. (b) Output disparity, D, for the Plant sequence computed with $N = 2$, *i.e.* three input images. (c) Graph of mean fusion times for each proposal scheme on the Plant sequence, as a function of N. (d) & (e) $\mathbf{I}_0$ and D computed with $N = 4$ for the Monkey sequence. (f) & (g) Texture-mapped regular meshes generated from $\mathbf{I}_0$ and D, then rotated considerably from the original viewpoint, for the Plant and Monkey sequences respectively.

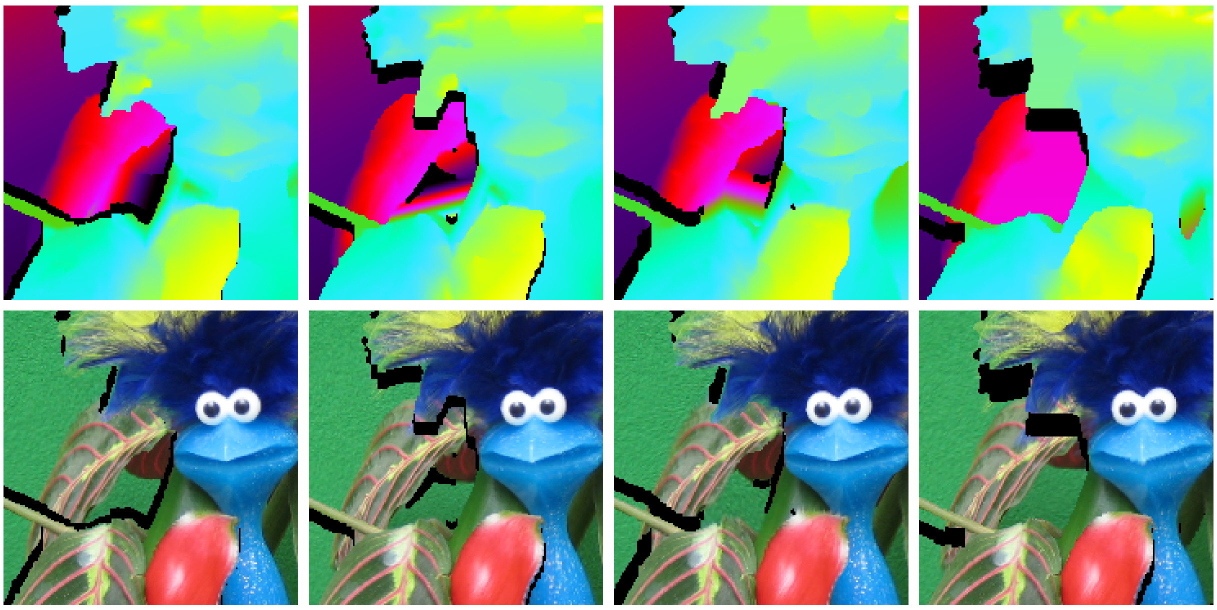


Figure 8.14: New view generation. Intermediate stages for rendering the Plant output image. *Top row*: Input disparity maps reprojected into the new view. *Bottom row*: Input images warped accordingly into the new view. Images are ordered from left to right in terms of their proximity to the new view.

that are discontinuous. The mesh is then projected into the output view using $\mathtt{P}_{i\to 0}$, which is computed using equation A.8 but swapping $\mathtt{P}_0$ and $\mathtt{P}_i$ around. Values at the integer (*i.e.* pixel) locations in the new view are linearly interpolated from projected values, which generally fall on non-integer locations.

The projected disparity maps and images for the Plant sequence can be seen in figure 8.14. The black areas of the images are caused by no points projecting into this area—*i.e.* this part of the scene was not visible in the input image. These areas can be reduced in size by combining the outputs of all the input images in a composition step, which goes as follows: the input images are ordered according to the closeness (in Euclidean distance) of their camera centres to that of the new view; the first image is projected into the new view; any holes (black areas) are filled in by the projected second image, and so on through all the input images.

With all the input image disparity maps precomputed, this rendering process can be extremely fast, especially if modern graphics hardware is used.

8.4.2 Results

The resulting output images generated using this forward transfer approach are shown in figure 8.15, with artifacts highlighted with red ovals and improvements highlighted with blue.

The images show that backward transfer NVS using accurate disparity maps, generated using a second order smoothness prior, can correctly reconstruct the low contrast texture of both the baize background of the Plant sequence and the monkey's face, and also reconstruct heavily occluded texture such as the two areas of leaf highlighted in the Plant image.

However, the down-sides of this approach are that it can leave some areas blank if they have not been seen in any of the input images (see the Plant output), and also that artifacts can appear at discontinuity boundaries, especially when such boundaries are highly detailed, such as around the monkey's fur, and the vertebrae in Dino1.

8.5 Conclusion

This chapter has introduced a powerful framework for optimizing a second order smoothness prior in stereo with geometrical visibility reasoning. It provides a means to combine arbitrary, ad hoc disparity proposals in a reasoned way, minimizing a single objective energy.

I have compared the performance of four different priors within this framework, and demonstrated that the second order prior with a linear truncated kernel, and its complementary optimization framework, produces disparity maps that more accurately reconstruct the scene, especially in low texture or highly slanted regions. I have shown that the algorithm can equally be applied to multi-view stereo with arbitrary camera viewpoints and does so for a computational cost roughly linear in N. In addition I have shown that the resulting disparity maps can be used to render accurate new views, which capture detailed texture from the input views.

This work, in concentrating on the optimization of a second order prior, has paid scant attention to the form of the data likelihood term, the use of a sampling-insensitive measure of photoconsistency, the form of the contrast dependent weighting of the smoothness term (*i.e.* the form of the CRF), the learning of optimal model parameters and the quality of the ad hoc proposals. I expect that improvements in these areas will bring about significant increases in performance. In addition, the optimizer used, QPBO, is relatively new to the field of Computer Vision, and therefore one can expect its performance to increase significantly in the future with the development of further techniques to improve label fixing, to the extent that the artifacts generated using a 2op quadratic prior may be overcome.

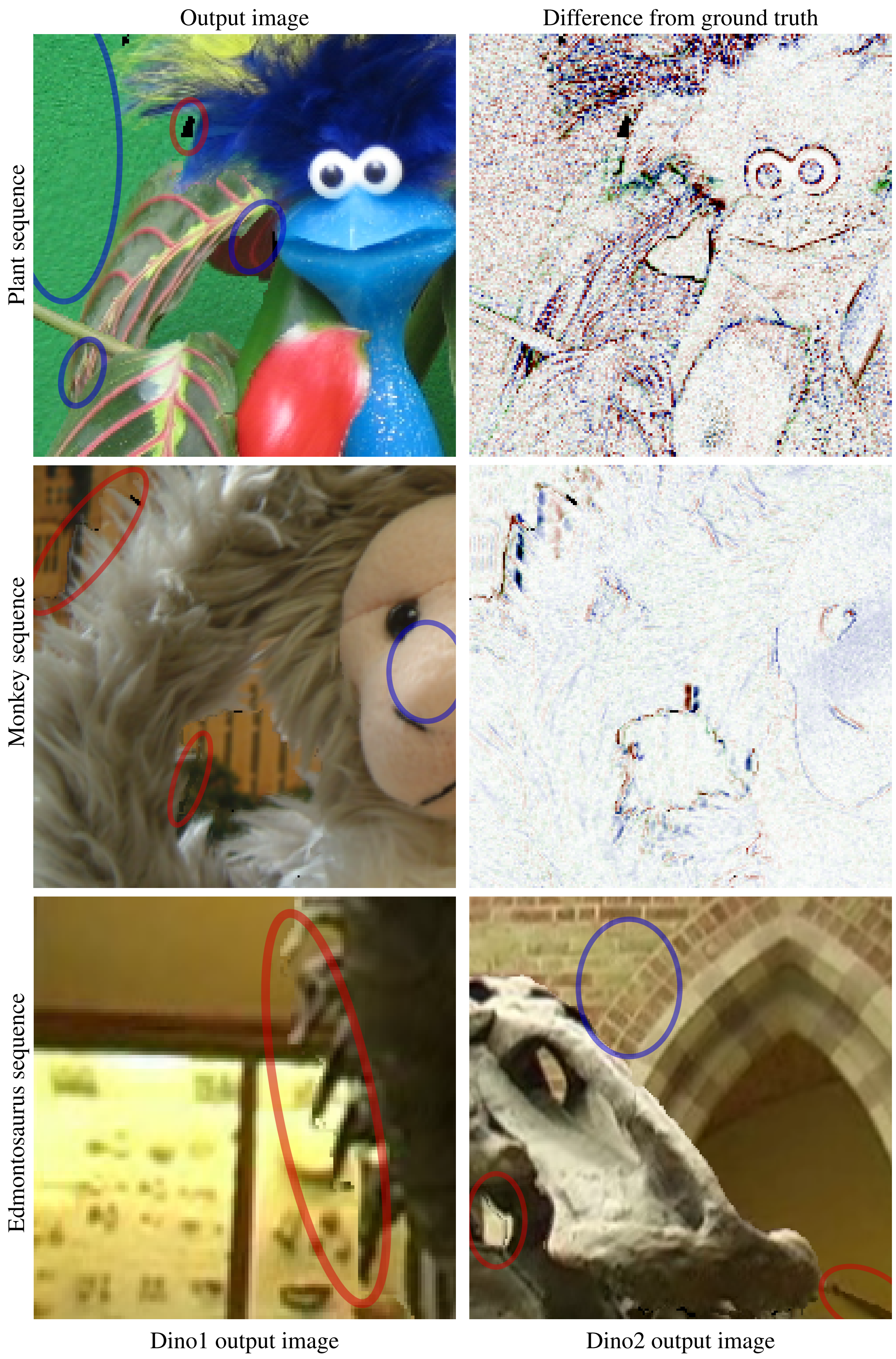


Figure 8.15: Output images. Output images for the four test sequences rendered by warping the input images using the computed disparity maps, with some points of interest highlighted.

Chapter 9

Conclusion

This chapter directly compares the priors for new view synthesis (NVS) investigated in chapters 4–8 of this thesis, summarizes the contributions made, and outlines potential avenues for future research.

9.1 Comparison of results

Chapters 4–7 of this thesis investigated various different forms of prior for backward transfer NVS, while chapter 8 looked at second order smoothness priors for forward transfer NVS, *i.e.* stereo. In this chapter I compare results of the best performing prior and optimization strategy from each of those chapters, which are as follows:

A – a hierarchical, example-based texture prior with 5×5 cliques, optimized in a multi-resolution EM-style framework (from chapter 4).

B – the Field of Experts prior of Roth & Black [RB05] with 5×5 cliques, optimized using ILS with Combined kicks (from chapter 5).

C – the Sampled, example-based texture prior with 2×1 cliques, optimized using TRW-S (from chapter 6).

D – a joint prior with 2×1 cliques, made up of a parametric prior on smoothness modulated with the non-parametric Sampled texture prior, and optimized using QPBOI-R in a fusion move framework (from chapter 7).

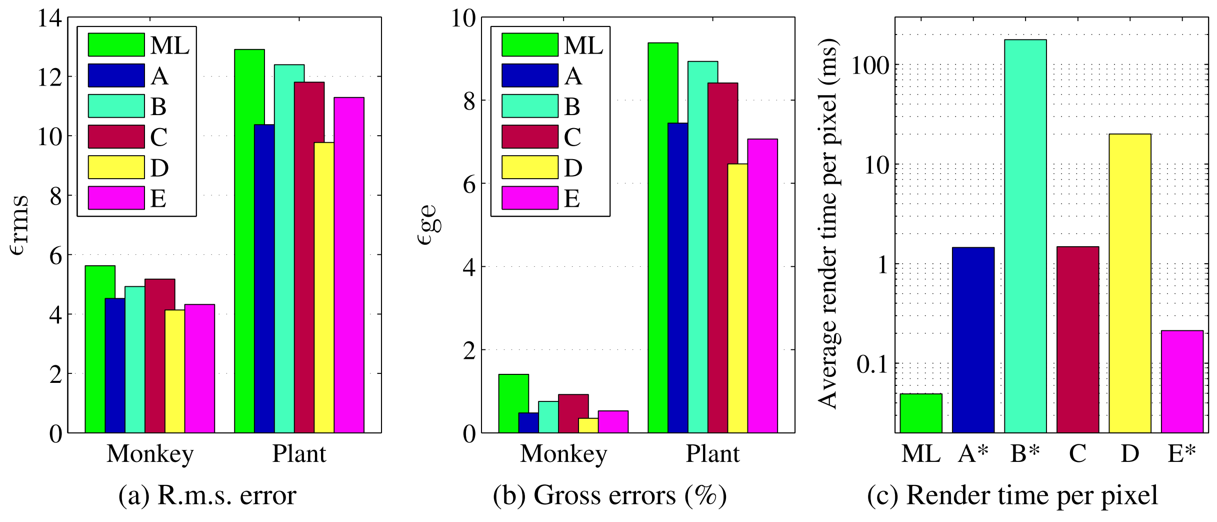


Figure 9.1: Quantitative reconstruction results. (a) R.m.s. error and (b) gross errors for reconstructions of the Monkey and Plant sequences, and (c) rendering time per pixel, averaged over the four test sequences, for the ML solutions and those of the five priors A–E described in §9.1. *One-off model learning/construction times not included.

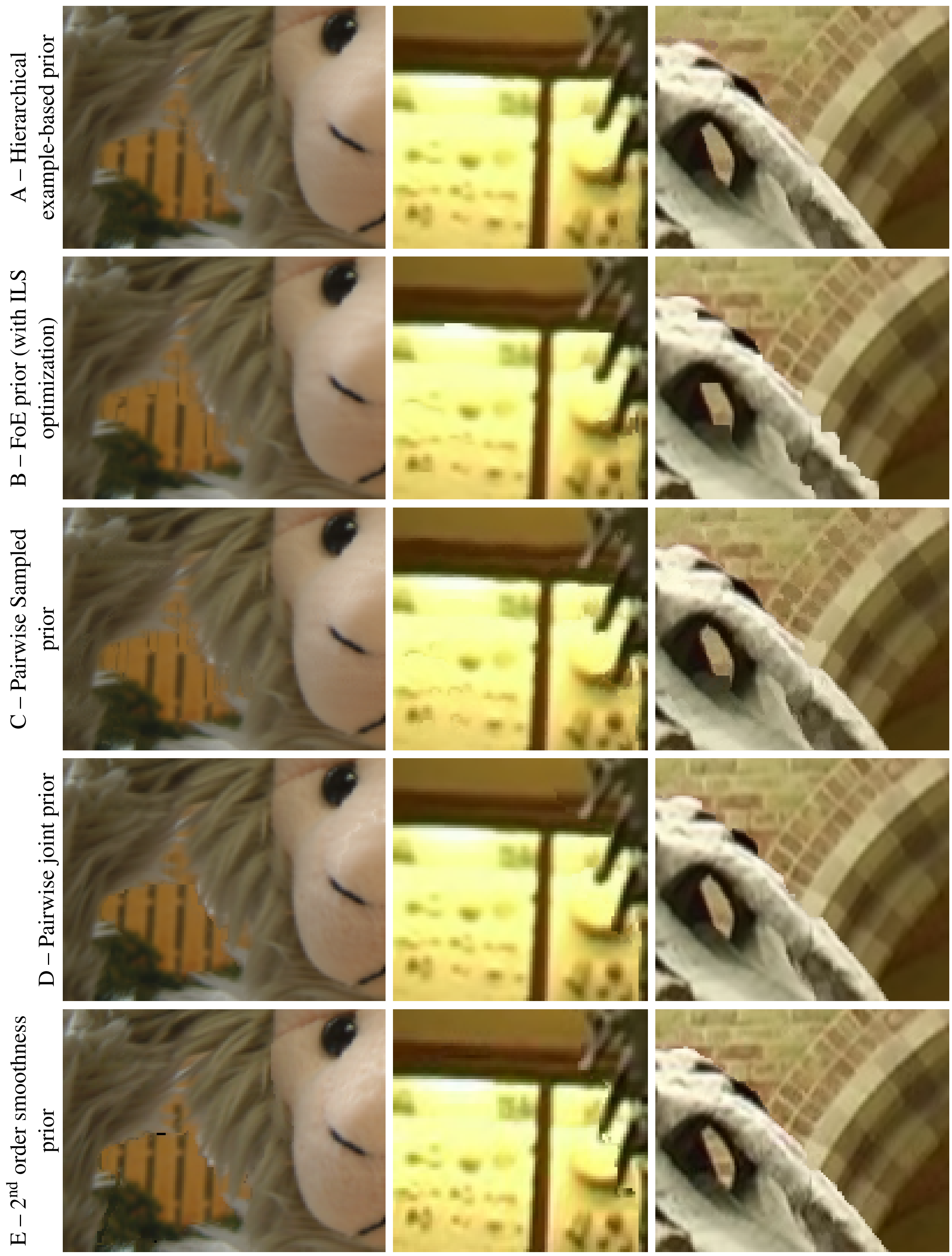
E – a second order smoothness prior (2op) with 3×1 cliques and truncated linear kernel, optimized using QPBOI-R in a fusion move framework, used to generate disparity maps for the input images, which are then projected into the output view (from chapter 8).

A quantitative comparison of the algorithms described above is given in figure 9.1, while zooms of output images highlighting qualitative differences between the algorithms are given in figures 9.2 & 9.3.

9.1.1 Quantitative comparison

Figure 9.1 compares the various algorithms on ϵ_{rms} and ϵ_{ge} for the Monkey and Plant sequences, as well as rendering time. All the priors offer a reduction in error rates over the ML solutions, suggesting that improving the visual plausibility of an output image, or the geometrical plausibility of a depth map, is likely to also make it a more accurate reconstruction of the actual scene. In terms of rendering time, the hierarchical prior (A) and the Sampled prior (C) are roughly the same speed (around 25 times slower than computing the ML solution), with the joint prior (D) an order of magnitude slower, and the FoE prior (B) another order of magnitude slower than that. The 2op prior (E), with no optimization required during rendering, is the fastest method—4 times slower than computing the ML, without hardware acceleration.

For the increase in rendering time, the FoE prior actually offers a decrease in quality over the 2op prior, coming last of the priors in Plant sequence quality, while the Sampled prior comes last for the


A – Hierarchical example-based prior
B – FoE prior (with ILS optimization)
C – Pairwise Sampled prior
D – Pairwise joint prior
E – 2nd order smoothness prior

(a) Monkey zoom (b) Dino1 zoom (c) Dino2 zoom

Figure 9.2: Qualitative results. Zooms of results for three of the four test sequences, using (*row-wise*) the priors A–D described in §9.1.

Monkey sequence. The hierarchical prior is better in some measures, but only the joint prior offers a universal improvement in quality for its slower rendering time, giving the lowest error rates across all measures. There is therefore only a speed/quality trade-off between the hierarchical, joint and 2op priors (and, of course, no prior), but, since the hierarchical and 2op priors require a substantial amount of time to generate the exemplar hierarchy and input disparity maps (one-off costs) respectively, the joint prior wins on both counts if only a handful of images will ever be rendered from a given input sequence.

9.1.2 Partially occluded regions

Partially occluded regions are shown in figure 9.2(a), to the left of the monkey's head, and in figure 9.3(b), beneath the feathers. Of the image priors, in the former case the hierarchical and joint priors reconstruct the texture most successfully, while in the latter case only the joint prior correctly reconstructs the ribs of the leaf, though at the cost of truncating some feathers. The 2op smoothness prior reconstructs the texture as well or better in each case, but for holes left in the images where the scene is deemed to be completely occluded.

9.1.3 Disparity discontinuities

Figure 9.2(b & c) shows some disparity discontinuities at the edge of the dinosaur's vertebrae and snout. On the snout, only the hierarchical prior gives a perfect reconstruction; the Sampled, joint and 2op priors generate a small nick in the snout, while the FoE prior generates a sawtooth edge. Around the vertebrae the 2op prior generates several artifacts, with performance less distinct between the other priors, though the joint prior generates a textural discontinuity along the cabinet frame resulting from an incorrectly located disparity boundary.

9.1.4 Detailed texture

High-frequency texture surrounded by homogenous texture tends to generate multi-modal data costs, with both the correct and surrounding, homogenous colours having a high likelihood. This makes it easy for texture priors to remove high-frequency, or detailed, texture. Examples are the brickwork for the hierarchical prior (figure 9.2(c)), the items in the cabinet in figure 9.2(b) and the stalk in figure 9.3(a) for both the FoE and Sampled priors, as well as a leaf 'rib' in figure 9.3(a) for the FoE prior. Since these artifacts must generate disparity discontinuities, the joint prior framework of chapter 7 is able to avoid them by encouraging surface smoothness, as is the 2op prior of chapter 8.

9.1.5 Low contrast texture

Regions of low contrast texture are shown on the monkey's face in figure 9.2(a) and the baize background of figure 9.3(c). The 2op prior renders both these regions very well, taking, as it does, texture directly from the input images. Of the other priors, only the joint prior successfully renders a portion of the baize background, while none of the priors manage to render the mottled fur on the monkey. This latter result may be due to the input images not being identically exposed in that sequence, such that the variation in colour across images for a given scene point is of a similar order to the variation of colour across the texture.

9.1.6 General *vs*. sequence-specific image priors

General image priors, *i.e.* those learned from a varied set of natural images, were used in chapter 5, in the form of the FoE prior of [RB05], and in chapter 6, in the form of the sparse derivative prior of [TRF03], both of which also happen to be parametric, filter-based priors. Sequence-specific priors, *i.e.* those learned from the input images of the sequence in question, were used in chapter 4, in the form of the example-based prior of [FWZ05], and in chapter 6, again in the form of an example-based prior, but with clique-specific exemplar libraries.

Both the FoE and sparse derivative priors are unimodal, encouraging homogenous texture. As the data costs in NVS tend to be multi-modal, these priors select the colours which give the a piecewise-constant reconstruction, with the effect that details seen in the input sequence, *e.g.* leaf 'ribs' (figure 9.3(a) and figure 6.10), are removed. By contrast, the example-based priors are multi-modal, allowing textured and homogenous patches with equal cost when they both exist in the input sequence. The result is that textured details seen in the input sequence are not removed, and also that the results are much more robust to the strength of the data costs.

The results suggest that sequence-specific priors are preferable for use in NVS. This is to be expected, given that they are more informed as to what the output images should look like than general priors. However, they might prove to be over-specific in situations where the output view is markedly different from input views. Forward transfer NVS has a definite advantage in this case, as no image prior is necessary.

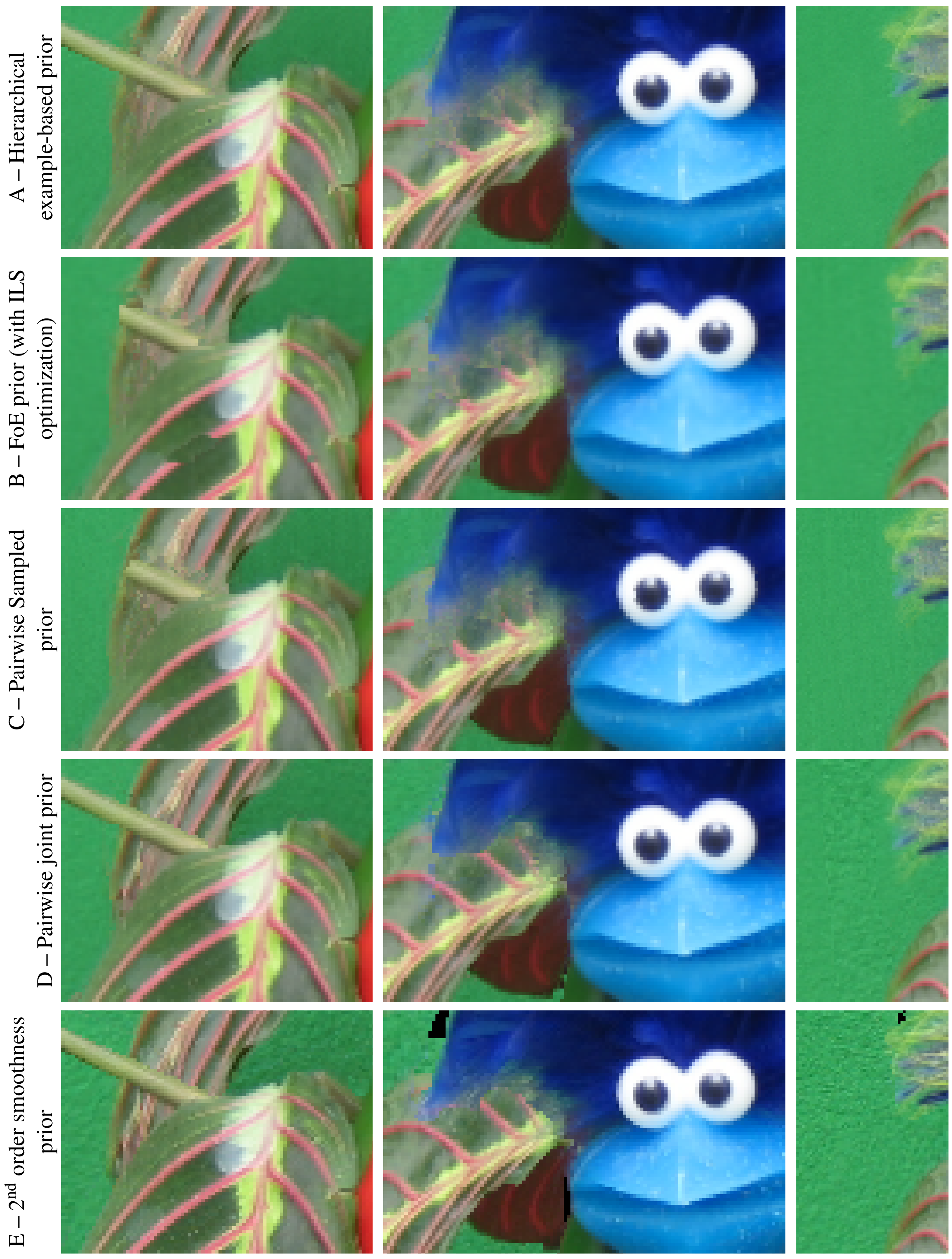

A – Hierarchical example-based prior
B – FoE prior (with ILS optimization)
C – Pairwise Sampled prior
D – Pairwise joint prior
E – 2nd order smoothness prior
(a) Plant zoom 1
(b) Plant zoom 2
(c) Plant zoom 3

Figure 9.3: Plant results. Zooms of the Plant sequence results using (*row-wise*) the priors A–D described in §9.1.

9.2 Contributions

This thesis has introduced several key contributions, as well as some more minor ones, in the areas of prior models for NVS, and optimization. These contributions are listed on a chapter by chapter basis.

9.2.1 Chapter 4

Chapter 4 investigated improvements to the example-based prior framework of Fitzgibbon *et al.* [FWZ05]. Optimization of the objective function was improved over that of [FWZ05] at a single resolution. Three optimization techniques, including ICM and EM-style ICM, were compared, and the EM-style method was found to give the best trade-off between speed and quality of reconstruction.

A novel, hierarchical, exemplar library tree structure was created, and traversed in a multi-resolution framework at a rate of one level per scale, reducing rendering time by a factor of around 30. The multi-resolution framework also improved the image quality by allowing larger scale features to be reconstructed at coarser scales. However, the method failed to reconstruct low contrast texture, and also texture in partially occluded regions.

9.2.2 Chapter 5

Chapter 5 investigated the performance of the general, parametric image prior, Field of Experts [RB05]. Several optimization techniques were tested. Greedy Iterated Conditional Modes (ICM), a novel adaptation of ICM which sequentially updates the node which gives the largest decrease in energy, was introduced, but found to offer no improvement in minimization power over (standard) Raster ICM, while being several times slower. A normalized cooling schedule (appendix B) was introduced for Simulated Annealing (SA) which ensures that iterations of SA are used efficiently. Iterated Local Search (ILS) was used with two novel perturbation methods, and the optimal splice method was developed to combine the resulting local minima in a fusion move framework. While SA was found to minimize the energy best, ILS was found to give the highest quality images.

The FoE prior itself was able to correct the high-frequency artifacts generated in the maximum likelihood images, but was found to remove texture in regions where homogenous texture has a high likelihood (along with the correct texture), and also to create strong vertical and horizontal edges at disparity discontinuities. In addition, it was not able to reconstruct low contrast texture and texture in partially occluded regions, as with the prior of the chapter 4.

9.2.3 Chapter 6

Chapter 6 investigated the performance of a range of pairwise clique image priors, optimized using TRW-S [Kol06]. The number of labels was reduced by minimizing out disparity and using modes over colour, as per [FWZ05]. A fast, deterministic method for finding colour modes (with disparity minimized out), given colour modes at each disparity, was developed. A deterministic method for finding colour modes at a single disparity, given the robust, truncated quadratic kernel used to reject outliers, was also introduced.

A new example-based prior (the Sampled prior) was introduced, which creates discriminative, clique specific exemplar libraries by projecting the output patch into input images at a range of disparities (making the assumption that the patch is fronto-parallel) and sampling. Despite the fronto-parallel assumption, the libraries were found to differentiate well between patches across genuine disparity boundaries and those across incorrect disparity boundaries.

The Sampled prior was shown to produce more accurate reconstructions than other pairwise priors, and the use of a global optimizer was shown to fix large scale errors in the ML images, which the local optimizers of previous chapters failed to correct. However, the prior failed to reconstruct low contrast texture, and the robust kernel used was found to improve rendering in only a few partially occluded regions.

9.2.4 Chapter 7

Chapter 7 used a prior on geometry to improve reconstruction. While not new in itself, several techniques were introduced to incorporate geometrical visibility reasoning and optimize disparity and colour at the same time (to a degree):

- A pairwise graph structure for asymmetrical geometrical visibility reasoning, building on the higher order clique model of [WQ05].
- Modelling colour based on the visibility of up to two input samples, generating triple cliques in the energy.
- Using the triple clique decomposition of [KZ04] for non-submodular cliques.
- Developing QPBO-R and QPBOI-R for improved labelling of nodes left unlabelled by QPBO.

In addition to this, the Sampled prior of the previous chapter was used to modulate the smoothness costs, as a replacement for the CRF terms of stereo methods, in order to encourage disparity boundaries to align with the true edges in the scene.

The resulting joint prior and optimization framework was found to correct even the largest scale errors in the ML image, and correctly reconstruct partially occluded and also low contrast texture. Some artifacts were generated around incorrect disparity discontinuities, but overall the performance of this algorithm was the best tested.

9.2.5 Chapter 8

The focus of chapter 8 was to develop a tractable global optimization framework for a second order smoothness prior. In doing so it used many of the developments introduced in the chapter 7, but additionally provided a framework for combining any set of ad hoc disparity proposals in a reasoned way. Three example methods for generating such proposals were given, one of which novelly used a set of under- to over-segmentations of the reference image to generate a set of piecewise-planar proposals.

Two types of kernel, truncated linear and truncated quadratic, were compared with both first and second order priors. The first order, truncated linear prior was unable to reconstruct planar and curved surfaces, while the other three priors were. Of those, the first-order, truncated quadratic prior was shown to reconstruct slanted textureless surfaces and even highly slanted textured surfaces less accurately than the second order priors. The second order, truncated quadratic prior was shown to present a difficult optimization problem that led to artifacts being generated in the output. However, the second order, truncated linear prior, combined with the optimization framework presented, was shown to improve on stereo results using the first order priors predominantly in use today.

New views generated using the resulting disparity maps, in a forward transfer approach, were shown to compare favourably with those results generated using the backwards transfer approaches, especially in the reconstruction of low contrast texture, but also highlighted the failure modes of the approach—leaving holes in the image and creating artifacts around discontinuity boundaries.

9.3 Future work

In this section I outline possible directions for future research.

9.3.1 New prior models

This thesis has merely touched the surface of prior models available for image and geometry regularization. Some other models, that have either come to light through the research presented here, or there was not time to try, are:

- **Sparse coding prior** – This model, which was discussed in §3.3.3, has not yet been applied to the NVS problem.
- **Triple clique texture prior** – Chapter 6 introduced a pairwise texture prior, while chapter 7 introduced a framework for optimizing energies with triple cliques. An obvious extension would be to create a triple clique texture prior in the same way as the pairwise prior, then optimize the energy using this new framework.
- **Large clique joint priors** – Chapter 7 introduced a joint prior over disparity and texture with pairwise cliques. This idea of combining disparity and texture regularization in the same model could be extended to large clique priors such as the example and filter-based priors of chapters 4 & 5 respectively.
- **Image epitome** – As discussed in §3.3.4, the Image Epitome of Jojic *et al.* [JFK03] provides a means of creating a redundancy-free exemplar library for example-based priors. A hierarchical prior could be constructed using epitomes in exactly the same way as the exemplar hierarchy of chapter 4, making inference even faster.
- **Multi-modal colour filters for FoE** – As discussed in chapter 5, the FoE prior as learned by Roth & Black [RB05] is both unimodal and does not model dependencies between colour channels (in YCbCr colour space). An FoE model that is both multi-modal (as discussed in appendix C) and learned (including the filters) on colour images might overcome some of the issues with the prior, especially if trained specifically on the input sequence. Such a model was learned, with reasonable success, in work I published [WRTF06], but the gradient-descent-based learning process has since proved to be unreliable, possibly due to there being many local minima in parameter space. New, improved learning techniques [WF07] might now make such an endeavour feasible.

9.3.2 Wider variety of output images w.r.t. input sequence

The test sequences used in this thesis have output viewpoints which are a similar distance from the scene, with a similar angle of view and resolution, to the input images, and which interpolate the input images (*i.e.* have input views either side). This has constrained the evaluation of priors to a very limited class of output view. It would be interesting to extend this evaluation to viewpoints which are further towards or away from the scene, which have a different resolution, or which extrapolate the input views.

9.3.3 Temporal consistency

Many applications for NVS involve the rendering of videos from a sequence of stills or a video with a different viewpoint or path. Recent research [SWR07] has shown that if frames are rendered independently, *e.g.* using the methods presented here, the output images can look plausible by themselves, but when played in a video any inter-frame differences in texture become very noticeable. A further avenue of research, which has already been investigated by Shahrokni *et al.* [SWR07], is therefore into prior models which encourage consistency between consecutive frames of video, known as "temporal consistency".

9.3.4 Higher-level cues

By dint of the fact that when looking at a scene through one eye, one is able to estimate depth and imagine with reasonable accuracy what the scene would like from a different angle, it is clear that the human vision system makes use of many higher-level cues, *e.g.* shading, prior knowledge of object sizes, *etc.*, to achieve this. *Single view reconstruction* algorithms, *e.g.* [PZF06, HEH08], are now being developed that make use of such cues. Incorporating such cues into an NVS framework would undoubtedly improve reconstruction, especially in situations where the input data is more sparse.

9.3.5 Towards perfect NVS

As well as the approaches mentioned above, I discuss here what other approaches might be considered to achieve perfect NVS. This thesis has shown that regularizing geometry can improve reconstruction, but that it also tends to generate errors around discontinuity boundaries. One way of improving this situation is to allow semi-transparency around edges, through the use of an alpha-matte, to give better anti-aliasing in these areas. This approach has recently been employed in stereo to good effect [BGRR09]. Another approach, which would also improve texture reconstruction, would be to increase the resolution at which geometry and texture are computed, allowing finer details to be reconstructed. There is already a great deal of literature, *e.g.* [IP91, PCRZ07], in this area, which is known as multi-frame super-resolution.

Appendix A

Projective geometry

A.1 Projection

Input images are assumed to be generated by central projection of light rays onto the image plane—the standard, pinhole camera model. The set of light rays passing through the camera centre are recorded on a planar image sensor some distance from the centre, forming an image. The light ray that was measured at each image pixel can therefore be defined by the line that passes from the centre of the pixel, through the camera centre and out into space.

Let us define a Euclidean coordinate frame for space in the world, which is $\mathbb{R}^3$, such that each point in space is defined by a vector, $\mathbf{X} = [X, Y, Z]^\top$. Now let us assume that a camera centre sits at the origin, $[0, 0, 0]^\top$, and that its image plane is the plane $Z = 1$. Points on the image plane can therefore be parameterized by a 2-d vector, $\mathbf{x} = [x, y]^\top$, with their equivalent coordinates in $\mathbb{R}^3$ defined by $\mathbf{X} = [x, y, 1]^\top = \mathring{\mathbf{x}}$. Now the light ray that was measured at pixel $[x, y]^\top$ can therefore be defined by the set of points $z[x, y, 1]^\top = [xz, yz, z]^\top$, where $-\infty < z < 1$. Conversely, if there is a point in space and one assumes that it lies on a ray passing through the camera centre, such that $[X, Y, Z]^\top = [xz, yz, z]$, it can be seen that $z = Z$, $x = X/Z$ and $y = Y/Z$. This coordinate transformation defines the projection of points in 3-d space onto the 2-d image plane, represented by the projection transform, $\pi(\cdot)$, thus

$$\pi\left(\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}\right) = \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix}. \tag{A.1}$$

A.2 Projection matrices

Clearly our input images are in general viewed from different locations and orientations, such that the parametrization of space described above can only hold for one image at a time. In order to overcome this a coordinate transformation from some reference coordinate frame, "world coordinates", to the coordinate frame of the i^{th} input image is defined. If $\mathbf{X}$ and $\mathbf{X}_i$ are coordinates in these frames respectively, then such a transformation would look like

$$\begin{aligned} \mathbf{X}_i &= \mathtt{R}(\mathbf{X} - \mathbf{C}), & \text{(A.2)} \\ &= \mathtt{R}\,[\mathtt{I}| - \mathbf{C}]\,\mathring{\mathbf{X}}, & \text{(A.3)} \end{aligned}$$

where $\mathtt{R}$ is a 3×3 rotation matrix representing the orientation of the image's coordinate frame, and $\mathbf{C}$ represents the coordinates of the image's camera centre in world coordinates. Together, $\mathtt{R}$ and $\mathbf{C}$ are known as the external camera parameters.

In addition to the external parameters there are internal camera parameters [HZ04, Chapter 6], which relate to the focal length, pixel size, principal point and skew of the camera. These parameters are stored in the calibration matrix $\mathtt{K}$, a 3×3 upper triangular matrix, which further transforms the points in the camera coordinate frame, thus

$$\mathbf{X}_i = \mathtt{KR}\,[\mathtt{I}| - \mathbf{C}]\,\mathring{\mathbf{X}} \tag{A.4}$$

The projection matrix, $\mathtt{P}_i = \mathtt{KR}\,[\mathtt{I}| - \mathbf{C}]$, a 3×4 matrix, therefore defines the full transformation from world coordinates to camera coordinates, thus:

$$\mathbf{X}_i = \mathtt{P}_i\mathring{\mathbf{X}}. \tag{A.5}$$

It is assumed that a projection matrix, $\mathtt{P}_i$, for each input image, I_i, is given as an additional input. These projection matrices can be measured or calculated using various approaches. In the case of this work a *structure from motion* algorithm [FZ98] in commercial software [2d303] is used to calculate the projection matrices from the input sequence. I assume that these projection matrices are noiseless; an alternative approach is to infer both a dense scene reconstruction for NVS *and* the input projection matrices from the input sequence in a single process.

The final input is the projection matrix of our output, or reference, view, $\mathtt{P}_0$, which is generated by the user. One can convert from homogenous world coordinates, $\mathring{\mathbf{X}}$, to homogenous reference camera coordinates, $\mathring{\mathbf{X}}_0$, thus:

$$\mathring{\mathbf{X}}_0 = \begin{bmatrix} \mathtt{P}_0 \\ 0\,0\,0\,1 \end{bmatrix} \mathring{\mathbf{X}}, \tag{A.6}$$

so the reverse transformation is simply

$$\mathring{\mathbf{X}} = \begin{bmatrix} \mathtt{P}_0 \\ 0\,0\,0\,1 \end{bmatrix}^{-1} \mathring{\mathbf{X}}_0. \tag{A.7}$$

Substituting this in for $\mathring{\mathbf{X}}$ in equation A.5 gives us the projection matrix, $\mathtt{P}_{0\to i}$, which transforms coordinates from the reference frame to the *i*th input frame:

$$\mathtt{P}_{0\to i} = \mathtt{P}_i \begin{bmatrix} \mathtt{P}_0 \\ 0\,0\,0\,1 \end{bmatrix}^{-1}. \tag{A.8}$$

A.3 Depth parametrization

The coordinate transformation between cameras described in the previous section requires that point coordinates are known in 3-d. Each pixel has a 2-d location on an image, but it was shown in §A.1 that this location, along with the camera's projection matrix, defines a ray in space; choosing a point along this ray allows this point to be projected into another image.

Let us parameterize the distance along the ray belonging to pixel $\mathbf{x}$ in the reference frame by the distance from the camera centre, perpendicular to the image plane, and call this distance the "depth", z, of the pixel. One can therefore write the position in I_i of any point along that ray as:

$$\pi\left(\mathtt{P}_{0\to i}\,[\mathbf{x}z, z, 1]^\top\right) = \begin{bmatrix} \mathbf{P}^i_1\,[\mathbf{x}z, z, 1]^\top / \mathbf{P}^i_3\,[\mathbf{x}z, z, 1]^\top \\ \mathbf{P}^i_2\,[\mathbf{x}z, z, 1]^\top / \mathbf{P}^i_3\,[\mathbf{x}z, z, 1]^\top \end{bmatrix} \tag{A.9}$$

$$= \begin{bmatrix} \mathbf{P}^i_1\,[\mathbf{x}, 1, z^{-1}]^\top / \mathbf{P}^i_3\,[\mathbf{x}, 1, z^{-1}]^\top \\ \mathbf{P}^i_2\,[\mathbf{x}, 1, z^{-1}]^\top / \mathbf{P}^i_3\,[\mathbf{x}, 1, z^{-1}]^\top \end{bmatrix} \tag{A.10}$$

where

$$\mathrm{P}_{0\to i} = \begin{bmatrix} \mathbf{P}_1^i \\ \mathbf{P}_2^i \\ \mathbf{P}_3^i \end{bmatrix}. \tag{A.11}$$

The transformation from equation A.9 to equation A.10 shows that distance along a ray can just as easily be parameterized by the "disparity"[1], $d = z^{-1}$.

In this work I parameterize distance along a ray by disparity, rather than depth, for two reasons:

Regular samples Consider, without losing generalization to the real world, two 1-d images viewing a 2-d world, with projection from one view to the other given by:

$$\begin{bmatrix} \cos\theta & -\sin\theta & C_1 \\ \sin\theta & \cos\theta & C_2 \end{bmatrix} \approx \begin{bmatrix} 1 & -\theta & C_1 \\ \theta & 1 & C_2 \end{bmatrix} \tag{A.12}$$

where θ is the angle between the optical axes, and $\mathbf{C}$ the translation vector between the two optical centres, of the two images. As the images are close to each other, relative to the scene they are viewing, θ and C_2 are assumed to be small, hence the approximation. A ray cast through a pixel of the first image is projected onto the image plane of the second image as follows:

$$x' = \pi\left(\begin{bmatrix} 1 & -\theta & C_1 \\ \theta & 1 & C_2 \end{bmatrix}\begin{bmatrix} x \\ 1 \\ d \end{bmatrix}\right) \tag{A.13}$$

$$= \frac{x - \theta + C_1 d}{\theta x + 1 + C_2 d} \tag{A.14}$$

For a small C_2, it can be seen that $\partial x'/\partial d \approx$ constant, therefore equally spaced disparities will generate approximately equally spaced image samples. As a result, certain regions of the image will not be over sampled relative to others, so image sampling can be kept to a minimum.

[1]The term "disparity" is traditionally associated with the horizontal displacement of correspondences between stereo image pairs, where it is actually proportional to the inverse of the scene "depth". I generalize its definition to mean inverse depth in all situations.

Correct smoothness costs Planes in space have zero second derivative of disparity, given a projective camera:

$$\begin{aligned}
\begin{bmatrix} n_1 \\ n_2 \\ n_3 \\ n_4 \end{bmatrix} \cdot \begin{bmatrix} xz \\ yz \\ z \\ 1 \end{bmatrix} &= 0 && \text{(A.15)} \\
d = \frac{1}{z} &= -\frac{n_1 x + n_2 y + n_3}{n_4} && \text{(A.16)} \\
\frac{\partial^2 d}{\partial x^2} &= 0, \qquad \frac{\partial^2 d}{\partial y^2} = 0, \qquad \frac{\partial^2 d}{\partial x \partial y} = 0. && \text{(A.17)}
\end{aligned}$$

The set of surfaces that only fulfil the constraints $\frac{\partial^2 d}{\partial x^2} = 0$ and $\frac{\partial^2 d}{\partial y^2} = 0$ are a superset of planar surfaces, which includes any surface whose intersections with the planes passing through each image row and column and the camera centre are straight lines. An example of a non-planar surface in this set is shown in figure 3.3(c).

The full projection function for projecting pixels from $\mathbf{I}_0^*$ into $\mathbf{I}_i$ is therefore defined as

$$\pi_i(\mathbf{x}, d) = \pi\left(\mathrm{P}_{0\rightarrow i}\begin{bmatrix} \mathbf{x} \\ 1 \\ d \end{bmatrix}\right). \tag{A.18}$$

Appendix B

A normalized cooling schedule

This appendix describes how a normalized temperature schedule is translated into an actual temperature for each iteration of Simulated Annealing (SA).

B.1 Problem statement

Let $\mathbf{X} = X_1, .., X_n$ be a vector of discrete variables, where the *i*th variable, X_i, can have an integer value (or label) from 1 to N_{L}^i, and where N_{L}^i, the number of labels per variable, can vary between variables. Let τ be the number of labels over all variables, thus

$$\tau = \sum_{i=1}^{n} N_{\mathrm{L}}^i. \tag{B.1}$$

The labelling, $\mathbf{X}$, has an associated probability, written as a Gibbs distribution, thus

$$p(\mathbf{X}) = \frac{1}{\zeta} \exp\left(-E(\mathbf{X})\right), \tag{B.2}$$

and the aim of SA is to find the most likely $\mathbf{X}$.

The conditional probability of a particular label, X_i, having a value x, given the values of all the other labels, written as $\mathbf{X}_{\not i}$, is given by the expression

$$p(X_i = \mathrm{x}|\mathbf{X}_{\not i}) = \frac{\exp\left(-E(\mathbf{X}_{\not i}, X_i = \mathrm{x})\right)}{\sum_{\mathrm{y}=1}^{N_{\mathrm{L}}^i} \exp\left(-E(\mathbf{X}_{\not i}, X_i = \mathrm{y})\right)}. \tag{B.3}$$

SA sequentially draws a new value for each label, X_i, from this distribution raised to the power T^{-1}, *i.e.*

$$p(X_i = \mathrm{x}|\mathbf{X}_{\not i})^{T^{-1}} = \frac{1}{A_i} \exp\left(-E(\mathbf{X}_{\not i}, X_i = \mathrm{x})/T\right), \tag{B.4}$$

$$A_i = \sum_{\mathrm{y}=1}^{N_\mathrm{L}^i} \exp\left(-E(\mathbf{X}_{\not i}, X_i = \mathrm{y})/T\right) \tag{B.5}$$

where T is called the "temperature", and A_i is the area of the unnormalized conditional probability distribution of X_i, in a process known as Gibbs sampling. Traditionally, samples are drawn using Monte Carlo (MC) methods—a new label is proposed, using some stochastic perturbation function, and accepted based on its likelihood compared to that of the current label according to the Metropolis algorithm [MRR$^+$53]. Specifically, if $e = E(\mathbf{X})$ and $e' = E(\mathbf{X}_{\not i}, X_i = \mathrm{x})$, where x is the proposed label, then the new label is always accepted if $e' < e$, otherwise it is accepted with probability $\exp((e - e')/T)$.

In this work the entire conditional distribution is computed, *i.e.* $\{p(X_i = \mathrm{x}|\mathbf{X}_{\not i})^{T^{-1}}\}_{\mathrm{x}=1}^{N_\mathrm{L}^i}$, and a value drawn randomly from this[1]. This alternative approach increases the computation time of a single iteration, but unlike MC methods it doesn't require a "burn in" period, therefore needs fewer iterations. Furthermore, computing the entire conditional distribution is required for the normalization process introduced here.

A single iteration of SA goes through the entire labelling once, drawing the new labels with a fixed value of T. A high value of T makes the distribution uniform, and as $T \to 0$ the distribution becomes more peaky, until it becomes a delta function on the most likely value. The algorithm is made up of several such iterations, with the temperature gradually being reduced according to some cooling schedule. A record is kept of the highest probability labelling found over the entire process. Generally, the slower the cooling schedule, the better the solution found.

On a typical low-level vision problem a single iteration will take a long time, so it is important to minimize the number of iterations required to find a good solution. The problem is knowing what is a good temperature to start at—too low and the initial labellings will not vary enough to escape from local minima; too high and iterations are wasted—and how fast the temperature should be reduced, both of which depend on the values of $E(\mathbf{X})$. I developed a normalized temperature scale in order to overcome this problem.

[1]This achieves the same as the traditional MC approach after several iterations, with proposals drawn randomly from a uniform distribution over all labels.

B.2 Normalized temperature

Let T_n be a normalized temperature that is on a scale from 0 to 100, such that at $T_\mathrm{n} = 100$ all label values are drawn from uniform distributions, while at $T_\mathrm{n} = 0$ they are drawn from delta functions on the most likely value. This normalized temperature is therefore a function of the shape of the distributions, and in particular how uniform they are.

A good measure of the distribution shape for this purpose is the inverse of the height of the mode—for a (discrete) delta function this will be 1, while for a uniform distribution it will be N_L^i, the number of labels. The sum of this measure over all variables, S, is computed as

$$\begin{aligned} S &= \sum_{i=1}^{n} \frac{A_i}{\max_{\mathrm{y}=1}^{N_\mathrm{L}^i} \exp\left(-E(\mathbf{X}_{\not i}, X_i = \mathrm{y})/T\right)} && \text{(B.6)} \\ &= \sum_{i=1}^{n} \frac{A_i}{\exp\left(-\min_{\mathrm{y}=1}^{N_\mathrm{L}^i} E(\mathbf{X}_{\not i}, X_i = \mathrm{y})/T\right)} && \text{(B.7)} \\ &= \sum_{i=1}^{n} \sum_{\mathrm{x}=1}^{N_\mathrm{L}^i} \exp\left(-\frac{M_\mathrm{x}^i}{T}\right), && \text{(B.8)} \\ M_\mathrm{x}^i &= E(\mathbf{X}_{\not i}, X_i = \mathrm{x}) - \min_{\mathrm{y}=1}^{N_\mathrm{L}^i} E(\mathbf{X}_{\not i}, X_i = \mathrm{y})). && \text{(B.9)} \end{aligned}$$

The normalized temperature, T_n, is converted into a target value for S, written S_n. The normalized temperature scale requires that at $T_\mathrm{n} = 100$, $S_\mathrm{n} = \tau$, *i.e.* all distributions are uniform, and that at $T_\mathrm{n} = 0$, $S_\mathrm{n} = n$, *i.e.* all distributions are delta functions. For temperatures in between, the values of S_n are interpolated on a linear scale, so S_n can be written as the following function of T_n:

$$S_\mathrm{n} = (T_\mathrm{n}/100) \times (\tau - n) + n. \tag{B.10}$$

B.3 Updating T

The temperature, T, needs to be set so that the actual sum, S, matches the desired sum, S_n. As the SA algorithm is iterative, the temperature at time t, T^t, is updated based on the previous values T^{t-1} and S^{t-1}, the latter being computed in the course of the previous iteration, and also the target value S_n^t. S^{t-1}

is given by

$$S^{t-1} = \sum_{i=1}^{n} \sum_{\mathrm{x}=1}^{N_{\mathrm{L}}^{i}} \exp\left(-\frac{M_{\mathrm{x}}^{i}}{T^{t-1}}\right). \tag{B.11}$$

Let us write $T^{t} = T^{t-1} \cdot \delta T$, so that

$$\begin{aligned} S_{\mathrm{n}}^{t} &= \sum_{i=1}^{n} \sum_{\mathrm{x}=1}^{N_{\mathrm{L}}^{i}} \exp\left(-\frac{M_{\mathrm{x}}^{i}}{T^{t-1} \cdot \delta T}\right), && \text{(B.12)} \\ &= \sum_{i=1}^{n} \sum_{\mathrm{x}=1}^{N_{\mathrm{L}}^{i}} \exp\left(-\frac{M_{\mathrm{x}}^{i}}{T^{t-1}}\right)^{\delta T^{-1}}, && \text{(B.13)} \\ &= \sum_{i=1}^{n} \sum_{\mathrm{x}=1}^{N_{\mathrm{L}}^{i}} \exp\left(-\frac{M_{\mathrm{x}}^{i}}{T^{t-1}}\right) \cdot \exp\left(-\frac{M_{\mathrm{x}}^{i}}{T^{t-1}}\right)^{\delta T^{-1}-1}. && \text{(B.14)} \end{aligned}$$

I now approximate the above expression by replacing the actual value of the second exponential in equation B.14 with the average value in each case. Specifically,

$$\begin{aligned} S_{\mathrm{n}}^{t} &\approx \sum_{i=1}^{n} \sum_{\mathrm{x}=1}^{N_{\mathrm{L}}^{i}} \exp\left(-\frac{M_{\mathrm{x}}^{i}}{T^{t-1}}\right) \cdot \left(\frac{1}{\tau} \sum_{i=1}^{n} \sum_{\mathrm{x}=1}^{N_{\mathrm{L}}} \exp\left(-\frac{M_{\mathrm{x}}^{i}}{T^{t-1}}\right)\right)^{\delta T^{-1}-1}, && \text{(B.15)} \\ &\approx \left(\frac{S^{t-1}}{\tau}\right)^{\delta T^{-1}-1} \cdot S^{t-1}. && \text{(B.16)} \end{aligned}$$

Some simple algebra then gives

$$\begin{aligned} \delta T^{-1} &\approx \frac{\log S_{\mathrm{n}}^{t} - \log S^{t-1}}{\log S^{t-1} - \log \tau} + 1, && \text{(B.17)} \\ &\approx \frac{\log S_{\mathrm{n}}^{t} - \log \tau}{\log S^{t-1} - \log \tau}. && \text{(B.18)} \end{aligned}$$

The above approximation is then used to compute the actual temperature used at each iteration, thus

$$T^{t} = T^{t-1} \cdot \frac{\log S^{t-1} - \log \tau}{\log S_{\mathrm{n}}^{t} - \log \tau}. \tag{B.19}$$

Appendix C

Multi-modal Product of Experts

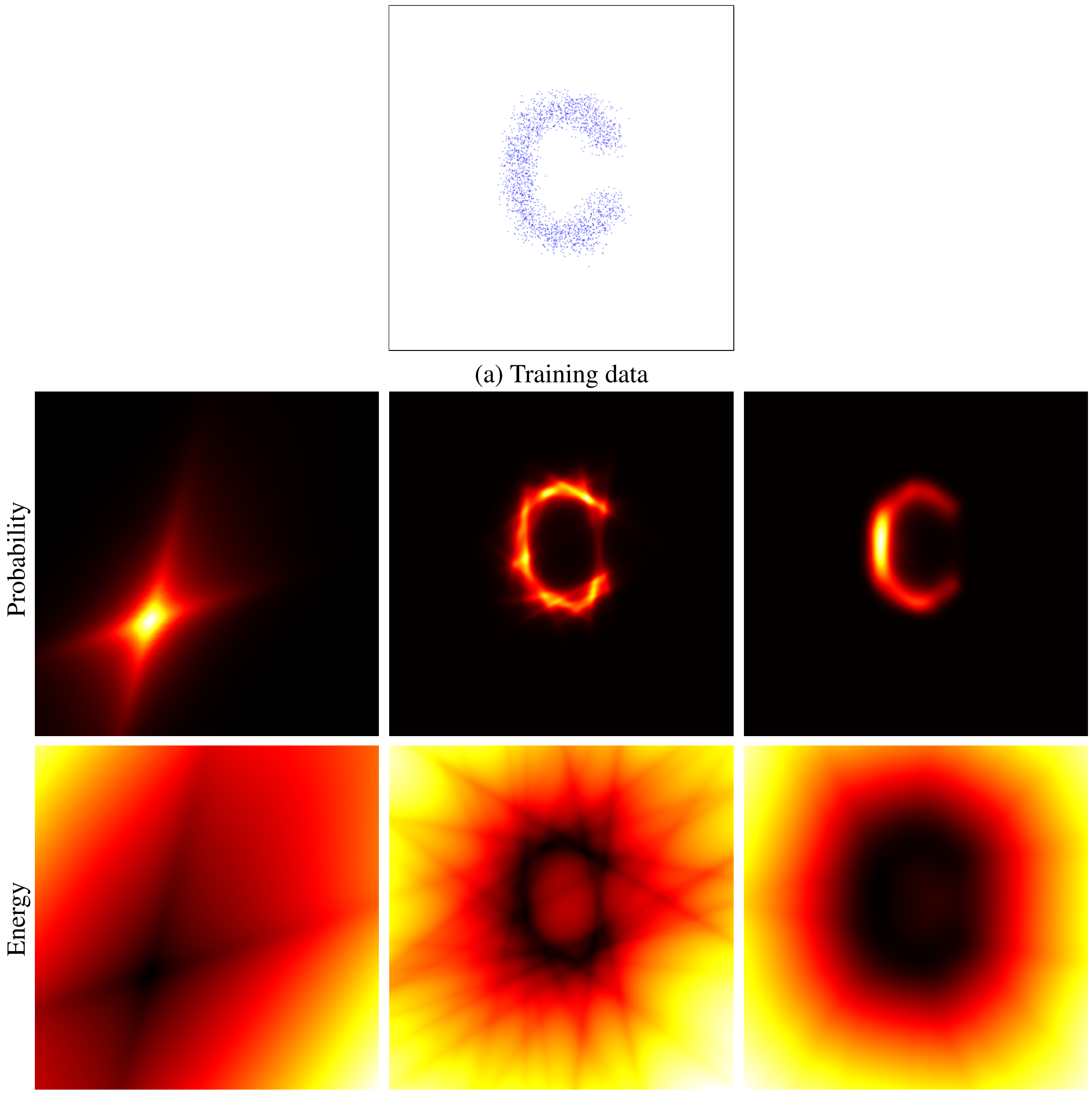
Chapter 5 identified a drawback of the Fields of Experts model of Roth & Black [RB05] being its uni-modal nature. Indeed, any distribution modelled by the product of Student's t-distributions of equation 3.9 will suffer from this problem. In [WRTF06] I introduced a modification to this model which allowed it capture multi-modal distributions, as this appendix will now show.

The original product of t-distributions model, as introduced in [WHO02], referred to here as the "standard" model, models the probability of a data vector, $\mathbf{X}$, as

$$p\left(\mathbf{X}\right) = \frac{1}{\zeta}\prod_{m=1}^{M}\left(1+\frac{1}{2}\left(\mathbf{J}_m^\top\mathbf{X}\right)^2\right)^{-\alpha_m}. \tag{C.1}$$

Let us take a toy, 2-d problem, where the probability distribution to be modelled is the shape of a 'C', as shown by the plot of the training data in figure C.1(a); the data is offset above and to the right of the origin. The model parameters, $\Theta = \{\mathbf{J}_m, \alpha_m\}_{m=1}^{M}$ ($M = 40$ here), are learned by seeking to maximize the likelihood of the training data using contrastive divergence [Hin02]. The learned standard model generates a unimodal distribution centred on the origin, as shown in figure C.1(a), producing a poor reproduction of the original distribution.

The term $\mathbf{J}_m^\top\mathbf{X}$ in equation C.1 can be interpreted as the perpendicular distance of $\mathbf{X}$ from a hyperplane which passes through the origin and whose normal is defined by $\mathbf{J}_m$. It is the constraint that hyperplanes pass through the origin that makes the resulting distribution unimodal. One can allow experts to use any hyperplane by concatenating the normal vector with an offset value, so that it's dimensionality becomes $|\mathbf{X}| + 1$, and homogenizing the data vector, thus: $\mathring{\mathbf{X}} = [\mathbf{X}^\top 1]^\top$. The resulting model, referred



(a) Training data

(b) Standard model (c) Offset model (d) Quadratic model

Figure C.1: A toy problem. (a) Training data drawn randomly from a toy 2-d probability distribution, a PoE model of which is then learned from said data. (b–d) The probability and energy distributions of the learned models given by equations C.1, C.2 & C.3 respectively (black is low, white is high).

to here as the "offset" model, is written as

$$p(\mathbf{X}) = \frac{1}{\zeta}\prod_{m=1}^{M}\left(1+\frac{1}{2}\left(\mathbf{J}_m^\top \mathring{\mathbf{X}}\right)^2\right)^{-\alpha_m}. \tag{C.2}$$

Because experts in this model need not pass through the origin, distributions learned using this model can be multi-modal, as demonstrated in figure C.1(c) (again a 40 expert model, learned using contrastive divergence).

Experts thus far have used linear functions of $\mathbf{X}$ within each t-distribution. It may be the case that the shape of the underlying manifold is better modelled by non-linear experts. Figure C.1(d) demonstrates the output using a 40 quadratic experts model (again learned using contrastive divergence), referred to here as the "quadratic" model, given by the equation below:

$$p\left(\mathbf{X}\right) = \frac{1}{\zeta} \prod_{m=1}^{M} \left(1 + \frac{1}{2}\left(\mathring{\mathbf{X}}^{\top} \mathtt{J}_m \mathring{\mathbf{X}}\right)^2\right)^{-\alpha_m}. \tag{C.3}$$

While the offset and quadratic models have an improved ability to learn complex distributions, a drawback of these multi-modal models is that they create many more local maxima of likelihood of the training data in parameter space, making the gradient-descent-based contrastive divergence approach to parameter learning find very poor parameters as a result, especially in the higher dimensional space of 5×5 image patches.

Bibliography

[2d303] 2d3 Ltd. Boujou: Automated camera tracking, 2003. `http://www.2d3.com`.

[ADSW02] A. Alvarez, R. Deriche, J. Sánchez, and J. Weickert. Dense disparity map estimation respecting image discontinuities: A PDE and scale-space based approach. *Journal of Visual Communication and Image Representation*, 13(1):3–21, March 2002.

[AEB06] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, November 2006.

[AKT08] K. Alahari, P. Kohli, and P. H. S. Torr. Reduce, reuse & recycle: Efficiently solving multi-label MRFs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, Alaska*, 2008.

[AMN$^+$98] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM*, 45(6):891–923, 1998.

[Bad86] R. Baddeley. Searching for filters with 'interesting' output distributions: an uninteresting direction to explore? *Network: Computation in Neural Systems*, 7(2):409–421, February 1986.

[Bar87] S. Barnard. A stochastic approach to stereo vision. In *Readings in computer vision: issues, problems, principles, and paradigms*, pages 21–25. Morgan Kaufmann Publishers Inc., 1987.

[Bel96] P. N. Belhumeur. A Bayesian approach to binocular steropsis. *International Journal of Computer Vision*, 19(3):237–260, August 1996.

[Bes74] J. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society, Series B (Methodological)*, 36(2):192–236, 1974.

[Bes86] J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society, Series B (Methodological)*, 48(3):259–302, 1986.

[BG04] M. Bleyer and M. Gelautz. A layered stereo algorithm using image segmentation and global visibility constraints. In *Proceedings of the IEEE International Conference on Image Processing*, volume 5, pages 2997–3000, October 2004.

[BG07] M. Bleyer and M. Gelautz. Graph-cut-based stereo matching using image segmentation with symmetrical treatment of occlusions. *Signal Processing: Image Communication*, 2:127–143, February 2007.

[BGRR09] M. Bleyer, M. Gelautz, C. Rother, and C. Rhemann. A stereo approach that handles the matting problem via image warping. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Miami*, June 2009.

[BHT06] E. Boros, P. L. Hammer, and G. Tavares. Preprocessing of unconstrained quadratic binary optimization. Technical Report RRR 10-2006, Rutgers Center for Operations Research, April 2006.

[Bis06] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[BJ89] A. Billionnet and B. Jaumard. A decomposition method for minimizing quadratic pseudo-boolean functions. *Operations Research Letters*, 8(3):161–163, June 1989.

[BJ01] Y. Y. Boykov and M. P. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In *Proceedings of the 8th International Conference on Computer Vision, Vancouver, Canada*, volume 2, pages 105–112, 2001.

[BK04] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, September 2004.

[BR96] M. J. Black and A. Rangarajan. On the unification of line processes, outlier rejection, and robust statistics with applications in early vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(1):57–91, July 1996.

[BSA98] S. Baker, R. Szeliski, and P. Anandan. A layered approach to stereo reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Santa Barbara*, pages 434–441, 1998.

[BT99] S. Birchfield and C. Tomasi. Multiway cut for stereo and motion with slanted surfaces. In *Proceedings of the 7th International Conference on Computer Vision, Kerkyra, Greece*, pages 489–495, September 1999.

[BV06] Y. Boykov and O. Veksler. Graph cuts in vision and graphics: Theories and applications. In *The Handbook of Mathematical Models in Computer Vision*. Springer, 2006.

[BVZ01] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001.

[BZ87] A. Blake and A. Zisserman. *Visual Reconstruction*. MIT Press, Cambridge, USA, August 1987.

[CB04] A. Criminisi and A. Blake. The SPS algorithm: patching figural continuity and transparency by split-patch search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Washington, DC*, volume 1, pages 342–349, June 2004.

[CD06] O. Cordón and S. Damas. Image registration with iterated local search. *Journal of Heuristics*, 12(1):73–94, March 2006.

[CG06] D. Cremers and L. Grady. Statistical priors for efficient combinatorial optimization via graph cuts. In *Proceedings of the 9th European Conference on Computer Vision, Graz, Austria*, pages 263–274, 2006.

[CM02] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[CMO04] Ondřej Chum, Jiří Matas, and Štěpán Obdržálek. Enhancing RANSAC by generalized model optimization. In *Proceedings of the Asian Conference on Computer Vision*, volume 2, pages 812–817, January 2004.

[CPT03] A. Criminisi, P. Perez, and K. Toyama. Object removal by exemplar-based inpainting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Madison, Wisconsin*, volume 2, pages 721–728, June 2003.

[CSB$^+$07] A. Criminisi, J. Shotton, A. Blake, C. Rother, and P. H. S. Torr. Efficient dense stereo with occlusions for new view-synthesis by four-state dynamic programming. *International Journal of Computer Vision*, 71(1):89–110, January 2007.

[CTCS00] J.-X. Chai, X. Tong, S.-C. Chan, and H.-Y. Shum. Plenoptic sampling. In *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, pages 307–318, 2000.

[DLR77] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39 B:1–38, 1977.

[EA06] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745, December 2006.

[EL99] A. Efros and T. Leung. Texture synthesis by non-parametric sampling. In *Proceedings of the 7th International Conference on Computer Vision, Kerkyra, Greece*, pages 1039–1046, September 1999.

[Elk03] C. Elkan. Using the triangle inequality to accelerate k-means. In *Proceedings of the 20th International Conference on Machine Learning, Washington DC, USA*, pages 147–153, 2003.

[FD05] D. Freedman and P. Drineas. Energy minimization via graph cuts: Settling what is possible. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego*, volume 2, pages 3939–3946, 2005.

[FH04a] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient belief propagation for early vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 261–268, 2004.

[FH04b] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, September 2004.

[FH06] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient belief propagation for early vision. *International Journal of Computer Vision*, 70(1):41–54, 2006.

[Fie87] D. J. Field. Relations between the statistics of natural images and the response properties of cortical cells. *Journal of the Optical Society of America*, 4(12):2379–2394, 1987.

[FJP02] W. T. Freeman, T. R. Jones, and E. C. Pasztor. Example-based super-resolution. *IEEE Computer Graphics and Applications*, 22(2):56–65, March/April 2002.

[FK98] O. D. Faugeras and R. Keriven. Variational principles, surface evolution, PDE's, level set methods, and the stereo problem. *IEEE Transactions on Image Processing*, 7(4):336–344, March 1998.

[FPC00] W. Freeman, E. C. Pasztor, and O. T. Carmichael. Learning low-level vision. In *International Journal of Computer Vision*, volume 40, pages 25–47, October 2000.

[FR56] L. R. Ford and Fulkerson D. R. Maximal flow through a network. *Canadian Journal of Mathematics*, 8:399–404, 1956.

[FSH$^+$06] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman. Removing camera shake from a single photograph. In *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, 2006.

[Fua93] P. Fua. A parallel stereo algorithm that produces dense depth maps and preserves image features. *Machine Vision Applications*, 6(1), 1993.

[FWZ05] A. Fitzgibbon, Y. Wexler, and A. Zisserman. Image-based rendering using image-based priors. *International Journal of Computer Vision*, 63(2):141–151, July 2005.

[FZ98] A. W. Fitzgibbon and A. Zisserman. Automatic camera recovery for closed or open image sequences. In *Proceedings of the European Conference on Computer Vision*, pages 311–326. Springer-Verlag, June 1998.

[Gen88] M. A. Gennert. Brightness-based stereo matching. In *Proceedings of the 2nd International Conference on Computer Vision, Tampa*, pages 139–143, December 1988.

[GG84] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741, November 1984.

[GLY95] D. Geiger, B. Ladendorf, and A. Yuille. Occlusions and binocular stereo. *International Journal of Computer Vision*, 14:211–226, 1995.

[GM85] A. Gagalowicz and S. D. Ma. Sequential synthesis of natural textures. *Computer Vision, Graphics and Image Processing*, 30:289–315, 1985.

[GP87] E. Gamble and T. Poggio. Visual integration and detection of discontinuities: The key role of intensity edges. Technical Report AI Memo No. 970, MIT Artificial Intelligence Lab., October 1987.

[GPS89] D. M. Greig, B. T. Porteous, and A. H. Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society*, 51(2):271–279, 1989.

[GR92] D. Geman and G. Reynolds. Constrained restoration and the recovery of discontinuities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(3):367–383, 1992.

[Gri81] W. E. L. Grimson. *From Images to Surfaces: A Computational Study of the Human Early Visual System.* MIT Press, 1981.

[GS05] P. Gargallo and P. Sturm. Bayesian 3D modeling from images using multiple depth maps. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego*, volume 2, pages 885–891, June 2005.

[Had02] J. Hadamard. Sur les problèmes aux dérivées partielles et leur signification physique. *Princeton University Bulletin*, pages 49–52, 1902.

[HC04] L. Hong and G. Chen. Segment-based stereo matching using graph cuts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Washington, DC*, pages 74–81, 2004.

[HD91] Y. Hu and T. J. Dennis. MAP estimation in image restoration by a local search enhanced genetic algorithm. In *Proccedings of the 6th International Conference on Digital Processing of Signals in Communications*, pages 123–128, September 1991.

[HEH08] D. Hoiem, A. A. Efros, and M. Hebert. Closing the loop on scene interpretation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, Alaska*, 2008.

[HHS84] P. L. Hammer, P. Hansen, and B. Simeone. Roof duality, complementation and persistency in quadratic 0-1 optimization. *Mathematical Programming*, 28:121–155, 1984.

[Hin99] G. E. Hinton. Products of experts. In *Proceedings of the Ninth International Conference on Artificial Neural Networks, Edinburgh, Scotland*, pages 1–6, 1999.

[Hin02] G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002.

[HJO$^+$01] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin. Image analogies. In *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, pages 327–340, 2001.

[Hor86] B. K. P. Horn. *Robot Vision.* MIT Press, Cambridge MA, 1986.

[Hyv05] A. Hyvärinen. Estimation of non-normalized statistical models by score matching. In *Journal of Machine Learning Research*, pages 695–709, 2005.

[HZ04] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[IG98] H. Ishikawa and D. Geiger. Occlusions, discontinuities, and epipolar lines in stereo. In *Proceedings of the 5th European Conference on Computer Vision, Freiburg, Germany*, pages 232–248, 1998.

[IG06] H. Ishikawa and D. Geiger. Rethinking the prior model for stereo. In *Proceedings of the 9th European Conference on Computer Vision, Graz, Austria*, pages 526–537, 2006.

[IHA02] M. Irani, T. Hassner, and P. Anandan. What does the scene look like from a scene point? In *Proceedings of the 7th European Conference on Computer Vision, Copenhagen, Denmark*, 2002.

[IP91] M. Irani and S. Peleg. Improving resolution by image registration. *Graphical Models and Image Processing*, 53:231–239, 1991.

[Ish03] H. Ishikawa. Exact optimization for Markov random fields with convex priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1333–1336, October 2003.

[JFK03] N. Jojic, B. J. Frey, and A. Kannan. Epitomic analysis of appearance and shape. In *Proceedings of the 9th International Conference on Computer Vision, Nice, France*, pages 34–41, 2003.

[KDM02] S. L. Kilthau, M. S. Drew, and T. Möller. Full search content independent block matching based on the fast Fourier transform. In *Proceedings of the IEEE International Conference on Image Processing*, volume 1, pages 669–672, 2002.

[KEBK05] V. Kwatra, I. Essan, A. Bobick, and N. Kwatra. Texture optimization for example-based synthesis. In *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, volume 24, pages 795–802, July 2005.

[KFCO$^+$07] J. Kopf, C.-W. Fu, D. Cohen-Or, O. Deussen, D. Lischinski, and T.-T. Wong. Solid texture synthesis from 2D exemplars. In *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, volume 26, August 2007.

[KFL01] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, February 2001.

[KGV83] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[KKT07] P. Kohli, M. P. Kumar, and P. H. S. Torr. $\mathcal{P}^3$ & beyond: Solving energies with higher order cliques. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis*, 2007.

[KLT08] P. Kohli, L. Ladický, and P. H. S. Torr. Robust higher order potentials for enforcing label consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, Alaska*, 2008.

[Koh07] P. Kohli. *Minimizing Dynamic and Higher Order Energy Functions using Graph Cuts*. PhD thesis, Oxford Brookes University, November 2007.

[Kol05] V. Kolmogorov. Tree-reweighted message passing algorithm for energy minimization – C++ implementation, 2005. `http://research.microsoft.com/research/downloads/download.aspx?FUID={D814ABFB-FC1F-47E6-B26F-DE896C374B41}`.

[Kol06] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10):1568–1583, October 2006.

[Kol07] V. Kolmogorov. Discrete MRF optimization software, November 2007. `http://www.adastral.ucl.ac.uk/~vladkolm/software.html`.

[KPT07] N. Komodakis, N. Paragios, and G. Tziritas. MRF optimization via dual decomposition: Message-passing revisited. In *Proceedings of the 11th International Conference on Computer Vision, Rio de Janeiro, Brazil*, 2007.

[KR06a] V. Kolmogorov and C. Rother. Comparison of energy minimization algorithms for highly connected graphs. In *Proceedings of the 9th European Conference on Computer Vision, Graz, Austria*, volume 2, pages 1–15, 2006.

[KR06b] V. Kolmogorov and C. Rother. Minimizing non-submodular functions with graph cuts - a review. Technical Report MSR-TR-2006-100, Microsoft Research, 2006.

[KS99] K. Kutulakos and S. Seitz. A theory of shape by space carving. In *Proceedings of the 7th International Conference on Computer Vision, Kerkyra, Greece*, pages 307–314, 1999.

[KSE$^+$03] V. Kwatra, A. Schödl, I. Essan, G. Turk, and A. Bobick. Graphcut textures: Image and video synthesis using graph cuts. In *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, 2003.

[KSK06] A. Klaus, M. Sormann, and K. Karner. Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In *Proceedings of the International Conference on Pattern Recognition*, pages 15–18, 2006.

[KSR$^+$08] P. Kohli, A. Shekhovtsov, C. Rother, V. Kolmogorov, and P. H. S. Torr. On partial optimality in multilabel MRFs. In *Proceedings of the 25th International Conference on Machine Learning, Helsinki, Finland*, 2008.

[KT05] P. Kohli and P. H. S. Torr. Efficiently solving dynamic Markov Random Fields using graph cuts. In *Proceedings of the 10th International Conference on Computer Vision, Beijing, China*, pages 922–929, 2005.

[KTP07] N. Komodakis, G. Tziritas, and N. Paragios. Fast, approximately optimal solutions for single and dynamic MRFs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis*, 2007.

[KZ01] V. Kolmogorov and R. Zabih. Computing visual correspondence with occlusions via graph cuts. In *Proceedings of the 8th International Conference on Computer Vision, Vancouver, Canada*, pages 508–515, 2001.

[KZ02] V. Kolmogorov and R. Zabih. Multi-camera scene reconstruction via graph cuts. In *Proceedings of the European Conference on Computer Vision*, volume 3, page 82, 2002.

[KZ04] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):147–159, 2004.

[KZG03] V. Kolmogorov, R. Zabih, and S. Gortler. Generalized multi-camera scene reconstruction using graph cuts. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, Lecture Notes in Computer Science, pages 285–300. Springer-Verlag, September 2003.

[LH96] M. Levoy and P. Hanrahan. Light field rendering. In *SIGGRAPH96*, 1996.

[LH05] S. Lefebvre and H. Hoppe. Parallel controllable texture synthesis. In *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, pages 777–786, 2005.

[LH08] Y. Li and D. Huttenlocher. Sparse long-range random field and its application to image denoising. In *Proceedings of the 10th European Conference on Computer Vision, Marseille, France*, 2008.

[LL08] W. Li and B. Li. Joint Conditional Random Field of multiple views with online learning for image-based rendering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, Alaska*, 2008.

[LLX$^+$01] L. Liang, C. Liu, Y.-Q. Xu, B. Guo, and H.-Y. Shum. Real-time texture synthesis by patch-based sampling. *ACM Trans. Graph.*, 20(3):127–150, 2001.

[LMP01] J. Lafferty, A. McCallum, and F. Pereira. Conditional Random Fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning, Williamstown, USA*, pages 282–289, 2001.

[LMS02] H. R. Lourenço, O. C. Martin, and T. Stützle. Iterated local search. In *Handbook of Metaheuristics*, pages 321–353. Kluwer, 2002.

[LRB07] V. Lempitsky, C. Rother, and A. Blake. LogCut – efficient graph cut optimization for Markov Random Fields. In *Proceedings of the 11th International Conference on Computer Vision, Rio de Janeiro, Brazil*, 2007.

[LRHB06] X. Lan, S. Roth, D. P. Huttenlocher, and M. J. Black. Efficient belief propagation with learned higher-order Markov Random Fields. In *Proceedings of the 9th European Conference on Computer Vision, Graz, Austria*, pages 269–282, 2006.

[LS88] S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society, Series B (Methodological)*, 50(2):157–224, 1988.

[LW04] A. Levin and Y. Weiss. User assisted separation of reflections from a single image using a sparsity prior. In *Proceedings of the 8th European Conference on Computer Vision, Prague, Czech Republic*, pages 602–613, 2004.

[LZ06] G. Li and S. W. Zucker. Surface geometric constraints for stereo in belief propagation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, New York*, pages 2355–2362, 2006.

[LZW02] A. Levin, A. Zomet, and Y. Weiss. Learning to perceive transparency from the statistics of natural scenes. In *Advances in Neural Information Processing Systems*, pages 1247–1254, 2002.

[LZW03] A. Levin, A. Zomet, and Y. Weiss. Learning how to inpaint from global image statistics. In *Proceedings of the 9th International Conference on Computer Vision, Nice, France*, pages 305–312, 2003.

[Mac03] D. Mackay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003.

[MCSF06] J. J. McAuley, T. S. Caetano, A. J. Smola, and M. O. Franz. Learning high-order MRF priors of color images. In *Proceedings of the 23rd International Conference on Machine Learning, Pittsburgh, USA*, pages 617–624, 2006.

[MP79] D. Marr and T. Poggio. A computational theory of human stereo vision. *Philosophical Transactions of the Royal Society of London, Series A*, 204:301–328, 1979.

[MRR$^+$53] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equation of state calculations by fast computer machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

[MS85] D. Mumford and J. Shah. Boundary detection by minimizing functionals. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Francisco*, pages 22–26, 1985.

[NN97] S. Nene and S. Nayar. A simple algorithm for nearest neighbor search in high dimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(9):989–1003, 1997.

[OF97] B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Research*, 37(23):3311–3325, 1997.

[OK93] M. Okutomi and T. Kanade. A multiple-baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(4):353–363, April 1993.

[PCRZ07] L. C. Pickup, D. P. Capel, S. J. Roberts, and A. Zisserman. Bayesian methods for image super-resolution. *The Computer Journal*, 2007.

[Pea88] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kauffman, San Mateo, California, 1988.

[PL98] R. Paget and I. D. Longstaff. Texture synthesis via a noncausal nonparametric multiscale Markov Random Field. *IEEE Transactions on Image Processing*, 7(6):925–931, June 1998.

[Pot07] B. Potetz. Efficient belief propagation for vision using linear constraint nodes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis*, 2007.

[PP93] K. Popat and R. Picard. Novel cluster-based probability model for texture synthesis, classification, and compression. In *Proceedings of the SPIE Conference on Visual Communications and Image Processing, Boston*, 1993.

[PTK85] T. Poggio, V. Torre, and C. Koch. Computational vision and regularisation theory. *Nature*, 317:314–319, 1985.

[PZF06] M. Prasad, A. Zisserman, and A. W. Fitzgibbon. Single view reconstruction of curved surfaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1345–1354, June 2006.

[RB05] S. Roth and M. J. Black. Fields of experts: A framework for learning image priors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego*, volume 2, pages 860–867, 2005.

[RC98] S. Roy and I. J. Cox. A maximum-flow formulation of the N-camera stereo correspondence problem. In *Proceedings of the 6th International Conference on Computer Vision, Bombay, India*, pages 492–502, 1998.

[RKLS07] C. Rother, V. Kolmogorov, V. Lempitsky, and M. Szummer. Optimizing binary MRFs via extended roof duality. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis*, 2007.

[Rot07] S. Roth. *High-Order Markov Random Fields for Low-Level Vision*. PhD thesis, Brown University, May 2007.

[RSZ06] A. Raj, G. Singh, and R. Zabih. MRF's for MRI's: Bayesian reconstruction of MR images via graph cuts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, New York*, pages 1061–1068, 2006.

[Sch96] D. Scharstein. Stereo vision for view synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Francisco*, pages 852–858, 1996.

[SD97] S. M. Seitz and C. R. Dyer. Photorealistic scene reconstruction by voxel coloring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Puerto Rico*, pages 1067–1073, 1997.

[Set98] J. A. Sethian. *Level Set Methods and Fast Marching Methods*. Cambridge University Press, Cambridge, 1998.

[SF06] D. Schlesinger and B. Flach. Transforming an arbitrary minsum problem into a binary one. Technical Report TUD-FI06-01, Dresden University of Technology, 2006.

[SFVG04] C. Strecha, R. Fransens, and L. Van Gool. Wide-baseline stereo from multiple views: a probabilistic account. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Washington, DC*, volume 1, pages 552–559, June 2004.

[SFVG06] C. Strecha, R. L. Fransens, and L. Van Gool. Combined depth and outlier estimation in multi-view stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, New York*, pages 2394–2401, 2006.

[SIFW03] E. B. Sudderth, A. T. Ihler, W. T. Freeman, and A. S. Willsky. Nonparametric belief propagation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Madison, Wisconsin*, volume 1, pages 605–612, 2003.

[Sim97] E. P. Simoncelli. Statistical models for images: Compression, restoration and synthesis. In *31st Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA.*, November 1997.

[SK00] H. Y. Shum and S. B. Kang. A review of image-based rendering techniques. In *IEEE/SPIE Visual Communications and Image Processing (VCIP) 2000*, pages 2–13, 2000.

[SLKS05] J. Sun, Y. Li, S. B. Kang, and H. Shum. Symmetric stereo matching for occlusion handling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego*, 2005.

[SP07] D. Scharstein and C. Pal. Learning Conditional Random Fields for stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis*, 2007.

[SS96] D. Scharstein and R. Szeliski. Stereo matching with non-linear diffusion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Francisco*, pages 343–350, 1996.

[SS02] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1):7–42, 2002.

[SS08] D. Scharstein and R. Szeliski. Middlebury stereo evaluation table and datasets, November 2008. `http://vision.middlebury.edu/stereo/`.

[SWR07] A. Shahrokni, O. J. Woodford, and I. D. Reid. Temporal priors for novel video synthesis. In *Proceedings of the 8th Asian Conference on Computer Vision, Tokyo, Japan*, pages 601–610, 2007.

[Sze99] R. Szeliski. A multi-view approach to motion and stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Fort Collins, Colorado*, volume 1, pages 157–163, 1999.

[SZS03] J. Sun, N.-N. Zheng, and H.-Y. Shum. Stereo matching using belief propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(7):787–800, July 2003.

[SZS$^+$06] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for markov random fields. In *Proceedings of the 9th European Conference on Computer Vision, Graz, Austria*, pages 16–29, 2006.

[SZS$^+$08] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for markov random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(6):1068–1080, June 2008.

[Ter83] D. Terzopoulos. Multilevel computational processes for visual surface reconstruction. *Computer Vision, Graphics and Image Processing*, 24(1):52–96, October 1983.

[Ter85] D. Terzopoulos. Computing visible-surface representations. Technical Report AI Memo No. 800, MIT Artificial Intelligence Lab., March 1985.

[TF03] M. F. Tappen and W. T. Freeman. Comparison of graph cuts with belief propagation for stereo, using identical MRF parameters. In *Proceedings of the 9th International Conference on Computer Vision, Nice, France*, volume 2, pages 900–906, 2003.

[TRF03] M. F. Tappen, B. C. Russell, and W. T. Freeman. Exploiting the sparse derivative prior for super-resolution and image demosaicing. In *Third International Workshop on Statistical and Computational Theories of Vision*, 2003.

[TSA01] P. H. S. Torr, R. Szeliski, and P. Anandan. An integrated Bayesian approach to layer extraction from image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):297–304, 2001.

[TSK01] H. Tao, H. S. Sawhney, and R. Kumar. A global matching framework for stereo computation. In *Proceedings of the 8th International Conference on Computer Vision, Vancouver, Canada*, pages 532–539, 2001.

[Vek05] O. Veksler. Stereo correspondence by dynamic programming on a tree. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego*, pages 384–390, 2005.

[Vek07] O. Veksler. Graph cut based optimization for MRFs with truncated convex priors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis*, 2007.

[Wei00] Y. Weiss. Correctness of local probability propagation in graphical models with loops. *Neural Computation*, 12(1):1–41, 2000.

[Wel07] M. Welling. Products of experts. *Scholarpedia*, 2(10):3879, 2007.

[WF05] O. Woodford and A. W. Fitzgibbon. Fast image-based rendering using hierarchical image-based priors. In *Proceedings of the British Machine Vision Conference*, volume 1, pages 260–269, 2005.

[WF07] Y. Weiss and W. T. Freeman. What makes a good model of natural images? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis*, 2007.

[WHO02] M. Welling, G. Hinton, and S. Osindero. Learning sparse topographic representations with products of student-t distributions. In *Advances in Neural Information Processing Systems*, pages 1359–1366, 2002.

[WJW03] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. MAP estimation via agreement on (hyper)trees: Message-passing and linear programming approaches. Technical Report UCB/CSD-3-1269, Computer Science Division, University of California, August 2003.

[WL00] L.-Y. Wei and M. Levoy. Fast texture synthesis using tree-structured vector quantization. In *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, pages 479–488, 2000.

[WQ05] Y. Wei and L. Quan. Asymmetrical occlusion handling using graph cut for multi-view stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego*, volume 2, pages 902–909, 2005.

[WRF07] O. J. Woodford, I. D. Reid, and A. W. Fitzgibbon. Efficient new view synthesis using pairwise dictionary priors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis*, 2007.

[WRTF06] O. Woodford, I. D. Reid, P. H. S. Torr, and A. W. Fitzgibbon. Fields of experts for image-based rendering. In *Proceedings of the 17th British Machine Vision Conference, Edinburgh*, volume 3, pages 1109–1108, 2006.

[WRTF07] O. J. Woodford, I. D. Reid, P. H. S. Torr, and A. W. Fitzgibbon. On new view synthesis using multiview stereo. In *Proceedings of the 18th British Machine Vision Conference, Warwick*, volume 2, pages 1120–1129, 2007.

[WS06] J. Winn and J. Shotton. The layout consistent random field for recognizing and segmenting partially occluded objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, New York*, 2006.

[WSI04] Y. Wexler, E. Shechtman, and M. Irani. Space-time video completion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Washington, DC*, volume 1, pages 120–127, 2004.

[WTRF08] O. J. Woodford, P. H. S. Torr, I. D. Reid, and A. W. Fitzgibbon. Global stereo reconstruction under second order smoothness priors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2008.

[YC04] J. Yao and W-K. Cham. An efficient image-based rendering method. In *Proceedings of the 17th International Conference on Pattern Recognition, Cambridge, UK*, volume 1, pages 88–91, 2004.

[YFW00] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Generalized belief propagation. In *Advances in Neural Information Processing Systems*, pages 689–695, 2000.

[YWY$^+$06] Q. Yang, L. Wang, R. Yang, H. Stewénius, and D. Nistér. Stereo matching with color-weighted correlation, hierachical belief propagation and occlusion handling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, New York*, pages 2347–2354, 2006.

[ZK07] C. L. Zitnick and S. B. Kang. Stereo for image-based rendering using image over-segmentation. *International Journal of Computer Vision*, 75(1):49–65, 2007.

[ZM97] S. C. Zhu and D. Mumford. Prior learning and Gibbs reaction-diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(11):1236–1250, November 1997.

[ZVG00] A. Zalesny and L. Van Gool. A compact model for viewpoint dependent texture synthesis. In *Proceedings of the European Conference on Computer Vision*, LNCS 2018/5, pages 124–143. Springer-Verlag, 2000.

[ZWM98] S. C. Zhu, Y. Wu, and D. Mumford. Filters, random-fields and maximum-entropy (FRAME): Towards a unified theory for texture modeling. *International Journal of Computer Vision*, 27(2):107–126, March 1998.