Shape-From-Texture: Spectral Distortion Analysis and 3D Reconstruction Algorithms

Fabio Galasso

Department of Engineering

University of Cambridge

A thesis submitted for the degree of

Doctor of Philosophy

15 Jul 2009

Declaration

This dissertation is submitted for the degree of Doctor of Philosophy. It is the result of my own work and includes nothing which is the outcome of work done in collaboration except where specifically indicated in the text and Acknowledgements. No part of this dissertation has been submitted to any other university. This dissertation contains less than the limits of 150 figures and 65,000 words. It contains approximately 58 figures and 59,500 words.

Ai miei genitori

To my parents

Acknowledgements

It has been a pleasure to work with my supervisor Joan Lasenby. I am grateful for her continual advice, support and encouragement, for her reliable presence and approachable mannerism, for her thoughtful feedback and for allowing me to follow my interests in research.

I am grateful to St. John's college and Giovanni Aldobrandini for the 'A Luisa Aldobrandini' Scholarship, which has supported my graduate studies. This has given me the privileged opportunity to come and study in Cambridge. I also thank the College and the Department of Engineering for funding my travel to conferences, which have been an important part of my development.

I would like to thank my colleagues and friends from the Signal Processing and Communications Laboratory, for their daily help in research and for making my days at work enjoyable. In particular I am grateful to Sze Kim, Charles, Ee Sin, Liam, Simon, Fred, Paris, Pashmina, Adrià, Henry, Paul and Andreas. Thanks to Jonathan for helping me with C++ and Stuart for the support in acquiring pictures.

I thank the researchers and professors who have replied to my requests for pictures and explanations, and in particular Boaz J. Super, W.-L. Hwang, C.-S. Lu, P.-C. Chung, Eraldo Ribeiro, Maureen Clerc, Dorothea Blostein, Narendra Ahuja, Corentin Massot, Yaser Sheikh, Niels Haering, Mubarak Shah, Tony Lindeberg, Antonio Criminisi, Andrea Li, Qasim Zaidi, Tomas Svoboda.

I enjoyed my time in Cambridge thanks to the great friends I met here, and I would like to mention Giorgio, Teresa, Silvio, Lizzy, Anna,

Monique, Wiktor, Aldo, Kamal, Josep, Josepmaria, Arantza, Javier, Elvira, Georg, Marta, Jat, Rok, Giovanni, Benny and Martin.

A special thank goes to Valentina, for accepting that I did not choose to pursue a PhD in the Bahamas (or any other exotic island) and to zio Francesco, for encouraging me to study abroad.

Finally I thank my parents, for their love and continual trust. They helped me face the difficulties that I encountered and injected me with the energy and enthusiasm to cope with and, most importantly, to enjoy this PhD.

Abstract

Understanding a scene is often the basis for subsequent interaction with it, for example, navigation and object avoidance. We note that understanding generally requires the interpretation of low level cues, e.g. shape, to accomplish higher level tasks.

Among the cues that can be used to infer the 3D structure of the scene, researchers have studied shading, contours, multiple views, shadows, etc., and combinations of these attributes.

Shape-From-Texture, first proposed by Gibson in 1950, is the area of Computer Vision which studies the recovery of the shape of a textured object from a monocular static image.

The study of Shape-From-Texture is motivated by psychological studies, according to which, texture is important in the perception of shape. Moreover, texture is ubiquitous.

However, the task is ill-posed for a machine and Shape-From-Texture is generally considered a hard problem. In fact, some initial assumption is always necessary to identify the characteristic of the texture conveying the shape information, and a restrictive one can preclude applicability to real images. For this reason we do not assume isotropy nor the presence of identifiable texels in this thesis, as these are rarely the cases in natural scenes.

We assume the very general 'stochastic' homogeneity of Malik and Rosenholtz (*International Journal of Computer Vision* '97). For deterministic textures, this means periodicity. For stochastic textures, i.e. textures assumed to be generated by a stochastic process, this means that the generating process is stationary under translation.

Given initial assumptions, Shape-From-Texture is generally seen as two distinct problems: the estimation of the texture distortion, due to the geometry of the scene and the projection from the 3D world to the 2D image; and the 3D shape reconstruction, given by interpreting the measured distortion. The work detailed in this thesis contributes to both these aspects.

In particular, the distortion information is characterized by the use of *local spatial frequencies*. A novel method is introduced which combines Fourier analysis and the use of Gabor functions to recover all the main instantaneous frequencies of the texture at each pixel.

The Fourier transform of the image provides valuable information about the number and range of local spatial frequencies contained in the texture, yielding fast setup and optimized computational expense. We use the spectrum amplitude of the texture to detect the significant scales, and adapt the well known method of Super and Bovik (*Pattern Recognition* '95), based on selected Gabor filters and their first derivatives, to estimate the relevant local frequencies at each scale.

This new method uses the multi-scale nature of textures by recovering the distortion information at all scales, while most related works estimate only a single distortion measure or use just two preferred directions in the spectral domain.

The second Shape-From-Texture problem, that of the 3D reconstruction from the multi-scale distortion, is then solved at different levels of complexity.

The simple case of a planar surface is solved for each estimated local spatial frequency with the method of Super and Bovik (*Pattern Recognition* '95) and a novel method is proposed to choose the best computed orientation according to a residual variance.

The case of surfaces with available frontal views is solved for each estimated local spatial frequency with a closed-form equation which provides the shape orientation locally from neighbouring pixels, and

for each pair of local spatial frequencies with the *singular value decomposition* and the method of Dugué and Elghadi (*Scandinavian Conference on Image Analysis* '99), which we extend to perspective views.

The case of general surfaces is solved for each pair of local spatial frequencies by using *local frequency variations* and *eigendecomposition*. This novel 3D reconstruction method draws inspiration from the method Massot and Hérault (*International Journal of Computer Vision* '08), generalizing the solution to more complex shapes and providing better accuracy and robustness.

Both the easier orthographic and the more general perspective cases are considered, and the advantage of using redundant local spatial frequencies is evaluated.

Furthermore, the implemented Shape-From-Texture algorithms are tested on semi-synthetic and real images. In fact, some new real images with known ground truth are acquired to fill in a notable gap in the literature. The accuracy of the proposed algorithms is compared with that of relevant work, when possible, and robustness is shown for a range of noise values.

The multi-scale distortion estimation and the study of geometric relations is applied to the Computer Graphics fields of *texture mapping* and *texture replacement*.

In particular, a texture mapping algorithm is implemented by using the geometric equations relating the surface texture to the image plane and the integration method of Frankot and Chellappa (*Pattern Analysis and Machine Intelligence* '88). The algorithm produces visually pleasing results and can be used to replace a homogeneous texture in an image once the 3D shape is recovered.

Alternatively, the texture replacement algorithm is an original approach to re-texturing, where the texture in the image is just replaced

by using the estimated local spatial frequencies, without recovering the 3D shape.

Finally two potential areas for future research in Shape-From-Texture are illustrated: Shape-From-Texture in a controlled environment and Shape-From-Texture for natural scenes. The former addresses the recovery of the shape and the shape changes of objects by laying a convenient textured film on them. The latter indicates the use of Shape-From-Texture to recover the structure of natural scenes from photographs of, for example, dry dams, sandy beaches or wooded hills.

Contents

Contents ix

List of Figures xii

List of Tables xx

List of Algorithms xxi

Nomenclature xxii

1 Introduction 1

1.1 Road Map 4

2 Overview of Shape-From-Texture 7

2.1 Assumptions on the Texture 7

2.2 Homogeneity and the Spectral Analysis of Textures 11

2.3 Viewing Geometry 13

2.4 Assumptions on the Camera Model and on the Texture Surface . 15

2.5 Integrating the Orientation Parameters to Recover the 3D Shape . 17

2.6 Surface Texture Model 19

2.7 Previous Work on Shape-From-Texture 20

2.8 Contribution 23

3 A Multi-Scale Distortion Estimation 25

3.1 A Method to Estimate Instantaneous Frequencies 26

3.2 The Setup of Gabor Filter Parameters 28

3.2.1 The case of a single LSF 29

3.2.2 The case of two LSFs 34
3.2.3 The case of a Brodatz texture 36
3.3 Computing All the Main LSFs 40
3.4 Examples of LSF Estimations 43
3.5 Final Remarks 50

4 The Orientation of a Planar Surface 52
4.1 Projection Model 53
4.2 Computing the Orientation of a Plane 55
4.3 Results 57
4.4 Final Remarks 63

5 Frontal Views in Shape-From-Texture 65
5.1 Detection and Localisation of a Frontal Point 66
5.2 A Frontal View and a Local Spatial Frequency 69
5.2.1 Projection model extended to developable shapes and orthographic projections 69
5.2.2 Computing the 3D coordinates of developable surfaces . . 73
5.2.3 Results 76
5.2.4 Remarks on the algorithm 83
5.3 A Frontal View and Two Local Spatial Frequencies 84
5.3.1 Viewing geometry and projection model revisited 84
5.3.2 Computing the 3D coordinates with singular value decomposition 88
5.3.3 Results and comparison 94
5.4 Final Remarks 98

6 Local 3D Shape from Local Frequency Variations 100
6.1 Computing Local 3D Coordinates 101
6.1.1 An equation for local frequency variations 101
6.1.2 Solving for slant and tilt locally 106
6.1.2.1 With eigenvalues and eigenvectors 107
6.1.2.2 Via integration 109
6.1.2.3 Remarks on the solutions 112

6.1.3 Algorithm structure 114
6.2 Results and Comparisons 115
6.3 Final Remarks 127

7 Texture Mapping and Replacement 129
7.1 Introduction to Texture Mapping and Replacement 130
7.1.1 Introduction to texture mapping 130
7.1.2 Introduction to texture replacement 133
7.2 Texture Mapping 135
7.2.1 Examples of texture mapping 140
7.3 Texture Replacement 144
7.3.1 Examples of texture replacement 148
7.3.2 Texture replacement when a frontal point is not available . 149
7.3.3 Examples of texture replacement when a frontal point is not available 153
7.4 Final Remarks 155

8 Conclusions and Future Work 157
8.1 Contributions 157
8.2 Summary of Work 158
8.3 Directions for Future Work 161

A Two other Methods to Solve for the Local Orientation from Local Frequency Variations 166
A.1 Solving for Slant and Tilt Locally Using Ratios of Determinants . 167
A.2 Solving for Slant and Tilt Locally Using Differences of Determinants 170

References 174

List of Figures

1.1 Samples of textures: (*top row*) textures where the single texels can be distinguished; (*bottom row*) textures which need to be defined statistically. 2

1.2 Example of SFT in a controlled environment: a textured cloth laid on a balloon can allow us to reconstruct the shape of the balloon . 4

2.1 Planar surface textured with circles, taken from [52] 8

2.2 Viewing geometry 14

3.1 (a) 1D cosine at frequency $\tilde{u} \approx 0.47$ rad/s; (b) its instantaneous frequency; (c) its spectrum amplitude; (d) 1D cosine with frequency varying from $\tilde{u} \approx 0.38$ rad/s to $\tilde{u} \approx 1.15$ rad/s; (e) its instantaneous frequency; (f) its spectrum amplitude 29

3.2 (a) 2D cosine at frequency $|\tilde{\mathbf{u}}| \approx 0.47$ rad/s; (b) its spectrum amplitude 30

3.3 (a) 2D cosine with frequency varying from $|\tilde{\mathbf{u}}| \approx 0.38$ rad/s to $|\tilde{\mathbf{u}}| \approx 1.15$ rad/s; (b) its spectrum amplitude; (c) true LSFs (*yellow area*) represented on the spectrum; (d) chosen Gabor functions (*yellow circles*) superposed on the spectrum 30

3.4 Lattice of Gabor filters used by Super and Bovik in [158, 159] . . 33

3.5 (a) Two 2D cosines superposed, at frequencies $|\tilde{\mathbf{u}}_1| \approx 0.47$ rad/s and $|\tilde{\mathbf{u}}_2| \approx 0.72$ rad/s, 45° apart in the spectral domain; (b) its spectrum amplitude 34

3.6 (a) Result of slanting the texture in fig. 3.5(a) by 40°; (b) its spectrum amplitude; (c) true LSFs (*yellow areas*) represented on the spectrum; (d) chosen Gabor functions (*yellow circles*) superposed on the spectrum 34
3.7 (a) D6 Brodatz texture; (b) its spectrum amplitude 37
3.8 (a) 3D slanted plane; (b) D6 rendered on the slanted plane of (a); (c) spectrum amplitude of (b); (d) true LSFs (*yellow areas*) represented on the spectrum; (e) chosen Gabor functions (*yellow circles*) superposed on the spectrum 37
3.9 (a) 3D cosine surface; (b) D6 rendered on the cosine surface of (a); (c) spectrum amplitude of (b); (d) true LSFs (*yellow areas*) represented on the spectrum; (e) chosen Gabor functions (*yellow circles*) superposed on the spectrum 38
3.10 (a) 3D cosine surface; (b) 128x128 semi-synthetic picture (as in fig. 3.9(b)) obtained by rendering Brodatz texture D6 on the cosine surface of (a); (c) spectrum amplitude of (b) with labels indicating the selected LSFs 44
3.11 Moduli and phases of the estimated LSFs of the image in figure 3.10(b) 45
3.12 Moduli and phases of the ground truth LSFs of the image in figure 3.10(b) 45
3.13 (a) 640x480 real picture of a rubber rug laid on a cylinder; (b) 180x180 real image extracted from (a); (c) spectrum of (b) with labels indicating the selected LSFs 47
3.14 Moduli and phases of the estimated LSFs of the image in figure 3.13(a) 48
3.15 (a) 640x480 real picture of a textured cloth laid on an inflated balloon; (b) 128x128 real image extracted from (a); (c) spectrum of (b) with labels indicating the selected LSFs 49
3.16 Moduli and phases of the estimated LSFs of the image in figure 3.15(a) 50

4.1 Viewing geometry and projection model 53

4.2 Images synthesized using the Brodatz textures ([18]) 58
4.3 Real images of textured planes 61
4.4 Full resolution (640x480) image of the towel in figure 4.3(j) 62

5.1 (a) 128x128 semi-synthetic picture obtained by rendering Brodatz texture D6 on a cosine surface, same as 3.9(b); (b) its spectrum with labels indicating the selected LSFs; (c) arrows indicating the respective spectral directions of the selected LSFs at the central pixel superposed on (a) 77
5.2 (a) Ground truth shape of the cosine surface in fig. 5.1(a), central 64x64 part; (b) orthographic and (c) perspective reconstructions achieved from the image in fig. 5.1(a) with algorithm 3 and LSF 2, central 64x64 part 77
5.3 (a) Orthographic and (b) perspective reconstructions achieved from the image in fig. 5.1(a) with algorithm 3 and LSF 4, central 64x64 part 77
5.4 (a) 180x180 real image of a rubber rug laid on a cylinder extracted from 3.13(a), as in fig. 3.13(b); (b) its spectrum with labels indicating the selected LSFs; (c) arrows indicating the respective spectral directions of the selected LSFs at the central pixel superposed on (a) 79
5.5 (a) Ground truth shape of the cylinder beneath the rubber rug in fig. 5.4(a), central 90x90 part; (b) orthographic and (c) perspective reconstructions achieved from the image in fig. 5.4(a) with algorithm 3 and LSF 1, central 90x90 part 79
5.6 (a) Orthographic and (b) perspective reconstructions achieved from the image in fig. 5.4(a) with algorithm 3 and LSF 3, central 90x90 part 79
5.7 (a) 640x480 real picture of a pair of trousers laid on a cylinder; (b) 128x128 real image extracted from (a); (c) spectrum of (b) with labels indicating the selected LSFs 80

5.8 (a) Ground truth shape of the cylinder beneath the trousers in fig. 5.7(b), central 64x64 part; (b) orthographic and (c) perspective reconstructions achieved from the image in fig. 5.7(b) with algorithm 3 and LSF 2, central 64x64 part 81
5.9 (a) 640x480 real picture of a sponge laid on a cylinder; (b) 128x128 real image extracted from (a); (c) spectrum of (b) with labels indicating the selected LSFs 81
5.10 (a) Ground truth shape of the cylinder beneath the sponge in fig. 5.9(b), central 64x64 part; (b) orthographic and (c) perspective reconstructions achieved from the image in fig. 5.9(b) with algorithm 3 and LSF 2, central 64x64 part 82
5.11 Semi-synthetic and real images of figs.5.1(a), 5.4(a), 5.7(b), 5.9(b), respective ground truth and reconstructed shapes achieved by using algorithm 5 in orthographic and perspective cases, with LSFs 2-4 ((c), (d)), 1-3((g), (h)), 1-2((k), (l)), 1-2((o), (p)). As in section 5.2.3, the central parts of the ground truth and reconstructed shapes are represented 93
5.12 (a) Frame 1 extracted from a 400x300 real video sequence of a textured cloth laid on a deflating balloon; (b) 128x128 real image extracted from (a); (c) spectrum of (b) with labels indicating the two selected LSFs 95
5.13 128x128 real images extracted from the frames of a video sequence of a textured cloth laid on a deflating balloon. Top row: frames 1,100,200,300; Mid row: respective orthographic reconstructions using algorithm 5 on LSFs 1 and 2; Bottom row: corresponding perspective reconstructions using algorithm 5 on LSFs 1 and 2 . . 95
5.14 First row: respective orthographic reconstructions of the frames (1,100,200,300) in figure 5.13 using algorithm 3 on LSF 1; Second row: respective perspective reconstructions using algorithm 3 on LSF 1 96

6.1 (*First row*) 128x128 semi-synthetic image of Brodatz texture D6 on a cosine surface, spectral directions of the selected LSFs and ground truth shape, central 64x64 part, already described in figs. 5.1 and 5.2(a). (*Second row*) Perspective reconstructions (same central part) of shape from image (a) using our method and the method of [121] on the indicated LSFs. (*Third row*) 180x180 real image of a rubber rug laid on a cylinder, spectral directions of the selected LSFs and ground truth shape, central 90x90 part, already described in figs. 5.4 and 5.5(a). (*Fourth row*) Perspective reconstructions (same central part) of shape from image (g) using our method and the method of [121] on the indicated LSFs 116

6.2 (*First row*) 128x128 real image of a pair of trousers laid on a cylinder, spectral directions of the selected LSFs and ground truth shape, central 64x64 part ((a) and (c) presented in figs. 5.7 and 5.8(a)). (*Second row*) Perspective reconstructions (same central part) of shape from image (a) using our method and the method of [121] on the indicated LSFs. (*Third row*) 128x128 real image of a sponge laid on a cylinder, spectral directions of the selected LSFs and ground truth shape, central 64x64 part ((g) and (i) presented in figs. 5.9 and 5.10(a)). (*Fourth row*) Perspective reconstructions (same central part) of shape from image (g) using our method and the method of [121] on the indicated LSFs 117

6.3 (a) 640x480 real picture of a pillowcase laid on a cylinder; (b) 240x240 real image extracted from (a); (c) spectrum of (b) with labels indicating the selected LSFs; (d) arrows indicating the respective spectral directions of the selected LSFs at the central pixel superposed on (b); (e) ground truth shape of the cylinder beneath the pillowcase in (b), central 120x120 part; (f,g,h) perspective reconstructions (same central part) of shape from image (b) using: our method on LSFs 1 and 2, the method of [121] on LSF 1, the method of [121] on LSF 2 119

6.4 (a) 640x480 real picture of a rubber rug laid on a planar surface; (b) 128x128 real image extracted from (a); (c) spectrum of (b) with labels indicating the selected LSFs; (d) arrows indicating the respective spectral directions of the selected LSFs at the central pixel superposed on (b); (e) ground truth shape of the plane beneath the rubber rug in (b), central 64x64 part; (f,g,h) perspective reconstructions (same central part) of shape from image (b) using: our method on LSFs 1 and 2, the method of [121] on LSF 1, the method of [121] on LSF 2 120

6.5 (a) Frame 1 extracted from a 400x300 real video sequence of a textured cloth laid on a deflating balloon; (b) 128x128 real image extracted from (a); (c) spectrum of (b) with labels indicating the two selected LSFs;(d) arrows indicating the respective spectral directions of the two LSFs at the central pixel superposed on (b) . 123

6.6 128x128 real images extracted from the frames of a video sequence of a textured cloth laid on a deflating balloon. (*first row*) frames 1,100,200,300; (*second row*) perspective reconstructions with our method using LSFs 1 and 2; (*third row*) perspective reconstructions with the method of [121] using LSF 1; (*fourth row*) perspective reconstructions with the method of [121] using LSF 2 124

6.7 (a) Cosine textured with a LSF (its spectral direction represented by the red arrow) orthogonal to the tilt direction; (b) cosine textured with a LSF orthogonal to the tilt direction (same as (a)) and a second LSF directed parallel to the tilt direction (the spectral directions of the LSFs are represented with arrows) 125

7.1 (a) D53 Brodatz texture; (b) segment of a sphere; (c) D53 Brodatz texture rendered on the segment of a sphere given in (b); (d) D76 Brodatz texture; (e) corner of a cube; (f) D76 Brodatz texture rendered on the corner of a cube given in (e) 141

7.2 (a) Real picture of a pillowcase laid on a cylinder, as illustrated in fig. 6.3(b); (b) perspective reconstruction of the cylindrical shape from image (a), as shown in fig. 6.3(f); (c) chessboard motif; (d) chessboard motif rendered on the shape of (b) 142
7.3 (a) Real picture of a rubber rug laid on a planar surface, as illustrated in fig. 6.4(b); (b) perspective reconstruction of the planar shape from image (a), as shown in fig. 6.4(f); (c) written page motif; (d) written page motif rendered on the shape of (b) 143
7.4 (a) Real picture of a textured cloth laid on an inflated balloon, with arrows indicating the spectral directions of the two selected LSFs, as illustrated in fig. 6.5(d); (b) D6 Brodatz texture, shown in fig. 3.7(a), replaced onto the image of (a) by using the two indicated LSFs 147
7.5 (a) Real picture of a rubber rug laid on a planar surface, as in fig. 7.3(a), with arrows indicating the spectral directions of the two selected LSFs, as illustrated in fig. 6.4(d); (b) written page motif of fig. 7.3(c) replaced onto the original image of (a) by using the two indicated LSFs 147
7.6 (a) Texture of two superposed cosines rendered on a plane; (b) planar surface underlying (a); (c) texture of (a), with arrows indicating the spectral directions of the two selected LSFs 153
7.7 (a) Written page motif, reproduced from fig. 7.3(c); (b) written page motif of (a) rendered onto the planar shape of fig. 7.6(b); (c) written page motif of (a) replaced onto the image of fig. 7.6(a) by using the two available LSFs and the central point of fig. 7.6(a) . 154
7.8 Details of the central parts of (a) fig. 7.7(b) and (b) fig. 7.7(c) . . 154
7.9 (a) Written page motif of fig. 7.7(a) pre-slanted according to the orientation of the central point of fig. 7.6(b); (b) pre-slanted written page motif of (a) replaced onto the image of fig. 7.6(a) by using the two available LSFs and the central point of fig. 7.6(a); (c) detail of the central part of (b) 155

8.1 240x240 real image of a dry dam 163

8.2 (a) Spectrum of the image in figure 8.1; (b) smoothed spectrum, spatial constant of smoothing Gaussian is $\gamma = 1$; (c) smoothed spectrum in (b) labeled to indicate the interest area for LSF1 . . 163
8.3 Modulus and phase of the estimated LSF of the image in figure 8.1 164
8.4 3D reconstruction of the image in figure 8.1, seen from the side . . 164

List of Tables

4.1 Tilt results of our method (τ_{GL}) on images 4.2(a)-(g), compared to the results of [74] (τ_{HLC}) (angles in degrees) 57

4.2 Slant results of our method (σ_{GL}) on images 4.2(a)-(g), compared to the results of [74] (σ_{HLC}). We choose for comparison the values they achieve with what they term the 'voting method' (angles in degrees) 59

4.3 Tilt and slant results of our method (τ_{GL},σ_{GL}) on images 4.2(h)-(o) (angles in degrees) 59

4.4 Surface orientations (τ/σ) estimated using our method on noisy versions of images 4.2(a),(l),(b),(e),(g),(o) - true values are in parenthesis (X indicates that the results were not reliable) (angles in degrees) 60

4.5 Tilt and slant results of our method (τ_{GL},σ_{GL}) on real images in figure 4.3 (angles in degrees) 62

5.1 Average estimation errors for σ and τ of the reconstructions illustrated in figs.5.2, 5.3, 5.5, 5.6, 5.8, 5.10, achieved using algorithm 3 in the orthographic and perspective cases 83

5.2 Average estimation errors for σ and τ of the reconstructions illustrated in fig. 5.11, achieved using the algorithm of 5 in the orthographic and perspective cases 92

6.1 Average estimation errors (in degrees) for σ and τ of the reconstructions illustrated in figs. 6.1, 6.2, 6.3, 6.4 122

List of Algorithms

1 Structure of the proposed algorithm to estimate the multi-scale distortion of a texture image 42

2 Structure of the proposed algorithm to estimate the orientation of a planar surface from multiple estimated LSFs 56

3 Structure of the proposed algorithm to recover the shape of a developable surface from multiple estimated LSFs by using one LSF at a time, given a frontal point or a point of known orientation . . 76

4 Proposed iterative algorithm to estimate the local orientations and shape in the perspective case by using SVD 90

5 Structure of the proposed algorithm to recover the 3D shape from multiple estimated LSFs by using pairs of LSFs via SVD, given a frontal point or a point of known orientation or an additional frontal patch of the texture 91

6 Structure of the proposed algorithm to recover the 3D shape from multiple estimated LSFs by using the local variations of pairs of LSFs 115

Nomenclature

Roman Symbols

f Focal length

I Identity matrix

$J^t(\mathbf{x}_i, \mathbf{x}_s)$ Transpose Jacobian of the vector function $\mathbf{x}_i(\mathbf{x}_s)$ with respect to $\mathbf{x}_s$

$J^t(\mathbf{x}_s, \mathbf{x}_i)$ Transpose Jacobian of the vector function $\mathbf{x}_s(\mathbf{x}_i)$ with respect to $\mathbf{x}_i$

$J^t_{\mathbf{x}_i\mathbf{x}_s}$ Transpose Jacobian of the vector function $\mathbf{x}_i(\mathbf{x}_s)$ with respect to $\mathbf{x}_s$, same as $J^t(\mathbf{x}_i, \mathbf{x}_s)$

$J^t_{\mathbf{x}_s\mathbf{x}_i}$ Transpose Jacobian of the vector function $\mathbf{x}_s(\mathbf{x}_i)$ with respect to $\mathbf{x}_i$, same as $J^t(\mathbf{x}_s, \mathbf{x}_i)$

$R(\tau)$ Rotation matrix, function of the angle τ

σ Slant

τ Tilt

$P(\sigma)$ Foreshortening matrix, function of the angle σ

U_i Matrix of image plane frequencies

$\mathbf{u}_i = (u_i, u_i)$ Image plane frequency at the image plane point $\mathbf{x}_i$

U_s Matrix of surface texture frequencies

$\mathbf{u}_s = (u_s, u_s)$ Surface texture frequency at the surface texture point $\mathbf{x}_s$

$\mathbf{x}_i = (x_i, y_i)$ Image plane coordinate

$\mathbf{x}_s = (x_s, y_s[, z_s])$ Surface texture coordinate

$\mathbf{x}_w = (x_w, y_w, z_w)$ World coordinate

z_0 z_w coordinate of the intersection of the optical axis and of the tangent plane at the considered surface texture point

Superscripts

$'$ LSF index

$''$ LSF index

Subscripts

i Image plane index

s Surface texture index

w World coordinate index

Acronyms

CG Computer Graphics

FFT Fast Fourier Transform

LSF Local Spatial Frequency

SFS Shape-From-Shading

SFT Shape-From-Texture

SVD Singular Value Decomposition

Chapter 1

Introduction

Understanding a scene is a fundamental task, for machines as well as for humans. It is the basis for subsequent interaction with the scene and the objects in it, for recognition and navigation.

Humans are usually able to realise this task nearly instantaneously, and sometimes it seems that they do not devote even a conscious thought to it.

Machines have usually at hand 2D views of the world (images) and sometimes series of images (videos) and images from multiple views. Computer Vision is the branch of engineering which studies images. It lets a machine understand a scene by extracting pieces of information: position, size, shape, colour, presence of people, etc.

We aim to extract shape, a low level piece of information, which is often the basis for higher level tasks, such as detection and interaction. Shape can be extracted from a variety of visual cues (e.g. shading, contours, multiple views, shadows, texture etc.) and often the extraction of shape from each cue corresponds to a particular field of Computer Vision, known as Shape-From-X, where X is the considered cue.

As also noted by Loh in [111], much work has been devoted to each corresponding area of Computer Vision, e.g. to Shape-From-Shading ([19, 20, 70, 72] or the surveys in [92, 178]), Shape-From-Multiple views ([15, 124, 150]), Shape-From-Contours ([90, 116, 164]), Shape-From-Shadows ([39, 64, 82, 176]), and sometime combining different cues ([36, 172]).

Figure 1.1: Samples of textures: (*top row*) textures where the single texels can be distinguished; (*bottom row*) textures which need to be defined statistically.

In the work we present here, we consider *texture*. Texture has been studied for over 30 years and received a variety of definitions, depending on whether scientists were considering its nature as composed of repetitive elements (texture elements or *texels*) or whether they were using its statistical characteristics. In the first case, a texture can be the pattern produced by the pebbles on a beach or the bricks of a wall, but it can also be the pattern given by a sandy beach, where the texel is the single grain. As an example of statistically defined texture we cite [151]:

> A region in an image has a constant texture if a set of local statistics or other local properties of the picture function are constant, slowly varying, or approximately periodic.

A texture defined in this sense can be for example the pattern created by the earth cracks in a dry dam, by the bark of a tree or by the dunes on the sand. Figure 1.1 presents some samples of textures.

Texture is mainly used for segmentation, classification, synthesis and Shape-From-Texture, and some works are aimed to the study of different tasks at the same time ([93]).

This doctorate work is concerned with *Shape-From-Texture* (SFT). Shape-From-Texture is the area of computer vision which studies the recovery of the shape of a texture object from a monocular static image.

From psychological studies we know that texture is important in the perception of shape ([22, 37, 161]). Furthermore, texture is ubiquitous, and a solution to the SFT problem would be a powerful tool. For example it would be useful to reconstruct, from an aerial picture, the shape of a dry dam, or of a hill covered up with a forest (the foliage of trees would be the analysed texture).

Shape-From-Texture is considered a difficult problem. It is clear from the difficulty in defining a texture, that an initial assumption is always necessary (e.g. some regularity of the pattern, or the presence of particular texels) in order to identify the characteristic of the texture conveying the deformation. SFT is then based on the assumption that the deformation observed in the texture is only due to the projection of the texture from the 3D world to the 2D image.

Given an initial assumption, and therefore once a definition of the texture is chosen, SFT consists of two distinct problems ([30, 42, 49, 52, 74, 80, 112, 118, 121, 139, 159]):

- Distortion estimation: quantitatively measure the distortion that the texture has undergone, as due to the geometry of the object;
- Shape reconstruction: mathematically interpret the distortion to recover the 3D of the textured object.

The study of SFT is generally motivated by applications, the initial stage of which is shape reconstruction. The Computer Graphics field of re-texturing would benefit from a SFT algorithm. In fact, once the 3D structure of a scene is recovered, its shape can be modified and the image re-rendered with a new texture. Our motivation, as well as our prospective study, is mainly focused in two directions:

- SFT in a controlled environment: SFT can be used to non-invasively study the shape of objects by laying a general textured cloth on them, as shown in figure 1.2. In addition, it can be used to analyse the frames of a video and non-invasively study the shape changes;

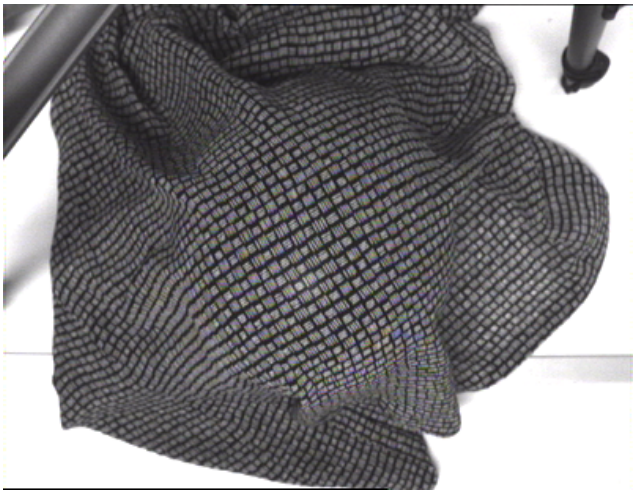


Figure 1.2: Example of SFT in a controlled environment: a textured cloth laid on a balloon can allow us to reconstruct the shape of the balloon

- SFT for natural scenes: SFT can be used to recover the 3D structure of textured natural scenes, e.g. the pictures of the dry dam with cracks in the earth and of the dunes on the sand in figure 1.1.

Finally, SFT can be integrated with other low level pieces of information and used to achieve higher level tasks in computer vision, e.g. improved interaction of a robot with the environment.

1.1 Road Map

This PhD thesis is organized as follows:

- Chapter 2 gives an overview on SFT, defining the initial assumptions, the viewing geometry and the surface texture model. There, we also present a review of previous works on SFT and illustrate our contribution to the field;

- Chapter 3 presents an original method to estimate the multi-scale distortion in the texture based on *local spatial frequency* (LSF). The Fourier transform

of the texture is used to set minimal groups of Gabor filters, with which the main LSFs of the texture are recovered, characterizing a multi-scale distortion estimation;

- Chapter 4 illustrates how each estimated LSF is utilised to compute the orientation of a textured plane by using the algorithm proposed by Super and Bovik in [158]. An original method to choose from the best estimated orientation is proposed, the method is tested on semi-synthetic and real textures, and results are compared with an existing method;

- Chapter 5 presents two algorithms which compute the shape of a textured object showing a frontal point. The first algorithm is an original contribution, it applies to both orthographic and perspective cases recovering unfoldable textured surfaces shapes from each LSF. The second algorithm is derived by extending the orthographic method of Dugué and Elghadi ([42]) to perspective cases, it recovers a shape, including non-developable shapes, from each pair of LSFs. Both the algorithms are tested on real textures;

- Chapter 6 presents an original 3D reconstruction algorithm which estimates the shape of a general textured object (planar and curved, unfoldable and non-unfoldable, with and without a frontal point) from the local variations of each pair of LSFs. The method draws inspiration from the method of Massot and Hérault ([121]), it is tested on real textures, and an application to video is shown;

- Chapter 7 presents an application of our geometrical algorithms to the computer graphics tasks of texturing and re-texturing, showing how the texture in an image can be changed once a shape has been recovered, and how it is also possible to replace the texture without estimating the shape, simply using the estimated distortion;

- Finally Chapter 8 draws conclusions on the work presented and illustrates the directions of possible future work.

The thesis is accompanied by a DVD on which a digital copy of the document is contained. This allows the reader to zoom in the represented vectorial figures. The DVD also contains the videos of the reconstructed 3D shapes from the real video sequence dataset presented in chapters 5 and 6.

Chapter 2

Overview of Shape-From-Texture

This thesis presents work on Shape-From-Texture, the area of Computer Vision which aims to recover the shape of textured objects and scenes from static monocular images.

SFT is considered a difficult problem, one of the reasons being that it is problematic to define 'texture'. In fact, every work on SFT starts with describing some initial assumptions about the nature of the texture.

Such assumptions are of fundamental importance in estimating the distortion in the image, which is supposed to derive only from the projection of the object from the 3D world to the 2D image, and they define the applicability of the developed algorithms.

Given a projection geometry and an additional assumption about the camera model, the estimated distortion can be translated into the 3D coordinates of the object, i.e. the recovered shape, by using a reconstruction algorithm.

These assumptions and the projection geometry are examined in this chapter, where we also offer an overview of the main contributions on the subject from the literature, and on how our contribution is likely to impact in the field of SFT.

2.1 Assumptions on the Texture

In a seminal work in 1950 by Gibson, the SFT problem was addressed for the first time. In [59], Gibson proposed extracting the 3D information from the *density texture gradient* of *texels* in a textured image.

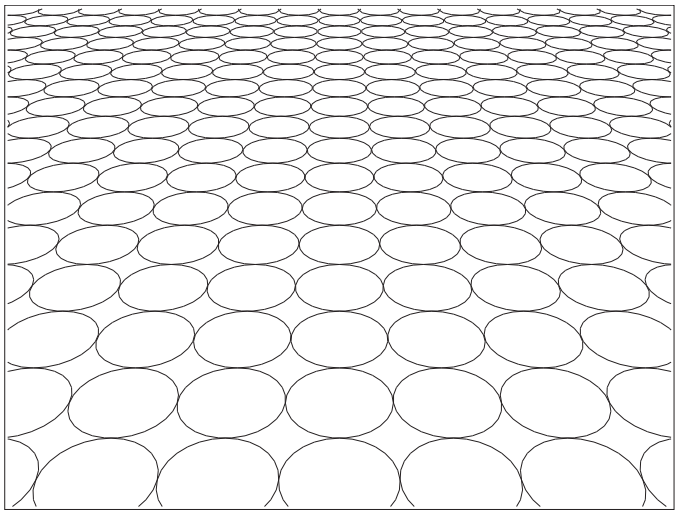


Figure 2.1: Planar surface textured with circles, taken from [52]

Both the terms *texels* and *texture gradient* were coined by Gibson. The former was to indicate that the texture was composed of identifiable texture elements, e.g. the bricks, pebbles and sand grains of figure 1.1 in chapter 1. The latter was to address the phenomenon that similar neighboring patches generally project in the retinal image to corresponding patches with different appearances, due to their different distance and orientation.

Today, Gibson's approach would be described as attempting to recover the shape from the density texture gradient, given a homogeneity assumption of constant density of the texels on the texture.

Although Gibson was lacking a precise mathematical formulation, his idea was seminal and remains one of the fundamental approaches to SFT: given an assumption of *homogeneity*, either based on texels or on the local statistics of a texture, the shape is recovered from the variations (gradients) between the texels or the selected statistical properties of patches, which occur when projecting the textured surface from the 3D world to the 2D image.

Drawing on the comprehensive studies of Stevens ([155]) and Gårding ([52]), we illustrate the most commonly used texture gradients. We refer to figure 2.1, where a planar surface with circular texels is slanted around the horizontal axis,

with the effect of transforming the circles into ellipses. The deviation from a homogeneous distribution can be described by the following:

- the compression gradient: due to the changes of the minor axes of the ellipses;
- the scaling or perspective gradient: due to the changes of their major axes;
- the foreshortening gradient: due to the changes of their aspect ratio;
- the area gradient: due to the changes of their areas;
- the density gradient: due to the changes of density of the ellipses.

As mentioned, being based on variations between patches, the texture gradients require the assumption of some sort of homogeneity, which depends on the particular gradient. For example, [119] uses the variations in the density of texels to recover the shape and orientation of the surface, and therefore requires the assumption that the density of the texture is constant, or nearly constant.

Of a different nature is the assumption of *isotropy*. Originally introduced by Witkin in [174], the isotropic assumption means that the scene texture does not have a dominant direction or orientation bias (e.g. [9, 53, 174]). In [111], isotropy means that the texels have a constant second moment or inertia about every axis. Therein the inertia is defined as a function of angle, given by

$$\text{Inertia}(\theta) = \sum_i \sum_j r^2 f(i,j) \tag{2.1}$$

where $f(i,j)$ is the value of the image of the texel at (i,j) and r is the perpendicular distance from (i,j) to the axis in the direction θ.

Applied to the slanted circles in figure 2.1, isotropy would consider the fact that the inertia of the texels is more spread along the horizontal axis, and quantify the distortion with the value of spread.

In general, by using a isotropic assumption, the orientation is recovered at a point by using the local anisotropy, i.e. the local deviation from isotropy.

Homogeneity and isotropy have been used in a number of studies on SFT, often posing the question whether one is to be preferred to the other, according to applicability and amount of information conveyed ([145]).

Studies have been done to distinguish whether the human visual system uses isotropy or homogeneity ([10, 37, 62, 86, 87, 88, 145, 161]), but this is still an open question.

With respect to the homogeneity assumption, isotropy has the advantage of providing the local orientation directly, due to the local deviation from isotropy, with no integration or comparison required. On the other hand, the homogeneity assumption does not introduce a direct parameterization and so requires the computation of a gradient, and the shape is usually recovered by integrating the local information over the image.

However, a large class of natural textures are homogeneous, while only a few are isotropic (as also maintained in [48, 118, 120]). In fact, when assuming homogeneity, the scene texture can also be anisotropic, as it is the variation of statistics between the image patches that counts, more than the statistics themselves.

Finally, we also present the assumption used by Blostein and Ahuja ([13]), by Forsyth and Lobay ([48, 110]), and by Loh and Hartley ([112]). These authors assume it is possible to recover the appearance of single texels. Part of the assumption resides in the fact that the quality of the obtained appearances have to be accurate enough to be used to determine the supposed frontal appearance of the texel composing the texture. [48, 110, 112] use the estimated frontal appearance of the texel to compute the affine transformations for the image points, so as to ultimately recover the shape by integration in the orthographic ([48, 110]) and perspective ([112]) cases.

The applicability of the last assumption is limited to the presence of texels, and on the capacity of an algorithm to identify and evaluate the information in each texel reliably. However texels are not frequent in natural images, which limits the applicability of the cited works.

So far, the assumptions that we have presented have not reached the level of generality that we would desire for an applicable SFT algorithm. In fact the last assumption is heavily limited by the presence and quality of the texels in the image. On the other hand, the isotropic assumption can find little applicability on natural textures (e.g. it fails on directional textures). Finally, even homogeneity, as we have presented it here (according to the early studies in SFT and therefore to

a spatial analysis), is limited by the presence of some texels and certain properties of them (density, aspect ratio, etc.). This kind of homogeneity has been named in [118] as a 'first order homogeneity'.

We need to review the definition of homogeneity to include the spectral analysis of textures. In this way, homogeneity would be a weak assumption applicable to textures such as sea waves, where it is not possible to distinguish single elements. A definition of homogeneity in the spectral analysis of textures is the subject of the next session.

2.2 Homogeneity and the Spectral Analysis of Textures

An approach based on the spectral analysis of textures was initiated by Bajcsy and Lieberman in 1976 ([6]) with the use of the Fourier power spectrum in local image windows. Spectral analysis in SFT proliferated in the nineties with a large number of works ([42, 54, 74, 118, 147, 158, 159]) giving notable results, and is now the main trend in SFT (e.g. [30, 98, 114, 121, 138]). Spectral analysis is mainly based on Fourier transforms (e.g. [114, 118, 139]) and Wavelet decomposition ([30, 60, 74]).

Spectral analysis is primarily concerned with the amplitude spectrum. The amplitude spectrum can describe the partitioning of energy in the image, according to spatial frequencies and orientations, and is invariant to small spatial shifts of the image.

Using spectral analysis and the amplitude spectrum, the texture distortion can be characterised by the affine distortion of spectrum elements, e.g. energy peaks ([138]), local moments ([159]), inertia ([114]). For example, Malik and Rosenholtz ([117, 118]) use the spectrogram of the local Fourier transform to compute the affine texture distortion between two patches of the textures. They argue that the affine texture distortion subsumes the different texture gradients and has enough information to recover the surface orientation and the curvature locally.

The homogeneity assumption in [117, 118] is not particularly about having a scene texture with a constant density, or texels with the same area or the same aspect ratio or the same appearance. According to Malik and Rosenholtz ([117, 118]), the homogeneity assumption is about having the 'same' texture at the different points of the surface in the scene. For deterministic texture, this would be seen as an assumption of periodicity. For stochastic textures, i.e. texture which can be considered to be realizations of stochastic processes, it can be formalized as stationarity under translation.

Formulated in this way, homogeneity allows us to take into consideration the particular properties of textures (e.g. density, aspect ratio, etc., which were already considered in the first order homogeneity assumption), the appearances of texels, and also stochastic textures not presenting specific texels. For example, the stochastic texture given by the dry dam with crack lines in figure 1.1 is homogeneous as long as the distribution of the cracks (direction, width, length, etc.) can be assumed to be generated by the same stochastic process which is stationary under translation. Many artificial (e.g. building facades) and natural scenes respect this homogeneity, which is considered the weakest and most applicable. This is the assumption of homogeneity that we make throughout this work.

A convenient way to analyse textures under the defined homogeneity, is via spectral analysis, e.g. by considering the affine distortion of such spectrum amplitude elements as energy peaks, local moments and inertia.

In this work, we characterize the texture distortion under the described homogeneity assumption by using the concept of *local spatial frequency* (LSF). As will be clear from chapter 3, a LSF parameterizes the information content of the texture at a particular frequency. For both a planar and curved surface texture, the LSF takes a different value at each point.

As the variation of this frequency is proportional to the local inclination, i.e. a variation of frequency is only introduced by the projection of the scene texture from the 3D world to the 2D image, it is natural to characterize the texture distortion by using the LSF (as also maintained by [120]). Furthermore, complex textures have usually more than one main spectral direction. Chapter 3 shows how each respective spectral direction can be parameterized by a LSF.

Our approach is similar to [42, 121, 158]. In particular, in his doctoral thesis ([120]), Massot names the variation of frequency a 'gradient of frequency', which seems a natural way to relate the newly defined homogeneity assumption with the first order homogeneity.

As argued in [120, 121], a frequency analysis is closer to the functioning of the human visual system, where one can identify a decomposition of the visual information into bands of frequencies and orientation.

2.3 Viewing Geometry

It is fundamental for the 3D reconstruction to define a viewing geometry, which is the basis for the mathematical relationship between the textured object surface and the observed image on the image plane. Also fundamental is the choice of the variables for indicating the orientation of the surface.

Let us assume for explanatory purposes that the texture object surface is planar.

As in all current works on SFT, we indicate the inclination of the surface with the *slant-tilt system*:

- Tilt is the direction on the image plane along which the distance to the viewed surface increases most rapidly;
- Slant is the degree of inclination of the surface, which determines how rapidly the image plane distance to the viewed surface increases along the tilt direction.

The adopted viewing geometry was introduced by Super and Bovik in [158] and remains today a commonly used setup (e.g. [42, 74, 121, 138]). In [158], it was defined for the case of a planar surface texture and a pin-hole camera model.

We explain it by referring to figure 2.2:

- the *world coordinate system* is indicated by $\mathbf{x}_w = (x_w, y_w, z_w)$, it has its origin coinciding with the focal point and the $-z_w$-direction coinciding with the optical axis;

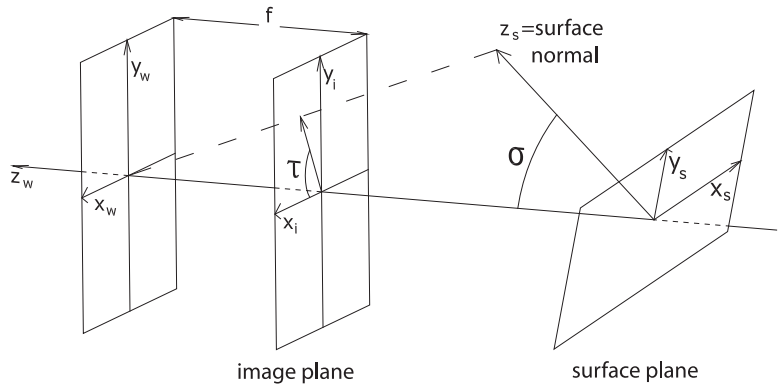


Figure 2.2: Viewing geometry

- the *image plane coordinate system* $\mathbf{x}_i = (x_i, y_i)$ is placed at $z_w = f < 0$, with $|f|$ being the focal length, such that $x_i = x_w$ and $y_i = y_w$.

We can then give a more rigorous definition of the slant and tilt, chosen to indicate the orientation of the surface plane:

- the *slant*, indicated with σ, is the angle between the surface normal and the optical axis, with values ranging from $-90°$ to $90°$;
- the *tilt*, indicated with τ, is the angle between the x_i-axis and the projection on the image plane of the surface normal, with values between $-180°$ and $180°$.

By using the orientation parameters (σ, τ), Super and Bovik ([158]) define a surface texture coordinate system to describe the surface:

- the surface texture coordinate system, indicated with $\mathbf{x}_s = (x_s, y_s, z_s)$, has got the x_s-axis aligned with the perceived tilt direction, the z_s-axis aligned with the surface normal, and y_s forming a right handed orthogonal coordinate system with x_s and z_s. The origin of $\mathbf{x}_s$ is on the intersection of the surface with the z_w-axis, at $z_w = z_0 < 0$.

As mentioned, the described viewing geometry is defined for a planar surface, but it can be easily extended to general surfaces. It is in fact possible to assume

that the surface is smooth and that at any point it can be locally approximated with the corresponding tangent plane, to which the viewing geometry will apply.

In subsequent chapters 5 and 6, we will deal with issues due to the fact that the $\mathbf{x}_s$ coordinate system changes with the tilt-direction, as the x_s axis is aligned with it at each point. As it would be desirable to have a consistent surface texture coordinate system $\mathbf{x}_s$ (i.e. $\mathbf{x}_s$ oriented along the same direction in a virtually unfolded surface texture), we propose in chapters 5 and 6 suitable rotations for alignment.

Much debated in SFT is whether a perspective or an orthographic camera model should be assumed. This is another assumption which could be limiting, depending on the particular problem, and will be discussed in the next section.

2.4 Assumptions on the Camera Model and on the Texture Surface

In SFT literature, we find three types of projections from the 3D world to the 2D image: orthographic projection ([159]), paraperspective projection ([2] and perspective projection ([52]).

In this thesis we consider the two most widely used models: the orthographic (or parallel) and perspective (or pin-hole) cameras. Both assumptions are approximations to real camera systems and to the human visual system.

In particular the *orthographic camera model* (e.g. used in [28, 159]) is suitable to model the scene when the viewing angle is small compared to the distance of the image plane to the surface, or equivalently when the variation of distance of the scene is small compared to the distance of the image plane to the scene. Under this assumption rays from the texture surface are projected onto the image plane in parallel, and slanting a surface (in a certain direction, i.e. the tilt direction) induces a foreshortening effect, which is a function of the slant angle. Furthermore, since the same result is obtained when slanting in a tilt direction and in a direction which differs from it by 180°, the orthographic camera model suffers from the well known tilt ambiguity.

The *perspective camera model* (e.g. used in [30, 78, 121, 158]) is the more realistic model, as it also takes into account the distance from the image plane and the presence of a focal point. However, the depth of field of a pin-hole camera model is infinite, whereas real camera lenses only have a limited depth of field. As also argued by Super and Bovik in [158], as long as the depth of field of the real system is large enough to encompass the surface region under analysis, the approximation is sufficient for our purposes.

In a perspective camera model the projecting rays converge to the focal point. As an effect of convergence, in addition to a foreshortening effect, in a perspective camera model we have a scale effect, due to the distance (i.e. closer objects to the image plane project to bigger images), and a position effect, due to the different angles of the projecting rays with respect to the optical axis (i.e. parallel lines on a planar texture surface converge in the image plane to a vanishing point).

Before comparing the orthographic and perspective camera models, it is worth mentioning another assumption about the shape of the considered textured surface. While earlier works focused on planar surfaces (e.g. [1, 59, 158]), more recently the interest shifted to curved surfaces (e.g. [30, 114, 118]. This is an important factor, as in general methods for planar textures cannot be extended to curved textures: 'there is a very restrictive class of texture distortion measures which are invariant with respect to surface curvature. Gibson's gradient of texture density does not belong to this class.' ([52]).

An assumption of perspective projection is required to study planar surfaces. In fact, homogeneity is preserved under affine transformations, and it therefore provides no information about the plane orientation under an orthographic projection model ([10, 34]). Furthermore many studies have investigated whether perspective effects are used by the human visual system ([3, 120, 128, 162]), and all of them agreed that these effects contribute to vision. In particular, in [120], Massot studies the importance of perspective effects in human vision in the case of planar texture surfaces compared to the importance of frequency variations (in his work 'frequency index'). He concludes that the perspective effects are particularly important in the discrimination of tilt, while frequency variations count more in the perception of slant.

On another note, other authors (e.g. [75, 110]) argue that the perspective effects are generally negligible when considering textured objects, as the range of variation of distances for the elements of the considered object is usually small compared to the distance of the object to the image plane. This translates into weak scale and position effects in the image, where the only reliable cue is then the foreshortening effect.

Throughout the work presented here, we consider both the orthographic and the perspective camera models. The former usually means easier mathematical equations, while the latter is required for defining a general algorithm, which can deal with both curved and planar surfaces. We keep in mind, however, the observation of [110] of negligible perspective effects for textured objects. This observation will be of crucial importance when determining a robust SFT algorithm, generally able to deal with both planar surfaces and textured objects (see chapter 6).

2.5 Integrating the Orientation Parameters to Recover the 3D Shape

SFT is aimed at reconstructing the shape underlying the observed texture. However most of the SFT algorithms (apart from those for planar surfaces) implement, as we do, algorithms for computing the local orientation parameters (σ,τ) at each pixel.

While it is easy to relate the local orientation to the local derivatives of the 3D shape, it is certainly not trivial to integrate these partial derivatives to recover the shape, in the form of its height field Z.

The former transformation is represented by the following equations, relating σ and τ to the gradients of the height field Z:

$$\frac{\partial Z}{\partial x} = -\tan\sigma \ \cos\tau \tag{2.2}$$

$$\frac{\partial Z}{\partial y} = -\tan\sigma \ \sin\tau \tag{2.3}$$

To then solve for the 3D shape, Z, a number of techniques exist (see [84, 85] for an overview).

Among such techniques, one approach is given by local integration along paths (e.g. [31, 66, 143, 175]): curve integrals on specific paths are solved to compute local height increments. While these methods are easy to implement and computationally efficient, they are highly dependent on the data accuracy and subject to noise propagation.

Global techniques treat the integration as an optimisation and achieve generally better results.

[70, 71] are examples of such methods. The authors consider the integration as a variational problem and solve it iteratively (although this raises problems about convergence).

In fact, notwithstanding a number of integration methods in the literature (e.g. [7, 14, 43, 73, 91, 126, 141, 142, 160]), the global integration technique which Frankot and Chellappa presented in 1988 ([50]) is widely used (e.g. [23, 57, 109, 134, 149, 154]), and is the method that we adopt here.

The method of Frankot and Chellappa imposes integrability by projecting the surface gradients onto a finite set of orthogonal and integrable functions. In particular, they show that a convenient set of integrable functions is given by the Fourier bases: in this case both the projection and the final integration are achieved by Fourier transforming, implementable by fast Fourier transform (FFT) algorithms.

In other words, the authors realise a least-squares fit of integrable gradients (the Fourier bases) to non-integrable gradients ($\partial_x Z$ and $\partial_y Z$) in one pass of the algorithm by using the FFT; then the shape is integrated by inverting the Fourier transform, i.e. inverse FFT.

The resulting method is elegant, simple and computationally efficient, and although it has been noted ([154, 170]) that it is sensitive to abrupt changes in orientation, this is usually not the case for the objects that one considers for SFT.

Finally, it is worth mentioning the work of Wei and Klette ([167, 168, 169, 170]). The authors introduce a surface curvature and area constraints into the functional to minimize. The two constraints, weighted by two corresponding regularization parameters λ and μ, seem to improve on the smoothness of the recovered 3D shape and on the robustness to noisy gradients. However a systematic study on the values to be chosen for λ and μ has not been presented yet.

2.6 Surface Texture Model

As we saw in the introductory chapter 1, a texture is generally difficult to describe unequivocally, one of the reasons being that it can be represented through different characteristics. For example, in [99] a texture is considered to be composed of 'vertical' elements and the SFT problem is solved by using self-occlusion, while [27, 89] use the corrugations of the texture to recover information about the illumination.

A common assumption in SFT (e.g. [30, 112, 158]) is to model texture as due to variations of surface reflectance, the proportion of incident light reflected. The surface is assumed to have a Lambertian reflection and the texture is supposed to be 'painted onto' it, without roughness or self-occlusion.

As indicated in [30], under the assumptions of Lambertian reflectance and painted texture, and given we have a developable surface (i.e. with zero Gaussian curvature), the surface reflectance R is related to the image intensity I by the following:

$$I(\mathbf{x}_i) = a(\mathbf{x}_i)R(\mathbf{x}_s(\mathbf{x}_i)). \tag{2.4}$$

In the above equation $\mathbf{x}_s(\mathbf{x}_i)$ expresses the back-projection of the image point $\mathbf{x}_i$ to the surface texture, while $a(\mathbf{x}_i)$ is a multiplicative shading term, used in shape from shading work ([107]). According to the homogeneity assumption that we made in section 2.2, R is a stationary stochastic process.

[30] shows how to estimate a from local averaging, by convolving the squared image, I^2, with a Gaussian filter. However, if the scale of variation of $\mathbf{x}_s$ is small compared to the scale of variation of the shading term, then the latter can be assumed to be constant in any small neighborhood. Moreover, as our method uses frequencies (as we describe in chapter 3) rather than amplitudes, it automatically normalizes for slow variations in illumination, shading and surface texture contrast. In fact in our work the images are analysed without any form of pre-processing, such as those used in [30, 121].

2.7 Previous Work on Shape-From-Texture

Texture was first recognized as a cue for the 3D perception of the world in the work of Gibson ([59]). In his seminal work in 1950, he coins the terms 'texel' and 'texture gradient' and attempts to recover the orientation of a planar surface uniformly covered with texels by using the density texture gradient.

The pioneering work of Gibson is followed by a number of studies relying on the detection of features in the texture, such as lines, edges, texels, etc., and on the use of texture gradients based on the detected features ([1, 13, 76, 77, 80, 81, 119, 155]).

Among the cited works, Stevens ([155]) discusses the feasibility of SFT. He assumes that the texels are nearly circular and illustrates interesting relations among the gradients and between these and the orientation of a planar surface. Aloimonos ([1]) follows the approach of Gibson using the density gradient and additionally the sum of texel contours. Marinos and Blake ([119]) use texels and homogeneity, assuming nearly constant density of texels on the surface and using the variation of density to compute the shape and orientation of the surface.

In fact, the cited early works mainly use what Malik and Rosenholtz ([118]) call 'first order homogeneity' (see section 2.1), the assumption of some uniform distribution or property of texels.

As mentioned in section 2.1, an assumption of a different nature is that of Witkin, who proposes the use of isotropy, meaning that the scene texture does not have a dominant direction or orientation bias. The work of Witkin ([174]) was followed by various other contributions ([9, 21, 53]) in which the local anisotropy is used as a measure of the local deformation. In these works, the local anisotropy is used to recover the local orientation directly without any need to integrate over the texture surface.

Isotropy is also the assumption effectively used by Gårding ([52, 53]), although his main contribution is primarily theoretical in the study of homogeneity: in [52] he derives the relationships between the changes in the texture gradients and the differential geometry of planar and curved surfaces.

As discussed in section 2.1, isotropy is an overly-restrictive hypothesis, and even more restrictive is the assumption of 'homotropy' by Stone. In [157] the

author assumes that the distribution of local vectors tangent to the contours must be invariant with position.

A different approach in SFT was started by Bajcsy and Lieberman in 1976. [6] is the first of a series of studies using spectral analysis and the homogeneity assumption. As discussed in 2.2, the trend is based on considering the variation of frequency as a distortion of a homogeneous texture due only to the projection from the 3D world to the image.

As we mentioned in section 2.2, a relevant number of studies were produced in the nineties, mainly based on Fourier transforms (e.g. [42, 94, 117, 118, 147, 148, 158, 159] and Wavelet decomposition (e.g. [74]).

In particular, of great impact were the works of Super and Bovik ([158, 158]), who employed a set of Gabor filters distributed over the whole image spectrum to estimate:

- a single LSF that they would use to recover the planar surface orientation based on minimization of variance ([158]);
- two canonical moments to recover first a frontal point of the curved textured surface, and then the surface itself ([159]).

Our work draws significant inspiration from these studies:

- our multi-scale algorithm for recovering the main frequencies of a texture is based on their observations of the properties of Gabor filters and how these can be used to recover an instantaneous frequency (see chapter 3 for greater detail and for an extensive comparison);
- we implement their algorithm for planar surface orientation estimation based on variance (see chapter 4), which constituted the test-bed for our frequency estimation method, and we make further use of the idea of using variance when more than two LSFs are available (see chapters 5 and 6).

We also note the contribution of Sakai and Finkel ([147, 148]), who base their method on the psychophysical observation that subjects seem to be sensitive to variations of peaks and average frequency.

Also of great importance was the contribution of Malik and Rosenholtz ([117, 118]), who introduce an affine texture distortion between image patches, which subsumes the different texture gradients and has enough information to recover the curvature locally. The spectra of the patches and the theoretical framework of Gårding are used to solve for the affine texture distortion, with convincing results on both synthetic and real textures. Most importantly, as mentioned in section 2.2, the assumption is made in [117, 118] that the 'same' texture covers the whole surface, rather than a more restrictive constant density or area of texels. In other words, 'homogeneity' becomes about assuming the texture to be a realization of a stationary stochastic process.

Also relevant to our work is the study of Dugué and Elghadi ([42]). Although their attempt to build a multi-scale frequency estimator appears very sensitive to noise, they present a very robust mathematical formulation of a 3D reconstruction algorithm which uses two LSFs to recover the shape by using the *singular value decomposition* (SVD). We implement and use their orthographic reconstruction algorithm in chapter 5, and extend it to deal with perspective cases.

Work on SFT, based on spectral analysis and on the new homogeneity assumption, has continued up until the recent time ([30, 98, 114, 121, 135, 138]).

In particular Ribeiro and Hancock ([135, 138]) observe the shift of peaks in the Fourier transform of image patches, although the noisy nature of the peak values may cause a correspondence problem, as also noted by the authors. Clerc and Mallat ([30]) use a new wavelet formulation with a geometrical framework similar to [52] and [118], obtaining convincing results. It is also worth noting the study of Loh and Kovesi ([114]), who use the inertia about the tilt direction and about the orthogonal direction to it, to determine the affine transformation between patches. However, [114] lacks a proper method to determine the position of a frontal point, which would be needed to ultimately reconstruct the shape automatically. In fact, their automatic implementation is based on detecting the frontal point with the algorithm of [159], which is sensitive to noise, as also maintained by Massot ([120]).

At the moment of writing this dissertation, we consider that the state of the art in SFT is expressed by the work of Massot and Hérault. They draw on studies of the physiology of the visual cortex and psychophysical experiments

(e.g. the works of Li and Zaidi in [100, 101, 102, 103, 104, 105]) to define a method for extracting a single instantaneous frequency from the image, and derive a 3D reconstruction algorithm which uses the local frequency variations to recover the shape in the perspective case. Their method is effective and its application is tested on many natural textures convincingly. In particular their 3D reconstruction algorithm has been a significant inspiration for developing the 3D reconstruction algorithm described in chapter 6. However, their method, based on a single estimated frequency, does not seem to be robust enough to deal with general shapes. Chapter 6 will treat this issue with details, showing convincing examples of how our proposed algorithm can be considered a further step in the state of the art of SFT.

Finally, we mention the contributions of Forsyth and Lobay ([48, 110]), and Loh and Hartley ([112]). The authors achieve quality results by using the single texel appearances to estimate affine transformations. As mentioned in section 2.1, the assumption that there are texels of similar appearance and that it is possible to identify and evaluate them seems to us too restrictive, especially in the case of natural texture.

Another final note is about the *scale*, i.e. the size of the considered image patches. The scale has been investigated by various authors (e.g. [96, 136, 156]). As will be clear in chapter 3, our algorithm automatically chooses the appropriate scale from the Fourier analysis of the image.

2.8 Contribution

Our work is likely to impact in the SFT field in a number of ways. Most notably we introduce:

- a multi-scale distortion estimation algorithm for extracting all the main LSFs from the image texture. Our novel method is unique, as other algorithms for SFT can only estimate a single distortion measure ([121, 158]) or at most consider two preferred directions in the spectral domain ([114, 159]). In fact, in many cases there is not a dominant LSF, and assuming so can

affect the distortion estimation, or the texture is rich and has more significant spectral components (as will be shown in chapter 3) and not using all of them would seem to be not using all the available information;

- different 3D reconstruction algorithms, which solve the SFT problem by using the estimated LSFs in increasingly general and challenging scenarios: an algorithm for estimating the orientation of a planar surface (see chapter 4); an algorithm for recovering a developable shape showing a frontal point and another which extends this to non-developable shapes (see chapter 5); a general algorithm which computes the 3D shape of curved and planar surfaces locally without any further assumption (see chapter 6).

Materials from this thesis have been published in the following papers:

- Fabio Galasso and Joan Lasenby, "Fourier Analysis and Gabor Filtering for Texture Analysis and Local Reconstruction of General Shapes", in the Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Miami, FL, June 2009
- Fabio Galasso and Joan Lasenby, "Fourier Analysis and Gabor Filtering for Texture Analysis and Local Reconstruction of General Shapes", Technical Report, Department of Engineering, University of Cambridge, CUED/F-INFENG/TR-610, 2008
- Fabio Galasso and Joan Lasenby, "Shape from texture via Fourier analysis", in the Proceedings of the International Symposium on Visual Computing, pages 803-814, Las Vegas, NV, December 2008
- Fabio Galasso and Joan Lasenby, "Shape from texture of developable surfaces via Fourier analysis", in the Proceedings of the International Symposium on Visual Computing, pages 702-713, Lake Tahoe, CA/NV, November 2007
- Fabio Galasso and Joan Lasenby, "Shape from texture: fast estimation of planar surface orientation via Fourier analysis", in the Proceedings of the British Machine Vision Conference, pages 660-669, University of Warwick, UK, September 2007

Chapter 3

A Multi-Scale Distortion Estimation

The *Fourier transform* of an image is known to supply general information about the frequency content of the image. It shows the range of the 2D spatial frequencies contained in the image. Furthermore the Fourier transform can be computed efficiently by the use of FFT algorithms.

On the other hand, the instantaneous value of one of the image frequencies at a pixel is usually estimated by evaluating the spatial-frequency spectrum locally, using windowed Fourier transforms ([51]), Wigner distributions ([29]) or Wavelet transforms ([38]).

This chapter presents an original algorithm which estimates the instantaneous frequency value at each pixel for each of the main frequencies in the image. The image frequencies, indicated as *local spatial frequencies* or LSFs, are recovered by using *2D Gabor functions* and their first derivatives. Fourier analysis is used to achieve fast setup of the Gabor filters and of their derivatives and to optimise the computational expense of convolutions. Furthermore the Fourier transform is also used to identify all the main LSFs in the image, so as to set up different groups of Gabor filters and to recover these LSFs.

The computed LSFs characterize a *multi-scale distortion estimation*, which makes our algorithm unique, as other algorithms for SFT can only recover one dominant frequency ([121, 158]) or at most consider two preferred spectral directions ([114, 159]). In fact many textures have a rich spectrum and they may

not have a single dominant frequency. To suppose that the texture has a single dominant frequency can therefore impair the frequency estimation; our algorithm is therefore a step forward in that it uses all the available information.

Inspiration for the development of the algorithm was originally drawn from the work of Super and Bovik ([158]), who show for the first time how to compute a frequency at a point by using a Gabor filter and its derivatives. Our method extends the original algorithm, introduces a multi-scale approach and optimises the computational expense.

The algorithm presented here supplies the distortion estimation. Chapters 4, 5 and 6 use this information to recover the shape.

3.1 A Method to Estimate Instantaneous Frequencies

Estimating an instantaneous frequency usually requires a local analysis of the spatial-frequency spectrum.

We consider a texture image $I(\mathbf{x})$ and a band-pass filter $h(\mathbf{x}, \mathbf{u})$, which is a function of a particular point $\mathbf{x} = (x, y)$ in the image and of a central frequency $\mathbf{u} = (u, v)$. We usually call the output of the convolution, $F(\mathbf{u}, \mathbf{x}) = I * h$, the local spectrum of the image I in the neighbourhood of the image point $\mathbf{x}$ and the frequency $\mathbf{u}$.

Ideally we would like to use a filter h which maximises both the spatial localisation about the considered point $\mathbf{x} = (x, y)$ and the spectral localisation about the considered frequency $\mathbf{u} = (u, v)$. However the uncertainty principle ([51]) states that there is a non-zero lower bound for the measure of a frequency $\mathbf{u}$ and its position $\mathbf{x}$.

A plausible choice for the filter seems then to be the use of 2D Gabor functions, because they satisfy that lower bound exactly, maximising both the spatial and frequency localisation.

2D Gabor functions are defined by

$$h(\mathbf{x}, \mathbf{u}) = g(\mathbf{x}) e^{2\pi j \mathbf{x} \cdot \mathbf{u}} \ , \tag{3.1}$$

with j the imaginary unit and

$$g(\mathbf{x}) = \frac{1}{2\pi\gamma^2} e^{\frac{-(\mathbf{x}\cdot\mathbf{x})}{2\gamma^2}} \ , \tag{3.2}$$

a 2D Gaussian function with variance γ^2.

2D Gabor functions can therefore be seen as 2D Gaussian functions modulated by a sinusoid of frequency $\mathbf{u}$, or alternatively, considering that the Fourier transform of a Gaussian is still a Gaussian, as 2D Gaussians centred on the frequency $\mathbf{u}$, i.e. band-pass filters of size determined by the variance γ^2 about the central frequency $\mathbf{u}$.

Super and Bovik show in their publications ([16, 65, 158, 159]) that an instantaneous frequency can be estimated (i.e. $\tilde{\mathbf{u}}(\mathbf{x})$) at an image point $\mathbf{x}$ by using a 2D Gabor filter $h(\mathbf{x}, \mathbf{u})$ and its two first order spatial derivatives $h_x(\mathbf{x}, \mathbf{u})$ and $h_y(\mathbf{x}, \mathbf{u})$ in x and y:

$$|\tilde{u}(\mathbf{x})| = \frac{|h_x(\mathbf{x}, \mathbf{u}) * I(\mathbf{x})|}{2\pi|h(\mathbf{x}, \mathbf{u}) * I(\mathbf{x})|} \tag{3.3}$$

$$|\tilde{v}(\mathbf{x})| = \frac{|h_y(\mathbf{x}, \mathbf{u}) * I(\mathbf{x})|}{2\pi|h(\mathbf{x}, \mathbf{u}) * I(\mathbf{x})|} \ . \tag{3.4}$$

The ambiguity, due to the fact that all quantities are positive, is solved by choosing the signs of $\tilde{u}(\mathbf{x})$ and $\tilde{v}(\mathbf{x})$ so as to minimize $|\tilde{\mathbf{u}} - \mathbf{u}|$.

As noted by the cited authors, this method has the advantage of being independent from the amplitude of the texture at the point. As it considers the frequencies, it therefore automatically normalizes for slow variations of illumination, shading and surface texture contrast.

Also to be noted is that the estimate can be assumed to be correct if the frequency we are measuring is in the pass-band of the filter. The question therefore arises of how to choose the central frequencies $\mathbf{u}$ and the spatial constants γ of the Gabor functions, i.e. their centres and widths, so as to be assured that the instantaneous frequency we need to estimate is inside the pass-band of the filter.

Let us assume that $\mathbf{u}_p$ is the instantaneous frequency value that we are to estimate at the pixel coordinate $\mathbf{x}_p$. Let us also indicate with $|F(\mathbf{x})|$ the absolute value of the result of the convolution of the image I with the Gabor filter h, with central frequency $\mathbf{u}$ and spatial constant γ. In particular, we consider $|F(\mathbf{x}_p)|$

the absolute value of the convolution result at the same pixel coordinate $\mathbf{x}_p$. In principle, the closer the instantaneous frequency that we are to measure $\mathbf{u}_p$ is to the central frequency $\mathbf{u}$ of the Gabor filter, the greater is the absolute value of the result of the convolution at the pixel $|F(\mathbf{x}_p)|$. However we also have to take into consideration the observed decrease in energy as we move from lower to higher frequencies in the spectrum, which is generally assumed to follow the law $\frac{1}{f^2}$ ([5]).

As in [158, 159], we use filters with constant *fractional bandwidth* (bandwidth divided by central frequency) of one octave. By doing so, to a good approximation we can assume that the filters contain the same amount of spectral energy, which allows us to compare their output and choose the filter which is best centred on the frequency to be measured.

Using filters with constant fractional bandwidth also has the advantage of measuring higher frequencies with better localization than lower frequencies and of being computationally less expensive compared to a method using filters with constant γ.

The following sections address the two issues raised in this section: how to set up the Gabor filters which isolate the main frequencies of the texture and how to select the appropriate frequency estimate at each point among those available for a particular LSF.

3.2 The Setup of Gabor Filter Parameters

Equations 3.3 and 3.4 in section 3.1 provide expressions to compute an instantaneous frequency estimate $\tilde{\mathbf{u}}(\mathbf{x})$ at an image point $\mathbf{x}$. As explained in section 3.1, it is important that the frequency being estimated is inside the pass-band of the filter.

The problem can be visualized graphically by considering a circle in the spectral domain: its centre represents the central frequency $\mathbf{u}$ of the filter and its radius represents the pass-band of the filter (with the circumference indicating the -3dB cutoff frequency), a function of the variance γ^2. Under this graphical convention, the frequency which is to be estimated must fall within the drawn circle, and each circle only gives a valid estimate of the frequencies inside it.

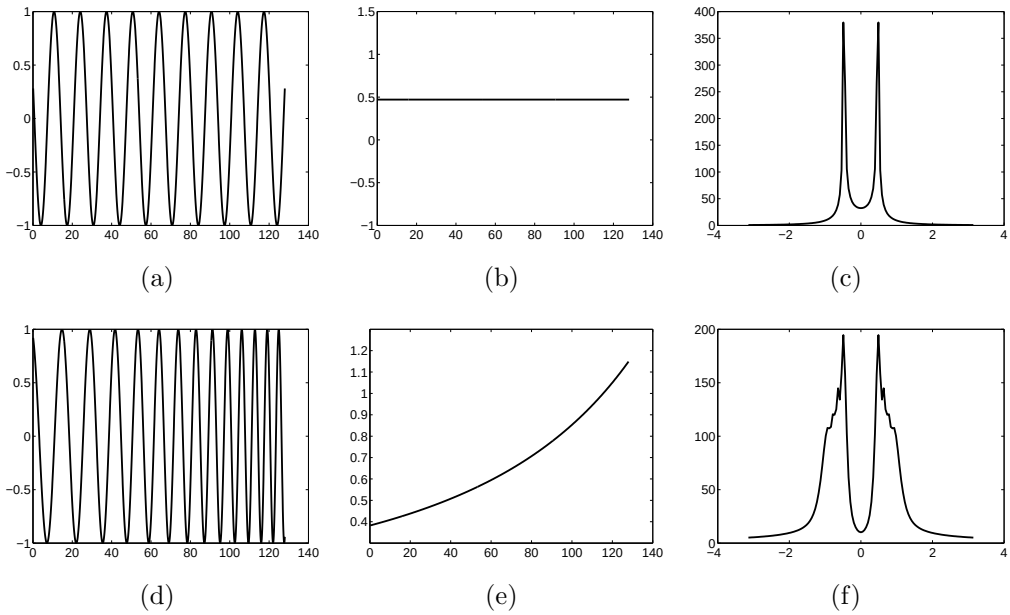


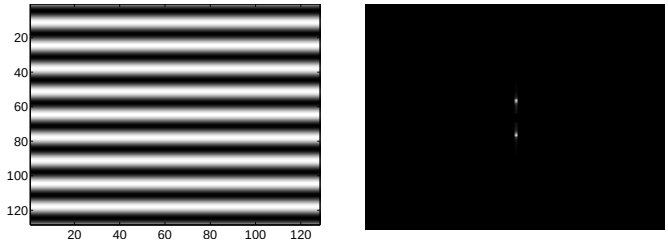
Figure 3.1: (a) 1D cosine at frequency $\tilde{u} \approx 0.47$ rad/s; (b) its instantaneous frequency; (c) its spectrum amplitude; (d) 1D cosine with frequency varying from $\tilde{u} \approx 0.38$ rad/s to $\tilde{u} \approx 1.15$ rad/s; (e) its instantaneous frequency; (f) its spectrum amplitude

We adopt this graphical convention in the following subsections. There we show how to use the Fourier transform of an image to set up the Gabor filters, so that the frequencies to be recovered are inside the pass-band of the filters. Furthermore, the following subsections also show how the Fourier transform of the image can be used to identify the main LSFs, so as to set up distinct groups of Gabor filters for each of them and recover different instantaneous frequencies.

A comparison is supplied with the method for LSF estimation used by Super and Bovik in [158, 159], which emphasises the lower computational expense and the multi-scale nature of our algorithm.

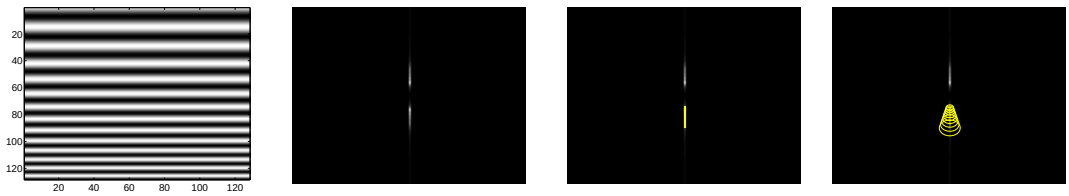
3.2.1 The case of a single LSF

Let us consider the case of a one dimensional cosine. Figure 3.1(a) represents a cosine which has a length of 128 samples and a constant instantaneous frequency of



(a) (b)

Figure 3.2: (a) 2D cosine at frequency $|\tilde{\mathbf{u}}| \approx 0.47$ rad/s; (b) its spectrum amplitude



(a) (b) (c) (d)

Figure 3.3: (a) 2D cosine with frequency varying from $|\tilde{\mathbf{u}}| \approx 0.38$ rad/s to $|\tilde{\mathbf{u}}| \approx 1.15$ rad/s; (b) its spectrum amplitude; (c) true LSFs (*yellow area*) represented on the spectrum; (d) chosen Gabor functions (*yellow circles*) superposed on the spectrum

$\tilde{u} \approx 0.47$ rad/s (following convention and assigning the value π to the largest admissible frequency in a signal of that length), equivalent to $\tilde{u} \approx 9.6$ cycles/signal. Figure 3.1(b) represents its constant instantaneous frequency while figure 3.1(c) represents the amplitude of the cosine spectrum. We notice that the spectrum amplitude is characterized by two symmetric spikes at the corresponding value (positive and negative) of the instantaneous frequency (≈ 0.47 rad/s), and the small spread is mainly due to quantization and truncation to 128 samples.

Figure 3.1(d) shows a cosine with continuously varying frequency, also known as a chirp. The instantaneous frequency of the cosine, ranging from $\tilde{u} \approx 0.38$ rad/s to $\tilde{u} \approx 1.15$ rad/s, is represented in figure 3.1(e), while its spectrum amplitude is shown in figure 3.1(f). As for to the previous case, we notice that the amplitude of the spectrum of this second cosine presents significant non-zero values exactly in correspondence with its range of frequencies. Here and in the following we indicate with the term 'significantly non-zero' the set of values of the spectrum amplitude greater than a threshold. The threshold is determined locally as a portion of the local maximum. In other words we write 'significantly non-zero values', as we consider values of the spectrum amplitude which stand from the local average, which is negligible with respect to these values.

Analogously, we consider a 2D image generated by a 2D cosine with constant instantaneous frequency $|\tilde{\mathbf{u}}| \approx 0.47$ rad/s, in figure 3.2(a), and the amplitude of its spectrum in figure 3.2(b). The latter shows two symmetric spikes in the corresponding positions of the instantaneous frequency value in the 2D frequency plane.

Then, as for the 1D case, we consider a 2D image generated by a 2D cosine with continuously varying frequency. The new 2D cosine, represented in figure 3.3(a), has frequency varying from $|\tilde{\mathbf{u}}| \approx 0.38$ rad/s to $|\tilde{\mathbf{u}}| \approx 1.15$ rad/s, the same range as in figure 3.1(d). In fact the image is a 2D version of the 1D case in figure 3.1(d). In particular, this new image was obtained by slanting the cosine of figure 3.2(a) through 40°. The amplitude of the Fourier transform of the considered cosine is illustrated in figure 3.3(b).

We aim to back-map all the instantaneous frequency values for all the pixels of the image in figure 3.3(a) onto its spectrum amplitude in figure 3.3(b). Since the image in figure 3.3(a) is synthesized, we know the instantaneous frequency values

at all pixels. For each pixel we take the value of the instantaneous frequency and we represent it on the amplitude spectrum of the image with a single yellow dot. This is done repeatedly for all pixels, and the result is illustrated in figure 3.3(c), which we call the back-mapping of the instantaneous frequency values onto the spectrum amplitude of the image.

As the frequency variation in the image is continuous, we obtain a continuous yellow area in the spectrum, which represents all the frequencies present in the image. As previously observed, the yellow area corresponds to the area of the spectrum amplitude which is significantly non-zero. This confirms that the spectrum supplies information on the range of frequencies contained in the image, and that significantly non-zero areas in the spectrum indicate which 2D frequencies are contained in the texture.

The algorithm we propose analyses the spectrum of the texture to determine its areas of interest, i.e. areas with significant non-zero values, and then uses this information to define the Gabor filters to use. In the particular case of the image in figure 3.3(a), two areas are identified in the spectrum (the significantly non-zero areas, corresponding to the ground truth yellow areas). Of these two we consider one because of symmetry. Then we set a group of Gabor filters on the area so as to cover it all, as illustrated in figure 3.3(d). As indicated in section 3.1, equations 3.3 and 3.4 require that the frequency to estimate be inside the pass-band of the filter being used. Covering the frequency range of the image with a group of Gabor filters as in figure 3.3(d) ensures that we have at least a filter for each instantaneous frequency value of the LSF under consideration. Furthermore the significant overlapping which one may observe in the figure serves to produce a robust LSF estimation, as we explain in section 3.3. From the point of view of the algorithm, setting a group of Gabor filters to cover the detected spectral area corresponds to choosing a set of central frequencies $\mathbf{u}$ (the centres of the circles) and a set of spatial constants γ (their radii).

As mentioned in section 3.1, we see that the deployed filters have constant fractional bandwidth of one octave. This allows us to compare their outputs prior to applying equations 3.3 and 3.4, as we explain in section 3.3.

For comparison purposes, figure 3.4 shows the setup of Gabor filters that Super and Bovik use in [158, 159]. This setup consists of Gabor functions arranged on

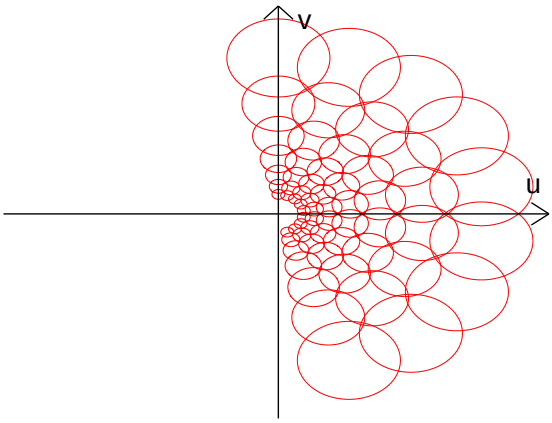


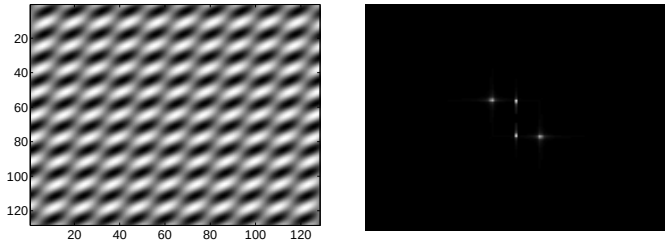
Figure 3.4: Lattice of Gabor filters used by Super and Bovik in [158, 159]

9 radial rays, 7 on each ray, separated by 20°. The central frequencies of the first filter of each ray are 6 cycles/image and increase by a factor $\sqrt{2}$ away from the origin. Not knowing the range of values of the frequency to estimate, Super and Bovik arrange their filters to cover all the spectrum (as above, we consider only half of the spectrum due to symmetry).

One clear advantage of our new technique is therefore a significant reduction of the computational expense. In [158, 159] 63 Gabor filters are used, as it can be seen from figure 3.4. These 63 Gabor filters correspond to 378 convolutions. In fact, as it can be noted from equations 3.3 and 3.4, the image is convolved with the Gabor filters and with their first order derivatives. Then the convolution results, 63x3 images, are further convolved with a Gaussian filter with spatial constant γ one twelfth of the image to post-smooth the previous convolution results. The procedure accounts for a total of 6 times more convolutions than the number of filters, exactly 378 for the complete lattice in [158, 159].

Our algorithm in this case uses 9 Gabor filters, corresponding to 27 (9x3) convolutions of the filters with the image, plus another 27 for post-smoothing. 54 convolutions are computed in total instead of 378, with a computational saving of about 86%.

More details on the frequency computation and the post-smoothing are given in section 3.3.



(a) (b)

Figure 3.5: (a) Two 2D cosines superposed, at frequencies $|\tilde{\mathbf{u}}_1| \approx 0.47$ rad/s and $|\tilde{\mathbf{u}}_2| \approx 0.72$ rad/s, 45° apart in the spectral domain; (b) its spectrum amplitude

(a) (b) (c) (d)

Figure 3.6: (a) Result of slanting the texture in fig. 3.5(a) by 40°; (b) its spectrum amplitude; (c) true LSFs (*yellow areas*) represented on the spectrum; (d) chosen Gabor functions (*yellow circles*) superposed on the spectrum

A second advantage will be clearer from the next section 3.2.2, where we demonstrate how our algorithm can cope with the concurrent presence of more LSFs by setting different groups of Gabor filters, a solution which is impossible using the method of Super and Bovik.

3.2.2 The case of two LSFs

Let us now consider the image in figure 3.5(a), where we can clearly observe two distinct LSFs. On the cosine from the previous example (frequency $|\tilde{\mathbf{u}}_1| \approx 0.47$ rad/s) we have superposed another cosine, with frequency $|\tilde{\mathbf{u}}_2| \approx 0.72$ rad/s, separated by 45° degrees from the first in the frequency plane. The amplitude of the spectrum of the image is given in figure 3.5(b), showing four peaks (but only half the spectrum will be considered because of symmetry) in the 2D frequency

plane, exactly corresponding to the frequencies of the cosines. For this image we can associate two instantaneous frequencies, i.e. two LSFs, to each point and these two frequencies in fact coexist at every pixel.

We apply the same slant of 40° as used in figure 3.3 to the image. Figure 3.6(a) shows the result: each cosine has now a continuously-varying frequency. Moreover the two LSFs change independently from each other, i.e. the first takes the same values as in figure 3.3 and the second equivalently acquires a range of 2D frequencies varying in the direction of the slant. Analogously to figures 3.3, figure 3.6(b) illustrates the amplitude of its Fourier transform, four non-zero areas corresponding to the spikes of the non-slanted cosine. By comparing figure 3.6(b) with figure 3.5(b) we observe that the two areas of interest are associated with the original cosine spikes, while the spread indicates a variation or distortion due to the slant.

Figure 3.6(c) shows the back-mapping of all the instantaneous frequency values for all the pixels of the image in figure 3.6(a) onto its spectrum amplitude in figure 3.6(b). As in the case of figure 3.3, the image is synthesized, so the ground-truth instantaneous frequency values for all the pixels are known. However, differently from the previous case of a single cosine, in the image of two superposed cosines in figure 3.6(a) each pixel has an instantaneous frequency value for each of the two LSFs, i.e. for each of the two cosines. For back-mapping the frequencies, for each pixel we take the two instantaneous frequency values corresponding to the two LSFs and we mark two dots on the amplitude spectrum of the image. Repeating this for each pixel yields the two yellow areas of figure 3.6(c), the back-mapping of the instantaneous frequency values onto the spectrum amplitude of the image.

As in the case of figure 3.3(c), as the frequency variation in the image is continuous, we obtain two continuous yellow areas in the spectrum. These two areas represent all the frequencies present in the image, pertaining to the two different LSFs. In other words, the significantly non-zero values of the spectrum amplitude of the image represent the range of variation of frequencies of the image, as already discussed in section 3.2.1.

Most importantly, a contiguous area identifies all the image frequency content pertaining to a particular LSF, i.e. all the instantaneous frequency values which

a particular LSF takes at the different image pixels. Distinct contiguous areas can therefore be used to identify the number of usable LSFs.

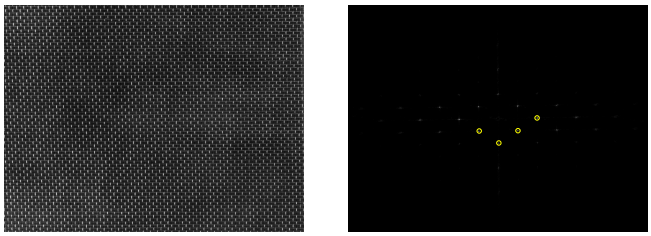
As already outlined in section 3.2.1, the proposed algorithm analyses the spectrum amplitude of the texture to determine the distinct contiguous areas of interest, i.e. the areas with significant non-zero values. In the case of figure 3.6, two such areas are identified (as in the previous example, half of the spectrum is considered due to symmetry). For each area a group of Gabor filters is set up, so as to cover all the relevant frequency values of the area with filters, as illustrated in figure 3.6(d). The two groups of Gabor filters correspond to two sets of values for the central frequencies $\mathbf{u}$ and the spatial constants γ. Furthermore the two so determined groups of filters are used separately to recover the instantaneous frequency values at all pixel for the two LSFs, as we explain in more detail in section 3.3.

By comparing the performance of our method in this case with [158, 159], we note that now 18 Gabor filters are set in total for the two groups, meaning 108 (18x6) convolutions. Although more computation is required in this case, due to a larger amount of information in the image (in fact twice as much), still the number of convolutions required is definitely smaller than the 378 convolutions of [158, 159], giving in our case a computational saving of about 71%.

Furthermore, our algorithm is capable of identifying the presence of two LSFs (or more, as the next section shows) and organizing different groups of Gabor filters, so as to recover both LSFs in the image. This is clearly not possible in the method of [158, 159] because Super and Bovik only use one group to cover the whole spectrum. As explained further in section 3.3, when one uses the method of [158, 159], the presence of more LSFs can impair the frequency estimation. This is undesirable since many natural textures have more than one LSF.

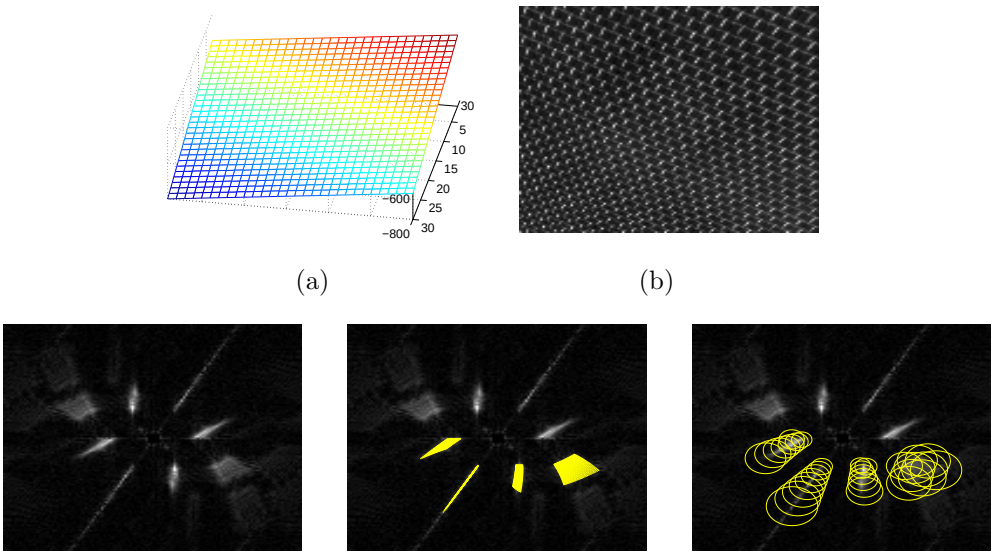
3.2.3 The case of a Brodatz texture

We now apply the described method for configuring the Gabor filters to a real image from the Brodatz database ([18]). Let us consider the texture D6 and its spectrum amplitude, shown in figures 3.7 (a) and (b) respectively. From the spectrum we deduce that the texture is mainly characterized by four LSFs,



(a) (b)

Figure 3.7: (a) D6 Brodatz texture; (b) its spectrum amplitude



(a) (b)

(c) (d) (e)

Figure 3.8: (a) 3D slanted plane; (b) D6 rendered on the slanted plane of (a); (c) spectrum amplitude of (b); (d) true LSFs (*yellow areas*) represented on the spectrum; (e) chosen Gabor functions (*yellow circles*) superposed on the spectrum

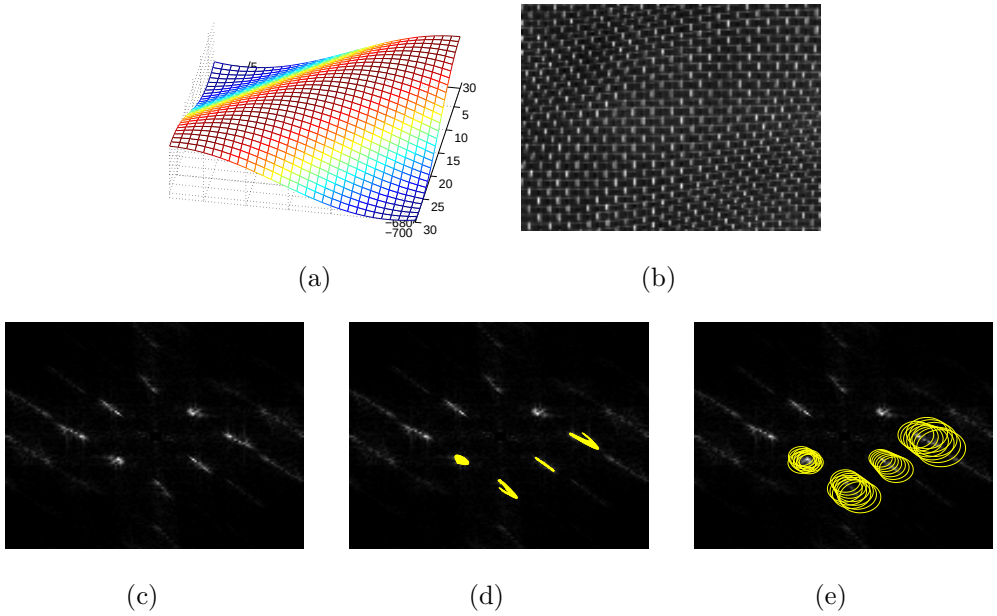


Figure 3.9: (a) 3D cosine surface; (b) D6 rendered on the cosine surface of (a); (c) spectrum amplitude of (b); (d) true LSFs (*yellow areas*) represented on the spectrum; (e) chosen Gabor functions (*yellow circles*) superposed on the spectrum

marked with small yellow circles (only half of the spectrum need be considered due to symmetry). By slanting and rotating the texture (figure 3.8(b), obtained by texturing the D6 texture onto the 3D planar surface of figure 3.8(a)), the four dots of figure 3.7(b) become corresponding areas in the spectrum (figure 3.8(c)). Figure 3.8(d) shows the back-mapping of the ground-truth instantaneous frequency values for each pixel for the four LSFs onto the spectrum amplitude. As in previous examples, the image is synthesized and the frequency values are known at each pixel of the image. Here four frequency values are in fact associated to each pixel. Again, we see that significant non-zero values in the spectrum amplitude represent corresponding frequencies of the image, and that distinct contiguous areas correspond to distinct LSFs. The four areas are used to set four distinct groups of Gabor filters. As illustrated by the four groups of yellow circles in figure 3.9(e), the filters are set up so as to cover the relevant identified areas. The procedure corresponds to setting four distinct sets of central frequencies $\mathbf{u}$ (centres of the circles) and spatial constants γ (their radii).

This also applies to figure 3.9(b), where the texture D6 is rendered on a more complex surface, the cosine of figure 3.9(a). In fact its spectrum, figure 3.9(c), shows areas corresponding to the original four LSFs of the spectrum in 3.7(b). Figure 3.8(d) represents the back-mapping (yellow areas) of the ground-truth instantaneous frequencies of each of the image pixels for the four LSFs on the spectrum amplitude. As in the previous case, significant non-zero values represent corresponding frequencies of the images, and distinct contiguous areas correspond to the distinct LSFs. Figure 3.9(e) therefore shows how we set four distinct groups of central frequencies $\mathbf{u}$ (centres of the circles) and spatial constants γ (their radii) in this case.

In both the examples in this section, as in the previous two, a significant overlapping of the filters serves to increase the robustness of the estimation. Furthermore the number of filters (and consequently convolutions) varies with the complexity of the image, but generally fewer are necessary than if sampling the whole spectrum as in [158, 159]. In particular, in the examples of this section, 32 and 38 Gabor filters are used respectively, meaning a total of 192 and 228 respective convolutions. Also in this case our algorithm saves a considerable amount of computational expense, about 49% and 40% respectively.

However we note here that the increased computational cost of our algorithm in this case results from recovering four times the information that the method of [158] could recover. Instead of recovering a single dominant frequency, where it is not even possible to define the meaning of dominant, all the four available LSFs can be recovered with our technique. In this way, the multi-scale nature of textures is addressed.

Finally we note that in the spectra represented in figures 3.7, 3.8 and 3.9, the zero and lowest frequencies were removed. This operation is performed before detecting the interest areas in the spectrum. In fact, both the zero frequency, corresponding to the average value of the pixels, and the low frequencies, corresponding to variations slower than 6 cycles per image, are very poorly informative and insufficient to use in a SFT algorithm, as also maintained in [158].

Similarly, the other represented spectra in this thesis have their lower frequencies suppressed, which also makes it possible to distinguish the areas of interest in them more easily.

3.3 Computing All the Main LSFs

Section 3.2 has introduced a novel method to identify the different usable LSFs in the texture, and to provide for each of them a set of Gabor filters which covers all the frequency variations for the respective LSF.

In particular, at each pixel of the image, each instantaneous value of a particular LSF is contained in the pass-band of at least a filter pertaining to the group of filters for that LSF.

This section addresses the question of the choice of the best filter (in the pertaining group) for each pixel for each LSF. Two important points, already discussed in section 3.1, are recalled: (i) the frequency to estimate must be in pass-band of the filter being used for the estimation; (ii) the absolute value of the result of the convolution of a Gabor filter with the image is greater when the central frequency of the filter $\mathbf{u}$ is closer to the true instantaneous frequency at the pixel (the one being estimated). The two points are also maintained by Super and Bovik in [158].

A solution to the question must take into account that the energy of the Fourier spectrum decreases from lower to higher frequencies, following on average the law $\frac{1}{f^2}$ ([5]). Gabor filters with the same variance γ^2, i.e. covering the same area on the Fourier spectrum, would therefore sample different portions of energy depending on their position on the spectrum, with filters in lower frequencies sampling more energy.

In order to deal with the latter issue, as already mentioned in section 3.1, we use Gabor filters with constant *fractional bandwidth* (bandwidth divided by central frequency), a solution already adopted by Super and Bovik in [158]. By doing so the spectrum energy sampled by the filter is independent of the position in the spectrum but only depends on the relative distance between the centre of the filter and the frequency to estimate.

The choice of the best Gabor filter at each pixel for each LSF is a consequence of the former two points. The choice is done by comparing the absolute value of the result of the convolution of the relevant group of Gabor filters with the image. In fact each group of Gabor filters is convolved with the image. Then at each pixel the greatest absolute value of such convolution results determines the best Gabor filter of the group for the pixel. A Gabor filter is therefore chosen for each group (pertaining to each LSF) at each pixel.

At each pixel, the instantaneous frequency value pertaining to each LSF is estimated by using the chosen Gabor filter in the corresponding group. The selected filter and its first derivatives are used to estimate the LSF at the pixel according to equations 3.3 and 3.4.

Finally, the same processing is repeated for all the available groups of filters, i.e. for all the detected areas in the spectrum amplitude. Instantaneous frequency values are estimated at each pixel for all of the groups, i.e. for all the LSFs.

The procedure, summed up in algorithm 1, estimates the value of all the identified LSFs at each image pixel.

Experimental results show that it is convenient to post-smooth the output of the convolution between the Gabor filter and the image with a Gaussian filter of spatial constant one twelfth the size of the image, before comparing the convolution results pertaining to the same LSF ([158]). By doing so, we reduce the noise. Furthermore and most importantly, the post-smoothing can be seen as a

Algorithm 1 Structure of the proposed algorithm to estimate the multi-scale distortion of a texture image

- The amplitude of the spectrum of the observed image texture is analyzed and areas of interest are detected, corresponding to the main LSFs in the texture. This can be done efficiently by Fourier transforming the observed image texture with a 2D FFT algorithm and then processing the image to detect non-zero contiguous areas.
- Each area determines a set of Gabor filters, so that the filters cover the whole group of frequencies pertaining to the particular area, as in figures 3.3, 3.6, 3.8 and 3.9.

for each set of filters **do**

- convolve the Gabor filters and their derivatives with the image, and smooth the outputs with a Gaussian filters;
- select the Gabor filter with largest amplitude output at each point;
- compute the (signed) instantaneous frequencies at each point using equations 3.3 and 3.4.

end for

way to 'encourage' the choice of the same filter for nearby pixels, so as to favour continuity in the reconstructed LSFs.

A further note is again addressed to the algorithm of [158] which uses one only lattice of Gabor filters covering all the spectrum. Let us consider in particular the case when a texture is characterized by more than one LSF and a dominant LSF cannot be defined (which is frequently the case for natural textures). The maximum of the absolute values of the convolution results at a pixel determines the choice of the Gabor filter to use for the frequency estimation at the pixel. In our algorithm we separate the groups of filters pertaining to the different LSFs and choose each maximum (at each pixel) inside the group. However, in [158] the authors only consider one group, given by all the filters of the lattice. Consequently, in [158] the chosen Gabor filter (the chosen maximum) for the frequency estimation may pertain to any of the LSFs present in the texture. As there is not a further criterion in the filter choice, apart from the convolution result absolute value, at different pixels the chosen Gabor filter may pertain to different

LSFs. Furthermore, as described in section 3.1, this may result in a correct frequency estimation of different LSFs at different pixels, as each chosen Gabor filters only estimates its pertaining LSF. For the explained reasons, using the algorithm of [158] may result in computing a single estimated LSF which 'jumps' to different LSFs at different pixels, which is certainly undesirable because not corresponding to a physical frequency.

Finally, the algorithm presented here lends itself well to being massively parallelized. In fact, each area and filter can be processed independently, so as to speed up the process significantly.

3.4 Examples of LSF Estimations

Here we show examples of estimations of LSFs of both semi-synthetic and real textures with the use of our proposed algorithm.

The first case that we consider is the semi-synthetic image of the real texture D6, extracted from the Brodatz database ([18]), rendered on a cosine shape.

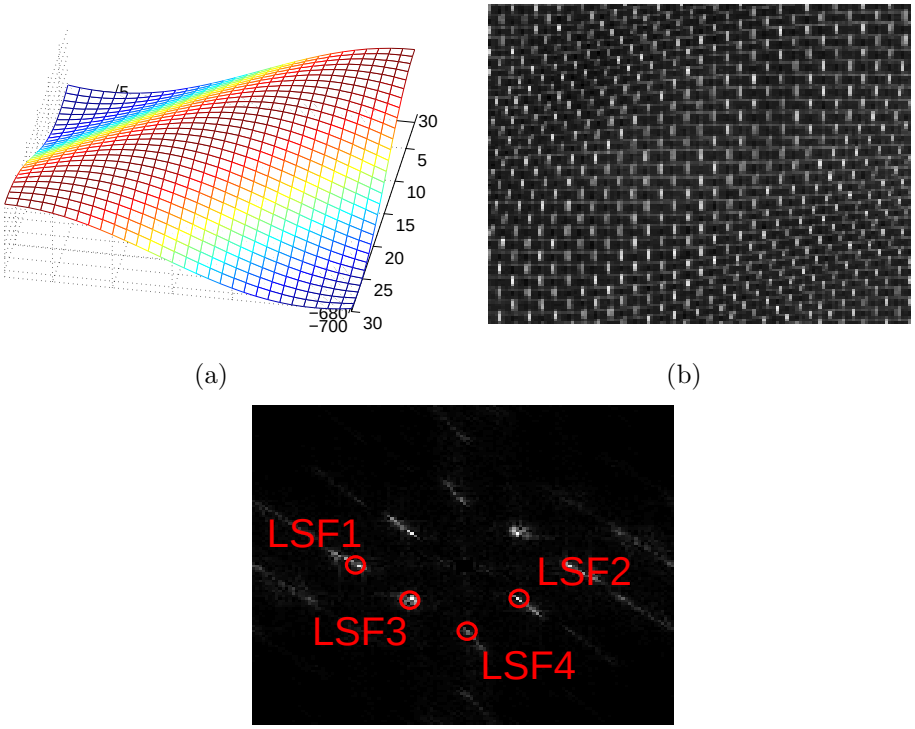
Figure 3.10(a) illustrates the 3D cosine surface, figure 3.10(b) illustrates the produced image, reproduced from figure 3.9(b), and figure 3.10(c) superposes labels on the spectrum, which we use to refer to the respective LSFs.

Applying our method to the image with the Gabor filters illustrated in figure 3.9(e) produces the LSFs, the moduli and phases of which are illustrated in figure 3.11.

As already described in section 3.2.3, in this case the ground truth value of the LSFs at each image pixel is known. The estimated LSFs can therefore be compared with the moduli and phases of the true LSFs, represented in figure 3.12.

From a comparison of the estimated and true LSFs (in figures 3.11 and 3.12 the coordinate axes of corresponding moduli or phases for the same LSF have been scaled to the same range of values) the accuracy of our frequency estimation can be observed.

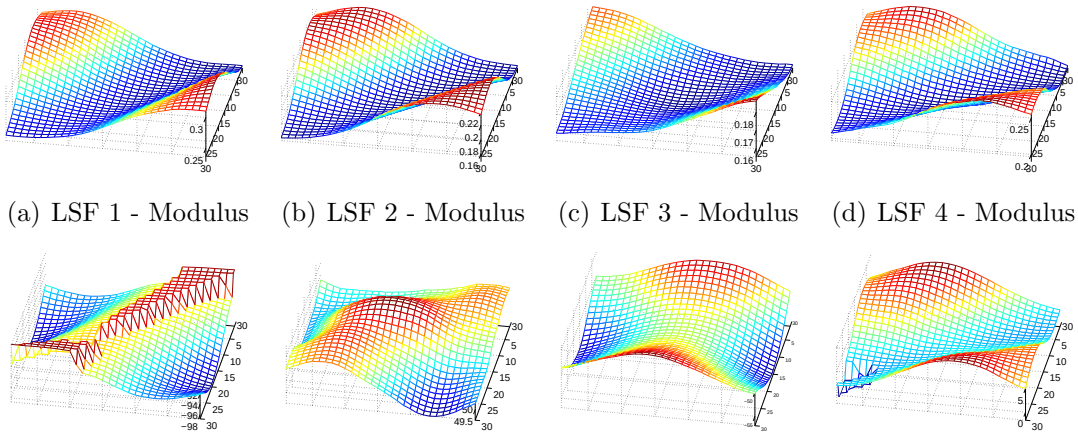
Our method therefore takes advantage of the property that textures generally contain multi-scale information. In particular, spectra with multi-scale distortion are analysed at different scales.



(a) (b)

(c)

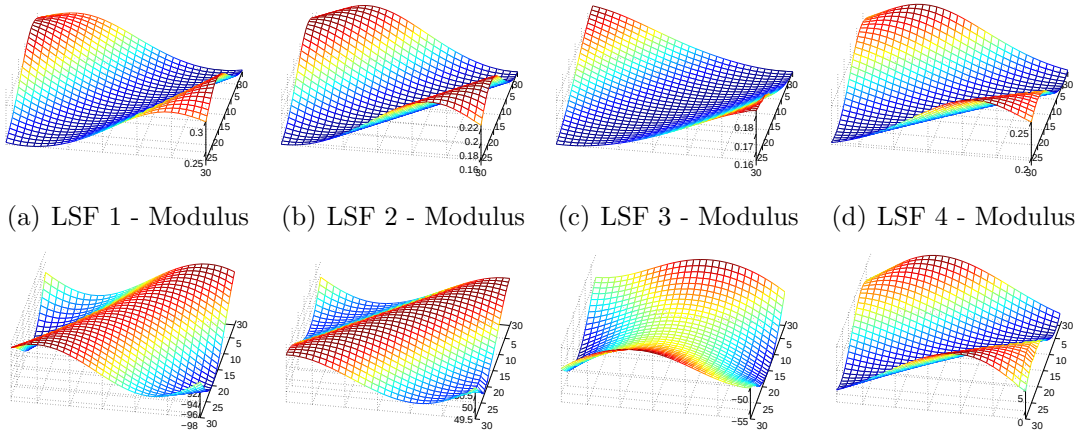
Figure 3.10: (a) 3D cosine surface; (b) 128x128 semi-synthetic picture (as in fig. 3.9(b)) obtained by rendering Brodatz texture D6 on the cosine surface of (a); (c) spectrum amplitude of (b) with labels indicating the selected LSFs



(a) LSF 1 - Modulus (b) LSF 2 - Modulus (c) LSF 3 - Modulus (d) LSF 4 - Modulus

(e) LSF 1 - Phase (f) LSF 2 - Phase (g) LSF 3 - Phase (h) LSF 4 - Phase

Figure 3.11: Moduli and phases of the estimated LSFs of the image in figure 3.10(b)



(a) LSF 1 - Modulus (b) LSF 2 - Modulus (c) LSF 3 - Modulus (d) LSF 4 - Modulus

(e) LSF 1 - Phase (f) LSF 2 - Phase (g) LSF 3 - Phase (h) LSF 4 - Phase

Figure 3.12: Moduli and phases of the ground truth LSFs of the image in figure 3.10(b)

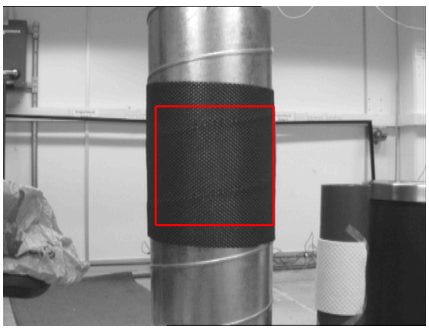
As we see in chapters 4, 5 and 6, the estimated LSFs may be redundant, depending on the particular shape to recover. However having multiple LSF estimates available opens the way to different possible means to increase the robustness of the algorithm: (i) more LSFs can be considered together or (ii) we may reconstruct a 3D surface for each LSF, and then choose the best available reconstruction according to some criteria. Alternatively, the two approaches can be combined: we can improve robustness by considering pairs of LSFs and then choose the best obtained result (using some criteria).

Furthermore, it is clear from the example of figure 3.10(c) that the algorithm can automatically determine the scale, without any need for further analysis (e.g. [96, 136, 156]). In fact the interest scale for the particular LSF is automatically determined by its position in the spectrum, which determines the scale at which the LSF is analysed, i.e. the position of the Gabor filters.

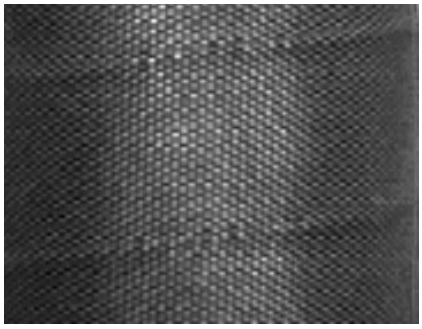
As already mentioned in section 3.2.3 and maintained by [158], very low frequencies, below 6 cycles per image, are not very informative for SFT and are removed from the spectra. In fact, we would assume that the LSF used to recover the shape has to be at least one order of magnitude bigger than the highest frequency associated with the shape, for having a good reconstruction.

It is also worth considering what happens when interest areas superpose. This may be the case when a texture composed of close frequencies is slanted. Most likely this would result in detecting one larger interest area and not separating the two LSFs. This case can be compared to estimating a texture with two main LSFs by only using the lattice of Gabor filters of Super and Bovik. As discussed in section 3.3, this would produce a single estimated LSF which might 'jump' to either of the two relevant LSFs at different pixels. In fact, the superposition can be detected from the presence of gaps in the frequency estimate. By considering a smaller window on the image (e.g. cropping the 96x96 central area from an original 128x128 image), the range of variation of frequencies analysed by the Fourier transform is smaller and hence there is less chance of observing the superposition.

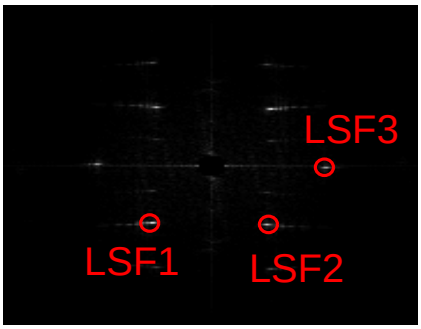
Next we show two examples of LSF estimation from real pictures. Both pictures have been acquired in the lab by using a Pulnix TM-6EX camera at a resolution of 640x480 with 256 grey levels.



(a)

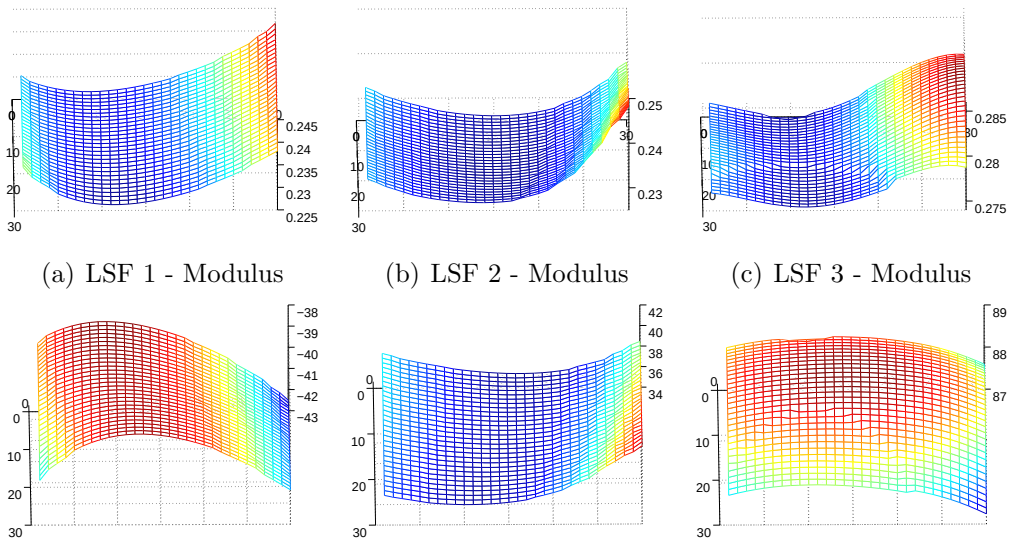


(b)



(c)

Figure 3.13: (a) 640x480 real picture of a rubber rug laid on a cylinder; (b) 180x180 real image extracted from (a); (c) spectrum of (b) with labels indicating the selected LSFs



(a) LSF 1 - Modulus (b) LSF 2 - Modulus (c) LSF 3 - Modulus

(d) LSF 1 - Phase (e) LSF 2 - Phase (f) LSF 3 - Phase

Figure 3.14: Moduli and phases of the estimated LSFs of the image in figure 3.13(a)

Figure 3.13(a) illustrates the image of a rubber rug laid on a metallic cylinder. The sub-image that we consider is presented in figure 3.13(b), and is the 180x180 central part of the picture.

As for the Brodatz texture, figure 3.13(c) labels the main areas of interest in the spectrum of the image and figure 3.14 shows the moduli and phases of the corresponding estimated LSFs.

The second real picture that we consider is illustrated in figure 3.15(a), and represents a textured cloth laid on an inflated balloon. The sub-image used is given by the central 128x128 part of the picture and is shown in figure 3.15(b).

Again, figure 3.15(c) labels the main areas of interest in the spectrum of the image and figure 3.16 shows the moduli and phases of the corresponding estimated LSFs.

Both these real cases are again considered in chapters 5 and 6, where we present the 3D reconstruction of the cylinder and the underlying balloon respectively.

Here we observe that both real images are reasonably noisy. In fact, in both

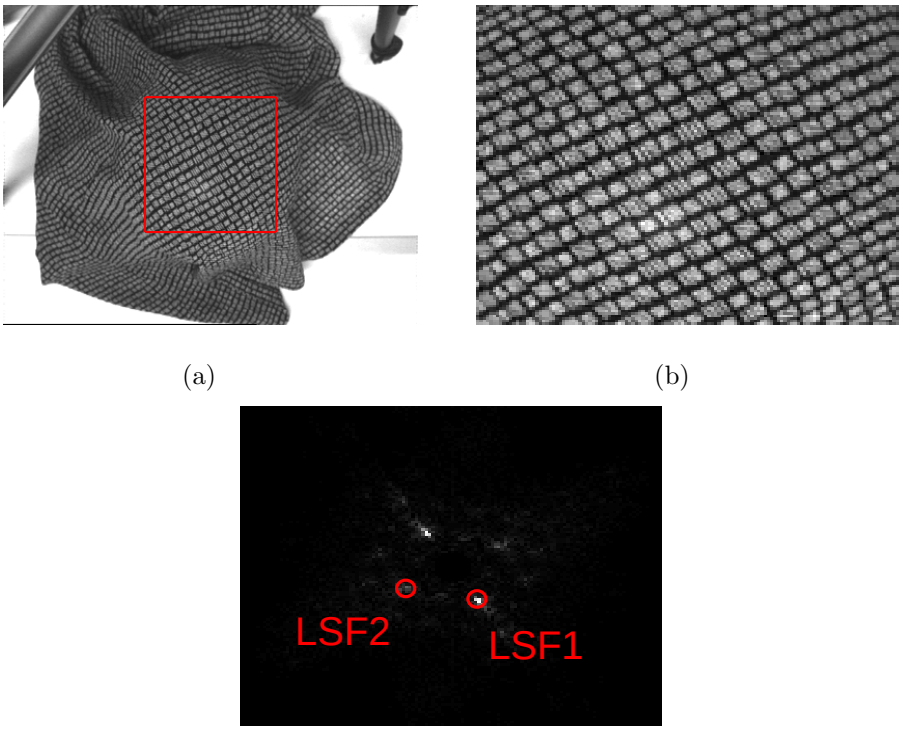
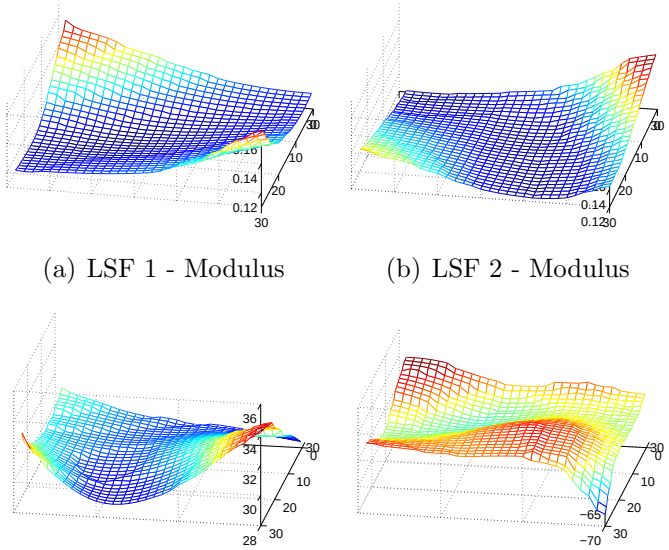


Figure 3.15: (a) 640x480 real picture of a textured cloth laid on an inflated balloon; (b) 128x128 real image extracted from (a); (c) spectrum of (b) with labels indicating the selected LSFs



(a) LSF 1 - Modulus (b) LSF 2 - Modulus

(c) LSF 1 - Phase (d) LSF 2 - Phase

Figure 3.16: Moduli and phases of the estimated LSFs of the image in figure 3.15(a)

cases the quality of the picture is not optimal and variations in illumination and contrast can be observed, especially for the rubber rug picture, where even the reflection of the underlying metallic cylinder and its structure (the two nearly horizontal bands) are present. Furthermore, both the texturing objects have imperfections; the nodes and holes of the rubber rug are not completely regular and the texture is actually 3D, and the textured cloth on the balloon is painted with non equal segments. Notwithstanding the relevant noise, the reconstructed LSFs do look plausible and do not present irregularities

3.5 Final Remarks

The chapter has introduced a novel algorithm for estimating the multi-scale distortion of texture images.

The original idea of Super and Bovik to estimate an instantaneous frequency by the use of Gabor filters ([158]) has been combined with the property of the Fourier transform to express the general frequency content of an image.

The algorithm detects all the main LSFs and sets up a minimum number of Gabor filters. Therefore optimised computational expense is combined with multi-scale analysis of textures.

The chapter has illustrated examples of multiple frequency estimation for both semi-synthetic and real images, showing the robustness of the algorithm against imperfections and variations of illuminations and contrast.

The algorithm is in principle easy to implement and intuitive, and is used in subsequent chapters to recover the texture shape.

Chapter 4

The Orientation of a Planar Surface

This chapter presents the case of Shape-From-Texture for planar textured surfaces.

Planar cases were often considered in early SFT works ([1, 13, 59, 74, 155, 156, 158]), because of their simplicity. However studies which use planar surfaces can also be found in more recent works on SFT ([34, 60, 98, 115, 137, 139, 140]). In fact, among the interesting characteristics of the planar case is the perspective convergence to vanishing points, which usually becomes negligible in non-planar cases (see section 2.4).

For computing the pose of a planar surface, one LSF is generally sufficient.

Here we use the algorithm of [158] to recover the plane orientation: an extensive search is done over the space of possible orientations (slant (σ) - (τ) tilt). The original algorithm using a single LSF is extended to comply with the availability of multiple frequencies. In fact each LSF provides an estimate for the orientation and an index of accuracy is introduced to choose the best estimate.

Here, the algorithm is applied to a variety of semi-synthetic and real images. As the surface model is simple and depends on just two variables (σ-τ), the results are very accurate.

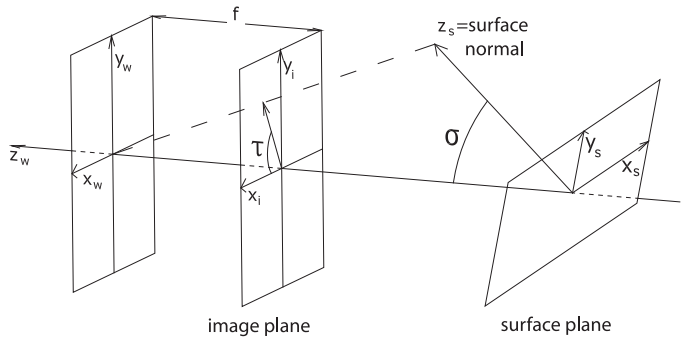


Figure 4.1: Viewing geometry and projection model

4.1 Projection Model

3D shape recovery in SFT is generally based on the preliminary definition of a viewing geometry and a projection model.

The former, as seen in section 2.3, defines the relevant system coordinates, i.e. world coordinates $\mathbf{x}_w = (x_w, y_w, z_w)$, image coordinates $\mathbf{x}_i = (x_i, y_i)$ and surface texture coordinates $\mathbf{x}_s = (x_s, y_s, z_s)$, as well as the orientation parameters σ-τ. Figure 2.2 is here re-presented to remind us of the adopted viewing geometry. As mentioned in section 2.3, the setup, originally introduced by [158], is currently used commonly in SFT.

The projection model relates the system coordinates and frequencies between the surface texture and the image plane. It usually depends on the assumptions about the surface texture shape, e.g. whether we are considering a planar or curved case, and on the methods one might introduce to simplify the solution (this is explained in more detail in subsequent chapters).

Here we consider a planar surface texture, viewed under the assumption of a pin-hole camera model assumption, and use the projection model of [158].

From the definition of the viewing geometry, we can express the surface texture coordinate axes in world coordinates:

$$\begin{aligned}
\hat{\mathbf{x}}_s &= (\cos\sigma\cos\tau, \cos\sigma\sin\tau, -\sin\sigma) \\
\hat{\mathbf{y}}_s &= (-\sin\tau, \cos\tau, 0) \\
\hat{\mathbf{z}}_s &= (\sin\sigma\cos\tau, \sin\sigma\sin\tau, \cos\sigma)
\end{aligned} \tag{4.1}$$

The equations in 4.1 determine the change-of-coordinate transformation between the surface texture and world coordinate systems:

$$\mathbf{x}_w = \begin{bmatrix} \cos\sigma\cos\tau & -\sin\tau & \sin\sigma\cos\tau \\ \cos\sigma\sin\tau & \cos\tau & \sin\sigma\sin\tau \\ -\sin\sigma & 0 & \cos\sigma \end{bmatrix} \mathbf{x}_s + \begin{bmatrix} 0 \\ 0 \\ z_0 \end{bmatrix} , \tag{4.2}$$

where z_0 is the z_w-coordinate of the textured plane where it crosses the optical axis.

We note that a surface texture point has $z_s = 0$ and assume from here on that the surface texture coordinate vector $\mathbf{x}_s$ is a 2D vector (i.e. $\mathbf{x}_s = (x_s, y_s)$) instead of a 3D vector.

The perspective projection from world coordinates to image plane coordinates is expressed by

$$\mathbf{x}_i = \frac{f}{z_w} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \mathbf{x}_w . \tag{4.3}$$

We then combine equations 4.2 and 4.3 to obtain the transformation between the surface texture and image plane coordinates:

$$\mathbf{x}_i = \frac{f}{z_w} \begin{bmatrix} \cos\sigma\cos\tau & -\sin\tau \\ \cos\sigma\sin\tau & \cos\tau \end{bmatrix} \mathbf{x}_s , \tag{4.4}$$

where

$$z_w = z_0 - x_s \sin\sigma , \tag{4.5}$$

and its inverse

$$\mathbf{x}_s = \tilde{z}_w \begin{bmatrix} \cos\tau & \sin\tau \\ -\cos\sigma\sin\tau & \cos\sigma\cos\tau \end{bmatrix} \mathbf{x}_i , \tag{4.6}$$

with

$$\tilde{z}_w = \frac{z_w}{f\cos\sigma} = \frac{z_0}{\sin\sigma(x_i\cos\tau + y_i\sin\tau) + f\cos\sigma} . \tag{4.7}$$

By using the transpose of the Jacobian (J^t) of equations 4.4 and 4.6, we obtain the relation between the surface texture frequencies $\mathbf{u}_s = (u_s, v_s)$ and the image plane frequencies $\mathbf{u}_i = (u_i, v_i)$ under a first order approximation.

In fact we have:

$$\mathbf{u}_i = J^t(\mathbf{x}_s, \mathbf{x}_i) \cdot \mathbf{u}_s , \tag{4.8}$$

where $J^t(\mathbf{x}_s, \mathbf{x}_i)$ is the transpose of the Jacobian of relation 4.6:

$$J^t(\mathbf{x}_s, \mathbf{x}_i) = \begin{bmatrix} \frac{\partial x_s}{\partial x_i} & \frac{\partial y_s}{\partial x_i} \\ \frac{\partial x_s}{\partial y_i} & \frac{\partial y_s}{\partial y_i} \end{bmatrix} \tag{4.9}$$

$$= -\tilde{z}_w^2 \frac{\sin\sigma}{z_0} \begin{bmatrix} x_i \cos^2\tau + y_i \cos\tau \sin\tau & -x_i \cos\sigma \cos\tau \sin\tau + y_i \cos\sigma \cos^2\tau \\ x_i \cos\tau \sin\tau + y_i \sin^2\tau & -x_i \cos\tau \sin^2\tau + y_i \cos\sigma \cos\tau \sin\tau \end{bmatrix}$$
$$+\tilde{z}_w \begin{bmatrix} \cos\tau & -\cos\sigma \sin\tau \\ \sin\tau & \cos\sigma \cos\tau \end{bmatrix}. \tag{4.10}$$

And we have:

$$\mathbf{u}_s = J^t(\mathbf{x}_i, \mathbf{x}_s)\mathbf{u}_i \ , \tag{4.11}$$

where $J^t(\mathbf{x}_i, \mathbf{x}_s)$ is the transpose of the Jacobian of relation 4.4:

$$J^t(\mathbf{x}_i, \mathbf{x}_s) = \begin{bmatrix} \frac{\partial x_i}{\partial x_s} & \frac{\partial y_i}{\partial x_s} \\ \frac{\partial x_i}{\partial y_s} & \frac{\partial y_i}{\partial y_s} \end{bmatrix} \tag{4.12}$$

$$= \frac{f}{z_w^2} \sin\sigma \begin{bmatrix} x_s \cos\sigma \cos\tau - y_s \sin\tau & x_s \cos\sigma \sin\tau + y_s \cos\tau \\ 0 & 0 \end{bmatrix}$$
$$+\frac{f}{z_w} \begin{bmatrix} \cos\sigma \cos\tau & \cos\sigma \sin\tau \\ -\sin\tau & \cos\tau \end{bmatrix}. \tag{4.13}$$

Most importantly, $J^t(\mathbf{x}_i, \mathbf{x}_s)$ can be expressed as a function of the image plane coordinates $\mathbf{x}_i$:

$$J^t(\mathbf{x}_i, \mathbf{x}_s) = \frac{\sin\sigma}{z_w} \begin{bmatrix} x_i & y_i \\ 0 & 0 \end{bmatrix} + \frac{f}{z_w} \begin{bmatrix} \cos\sigma \cos\tau & \cos\sigma \sin\tau \\ -\sin\tau & \cos\tau \end{bmatrix}. \tag{4.14}$$

Equation 4.14 is used in next section to back-project a LSF computed on the image plane to the surface plane.

4.2 Computing the Orientation of a Plane

We explain here how the proposed algorithm processes the image texture multi-scale distortion to produce the orientation of the surface texture plane. As already mentioned in the introduction to the current chapter, the algorithm relies on the corresponding orientation estimation algorithm from [158], which we extend here to deal with multiple LSFs.

As discussed in section 2.2, we make the assumption of 'stochastic' homogeneity of Malik and Rosenholtz ([118]), implying that the relevant LSFs are almost constant over the textured surface under analysis. The instantaneous frequency values which we estimate on the image may in fact be back-projected onto the surface texture. According to the 'stochastic' homogeneity assumption, the back-projected frequencies $\mathbf{u}_s = (u_s, v_s)$ are almost the same at each surface texture point $\mathbf{x}_s$. The assumption includes as a corollary that the variances of both u_s and v_s (therefore the sum of their variances) over the surface texture points $\mathbf{x}_s$ are approximately zero.

The planar-orientation-estimation algorithm uses the corollary to recover the orientation parameters (slant σ and tilt τ) and to select the best estimate from the multiple LSFs. Algorithm 2 summarizes the proposed technique.

Algorithm 2 Structure of the proposed algorithm to estimate the orientation of a planar surface from multiple estimated LSFs

for each estimated LSF **do**
  for each pair of the orientation parameters (σ,τ) **do**
    for each pixel of the image **do**
      • back-project the estimated LSF at the pixel (i.e. $\mathbf{u}_i$) to the planar surface texture (i.e. $\mathbf{u}_s$) using equations 4.11 and 4.14 with the selected parameters (σ,τ)
    end for
    - compute the variance (sum of the variances) $V_{\sigma,\tau}$ of the back-projected texture frequency $\mathbf{u}_s$ (u_s and v_s) associated with the pair (σ,τ) over the surface texture points ($\mathbf{x}_s$)
  end for
  • compute the orientation of the plane associated with the selected LSF by choosing the pair ($\tilde{\sigma}$,$\tilde{\tau}$) which gives the minimum variance $V_{\tilde{\sigma},\tilde{\tau}}$
end for
- choose the best planar orientation estimate by selecting the LSF which determines the smallest of the minima variances $V_{\tilde{\sigma},\tilde{\tau}}$'s

The illustrated algorithm is based on the assumption, already maintained by Super and Bovik in their implementation, that smaller variances of the back-

Image	**True** τ	τ_{GL}	$\lvert\tau_{GL} - \tau\rvert$	τ_{HLC}	$\lvert\tau_{HLC} - \tau\rvert$
D20	160	160.45	0.45	160.05	0.05
D52	-60	-60.28	0.28	-61.08	1.08
D57	90	91.00	1.00	91.41	1.41
D65	60	59.48	0.52	55.00	5.00
D82	90	89.63	0.37	90.71	0.71
D84	135	134.16	0.84	128.58	6.42
D95	-155	-154.92	0.08	-158.57	3.57

Table 4.1: Tilt results of our method (τ_{GL}) on images 4.2(a)-(g), compared to the results of [74] (τ_{HLC}) (angles in degrees)

projected LSFs onto the surface texture correspond to better estimates, as the theoretical value of variance is approximately zero. The idea is here also used to select which LSF determines the best orientation estimate.

As observed in section 3.3 for algorithm 1, we see that algorithm 2 also lends itself to a parallel implementation. Furthermore, the 2D search for the minimum variance $V_{\sigma,\tau}$ in the plane (σ,τ) can be performed in a coarse-to-fine manner to yield greater computational efficiency and arbitrary precision.

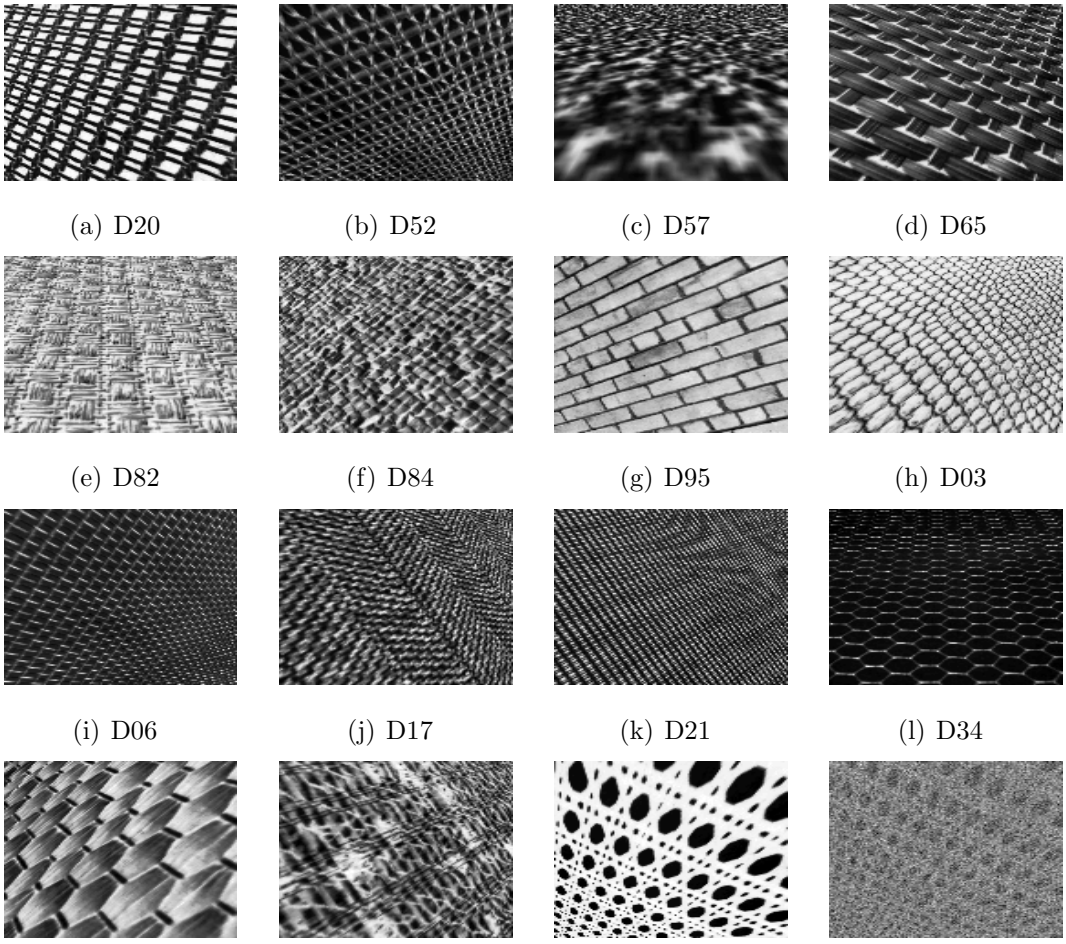
4.3 Results

We demonstrate the illustrated method on two sets of images.

The first is shown in figure 4.2. The images in this set were synthesized by mapping real textures from the Brodatz database ([18]) onto an inclined surface and then rendering it as a new image. The second set is shown in figure 4.3. The pictures in it are real images, and are presented in the second part of this section.

All processed images were 128x128 pixels with 256 grey levels. The back-projection of the computed LSFs for each value (σ, τ) was done just for the middle section of the image, in this case 64x64, so as to avoid edge effects.

Although we could not compare our results with those of [158], because we could not gather all necessary data from their original setup, we compared our results to those of Hwang et al. ([74]), who improved on the results of [158].



(a) D20 (b) D52 (c) D57 (d) D65

(e) D82 (f) D84 (g) D95 (h) D03

(i) D06 (j) D17 (k) D21 (l) D34

(m) D56 (n) D81 (o) D101 (p) D101 - white Gaussian noise with SNR=-5dB added

Figure 4.2: Images synthesized using the Brodatz textures ([18])

Image	**True** σ	σ_{GL}	$\lvert\sigma_{GL} - \sigma\rvert$	σ_{HLC}	$\lvert\sigma_{HLC} - \sigma\rvert$
D20	37	37.36	0.36	37.53	0.53
D52	50	49.55	0.45	49.58	0.42
D57	70	70.37	0.37	67.30	2.70
D65	50	48.71	1.29	54.38	4.38
D82	50	49.16	0.84	48.62	1.38
D84	35	35.32	0.32	33.26	1.74
D95	27	25.89	1.11	28.45	1.45

Table 4.2: Slant results of our method (σ_{GL}) on images 4.2(a)-(g), compared to the results of [74] (σ_{HLC}). We choose for comparison the values they achieve with what they term the 'voting method' (angles in degrees)

Image	**True** τ	τ_{GL}	$\lvert\tau_{GL} - \tau\rvert$	**True** σ	σ_{GL}	$\lvert\sigma_{GL} - \sigma\rvert$
D3	45	43.30	1.70	50	49.74	0.26
D6	-30	-30.60	0.60	50	49.48	0.52
D17	30	28.86	1.14	50	44.87	5.13
D21	30	30.80	0.80	40	39.87	0.13
D34	90	90.03	0.03	50	49.28	0.72
D56	135	133.06	1.94	50	50.16	0.16
D81	30	28.81	1.19	45	46.68	1.68
D101	-120	-121.49	1.49	55	54.42	0.58

Table 4.3: Tilt and slant results of our method (τ_{GL},σ_{GL}) on images 4.2(h)-(o) (angles in degrees)

	SNR (dB)				
Image (σ/τ)	∞	20	10	0	-5
D20 (160/37)	160.4/37.4	159.5/37.1	159.6/36.8	159.6/37.3	157.3/37.3
D34 (90/50)	-90.0/49.3	88.1/46	88.7/45.7	88.7/44.1	X
D52 (-60/50)	-60.3/49.5	-58.7/47.6	-64.4/49	-67.2/46.6	-61.6/34.3
D82 (90/50)	89.6/49.2	-89.1/51.3	90.6/52.3	86.1/45.4	X
D95 (-155/27)	-154.9/25.9	-158.8/25.2	-159.8/26.2	-160.8/25.4	-160.6/24.5
D101 (-120/55)	-121.5/54.4	-118.4/55.1	-118.7/55.1	-119.2/54.7	-120.7/55.1

Table 4.4: Surface orientations (τ/σ) estimated using our method on noisy versions of images 4.2(a),(l),(b),(e),(g),(o) - true values are in parenthesis (X indicates that the results were not reliable) (angles in degrees)

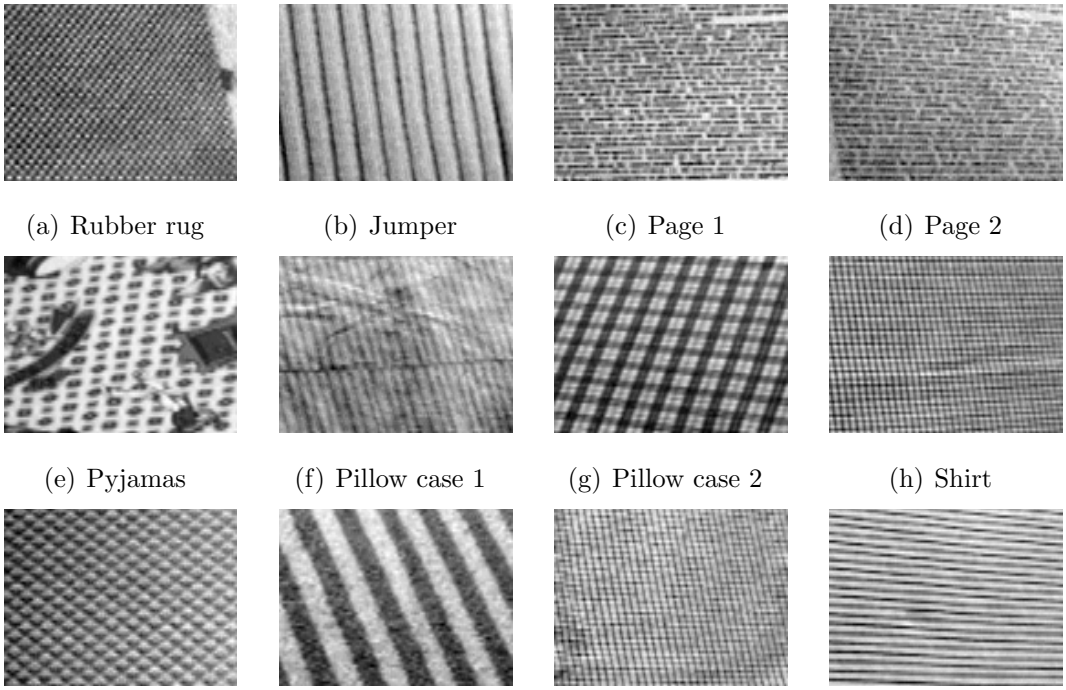
Please note that the comparison with [74] only applies to the numerical results. The cited work uses a different technique, as outlined in chapter 2, and a full comparison is not relevant to our work.

In particular, images 4.2(a)-(g) are taken from the data set of [74]. Tables 4.1 and 4.2 compare the results achieved on this subset. Our average estimation errors for τ and σ are 0.51° and 0.68° respectively, while Hwang et al. achieve corresponding values of 2.6° and 1.8°. The accuracy of our method is higher.

In addition to this, table 4.3 reports the results obtained on the other 8 textured surfaces (images 4.2(h)-(o)), with average errors for tilt and slant of respectively 1.11° and 1.15°. Again our algorithm shows a slightly better average performance than [74] on this other subset of semi-synthetic images.

To show the robustness of our algorithm we have added white Gaussian noise with SNR ranging from 20 to -5 dB to textures D20, D34, D52, D82, D95 and D101 (e.g. in figure 4.2(p) texture D101 with SNR=-5dB noise added). We used SNR(dB)=$10\log_{10}\left(\frac{\sigma_i^2}{\sigma_n^2}\right)$, where σ_i^2 and σ_n^2 are the variances of the image and of the added noise respectively. Table 4.4 shows results of computed orientations, which do not differ much from the noiseless values, even in the case of higher noise.

On another note, an interesting result was obtained by considering the 96x96 central part of the image synthesized from texture D101. The new reconstruction



(a) Rubber rug (b) Jumper (c) Page 1 (d) Page 2

(e) Pyjamas (f) Pillow case 1 (g) Pillow case 2 (h) Shirt

(i) Sponge (j) Towel (k) Trousers (l) T-shirt

Figure 4.3: Real images of textured planes

is more accurate, as the error for τ and σ passes from 1.49° and 0.58°, to 0.6° and 0.35° respectively. We justify this by considering the nature of our method. A single area of the Fourier transform expresses how much a single frequency is spread in the image. A smaller image corresponds therefore to a smaller area, because the single frequency varies less. This implies less risk of superposition of areas and better separation of one from the other, resulting in better orientation estimation. As already mentioned in section 3.4, this characteristic is a key feature in the application of our algorithm to deal with complex textures.

Figure 4.3 presents the set of real images we tested our algorithm on. All of them are given by the central 128x128 part of corresponding 640x480 pictures that we acquired in the lab.

Figure 4.4 presents one of the whole images, the towel, from which figure 4.3(j) was derived. As can be seen, the textured object was laid flat on a panel of known orientation and photographed with a Pulnix TM-6EX camera.

Image	**True** τ	τ_{GL}	$\lvert\tau_{GL} - \tau\rvert$	**True** σ	σ_{GL}	$\lvert\sigma_{GL} - \sigma\rvert$
rubber	-61.2	-61.7	0.5	-35.3	-33.4	1.9
jumper	-76.8	-70	6.8	-33.6	-32.1	1.5
page1	27.2	26.7	0.5	-23.6	-23.4	0.2
page2	-56.4	-58.7	2.3	-36.9	-34.0	2.9
pyjamas	27.2	28.8	1.6	-23.6	-20.2	3.4
p'case1	56.4	56.8	0.4	-36.9	-34.4	2.5
p'case2	33.5	32.3	1.2	-32.7	-33.6	0.9
shirt	-76.8	-72.5	4.3	-33.6	-31.2	2.4
sponge	21.7	22.1	0.4	-25.5	-25.1	0.4
towel	-33.6	-33.8	0.2	-38.8	-39.9	1.1
trousers	-61.2	-61.3	0.1	-35.3	-35.2	0.1
T-shirt	-56.4	-58.7	2.3	-36.9	-35.7	1.2

Table 4.5: Tilt and slant results of our method (τ_{GL},σ_{GL}) on real images in figure 4.3 (angles in degrees)

Figure 4.4: Full resolution (640x480) image of the towel in figure 4.3(j)

The chosen textured objects were mainly fabrics belonging to clothes or pillow cases or towels with some homogeneous pattern on them, but also included some different materials, e.g. a sponge, a rubber rug and a printed page.

It is clear that the pictures are affected by variations in illumination (4.3(i)) and self shadowing (4.3(b)). Also some textures are imperfect (4.3(c) and 4.3(a)), some clothes have evident creases (4.3(f)) or occlusions in the pattern (4.3(e)).

Table 4.5 shows the results we obtained with our algorithm, compared to the ground truth (which was obtained using a multiple camera system to give the orientation of the plane prior to the experiment). On average, tilt and slant were estimated with errors of 1.7° and 1.5° respectively.

These data, together with the results on semi-synthetic textures, confirm the robustness of our algorithm and its improved accuracy (on average, with respect to the method in [74]) on a wide range of textures.

A final note is made about the computational cost of our proposed algorithm, compared with [158], as shown in sections 3.2.1, 3.2.2 and 3.2.3. Processing all the 27 semi-synthetic and real images, 69 interest areas of the Fourier transform were detected, therefore making an average of 2.55 areas per image. Altogether these areas determined 394 Gabor functions, giving an average of 5.71 per area. Thus the number of convolutions per image was on average 87.55. This means a considerable computational saving with respect to the algorithm of [158], which uses 378 convolutions per image, i.e. we saved 76.84% in computing power.

4.4 Final Remarks

The chapter has presented a logical extension of the simple algorithm for planar orientation estimation of [158].

To exploit the availability of multiple LSFs we have proposed an index of accuracy based on variance. In fact every LSF is used to estimate the orientation of the textured plane, and the best result is selected according to the minimum variance.

The algorithm could not be compared with the original results of [158], for lack of data from their setup. However it performed better than the algorithm of [74], which improved on the results of [158].

Robustness is also proved on a variety of semi-synthetic and real textures. Our algorithm was not affected by simulated Gaussian noise, variations in illumination, self shadowing, imperfections in the texture pattern, creases and partial occlusions.

Most importantly, although the planar case is particularly simple, the improved accuracy and robustness gives further evidence of the quality of the multi-scale instantaneous frequency estimation presented in chapter 3. The algorithm is here applied to a range of deterministic and stochastic homogeneous textures, with varying presence of perspective effects.

Subsequent chapters will present more challenging 3D reconstructions.

Chapter 5

Frontal Views in Shape-From-Texture

This chapter illustrates the recovery of Shape-From-Texture in the case of an image of a textured scene showing a *frontal point*.

A frontal point is defined as a point on the surface texture at which the tangent plane is parallel to the image plane. It is an important piece of information to the 3D reconstruction algorithm, as it provides the fronto-parallel appearance of the texture.

Frontal appearance has been utilised in many studies (e.g. [42, 48, 110, 112, 113, 114, 159]) to locally recover the surface orientation via a comparison with the local patches.

More naively, the use of a frontal appearance can also be related to the use of isotropy (e.g. [9, 53, 174]): the local orientation is recovered by measuring the deviation from isotropy, where the frontal appearance is assumed to be isotropic.

In this chapter, two algorithms are presented which recover the 3D shape from a texture, given its frontal appearance or given that a frontal point or a point of known orientation is displayed in the observed image.

The first algorithm reconstructs the 3D surface coordinates of a developable shape for each estimated LSF and selects the best available reconstruction. Here by developable shape we mean a surface which can be opened flat to an unfolded planar surface. On a developable shape a planar texture, e.g. a picture or a cloth,

can be laid without stretches, while we get stretches when we lay the texture on a non-developable (or equivalently non-unfoldable) surface, e.g. a sphere segment.

A second algorithm is proposed to overcome the limitation to developable surfaces and improve robustness. This second algorithm recovers a shape from each available pair of LSFs by using SVD and similarly selects the best available reconstruction.

For the two algorithms, both orthographic and perspective equations are presented. In particular the first algorithm is original, while the second algorithm draws on the orthographic 3D reconstruction method of Dugué and Elghadi ([42]), which we extend to deal with perspective cases.

The algorithms are tested and compared on a semi-synthetic image and on a newly proposed set of real images, for which the ground truth was known. Both algorithms are accurate, with results in line with the state of the art in SFT, and improved robustness is shown for the second.

5.1 Detection and Localisation of a Frontal Point

The presence of a frontal view of the texture in the observed image is an important cue. It provides information about the statistics and appearance of the original texture motif and, at the same time, the localisation of a point in the image plane of known orientation ($\sigma = 0$).

As argued by [113, 114], the position of a frontal point could certainly be provided by a simple additional manual input, e.g. a mouse-click. The small additional information would be a reasonable cost for a 3D reconstruction of the shape.

However we propose here to automatically detect the location of a frontal point, and possibly to be able to say when the texture is not showing a frontal point.

We draw on an algorithm introduced by Super and Bovik in [159], and thereafter used by other authors (e.g. [42, 114]), sometimes with disappointing results (as argued by [114]).

The algorithm easily integrates with the method that we proposed for estimating the main LSFs in chapter 3, as it is based on Gabor filters and on convolving

them and their first order derivatives with the original image. This is convenient as the computation is shared with the previous frequency estimation.

Let us first consider the original algorithm described by Super and Bovik in [159].

We define $F(\mathbf{u}, \mathbf{x}_i)$ the local spectrum of an image I, in the neighbourhood of the image point $\mathbf{x}_i = (x_i, y_i)$ at the frequency $\mathbf{u} = (u, v)$, as seen in section 3.1. We also define $A(\mathbf{u}, \mathbf{x}_i)$ as the amplitude of F; i.e. $A(\mathbf{u}, \mathbf{x}_i) = |F(\mathbf{u}, \mathbf{x}_i)|$. Then, as indicated in [159], the second order moments of the spectrum amplitude A at the image point $\mathbf{x}_i$ are given by:

$$a(\mathbf{x}_i) = \int\int u^2 A^2(\mathbf{u}, \mathbf{x}_i) d\mathbf{u} \tag{5.1}$$

$$b(\mathbf{x}_i) = \int\int uv A^2(\mathbf{u}, \mathbf{x}_i) d\mathbf{u} \tag{5.2}$$

$$c(\mathbf{x}_i) = \int\int v^2 A^2(\mathbf{u}, \mathbf{x}_i) d\mathbf{u} \tag{5.3}$$

Super and Bovik utilise the lattice of Gabor filters that cover the whole spectrum of the image, as represented in figure 3.4. They write the second order moments as discrete summations of the output of convolutions of the first derivatives of the Gabor filters with the image. Then a further summation of the output of the convolutions of the Gabor filters (undifferentiated) with the image is used to normalise the moments for slow variations due to shadowing (we refer the reader to [159] for a detailed description).

Finally they transform the computed second order moments a, b and c into a canonical form, which is independent of the coordinate system:

$$M(\mathbf{x}_i) = \frac{1}{2}\left(a(\mathbf{x}_i) + c(\mathbf{x}_i) + \sqrt{b^2(\mathbf{x}_i) + [a(\mathbf{x}_i) - c(\mathbf{x}_i)]^2}\right) \tag{5.4}$$

$$m(\mathbf{x}_i) = \frac{1}{2}\left(a(\mathbf{x}_i) + c(\mathbf{x}_i) - \sqrt{b^2(\mathbf{x}_i) + [a(\mathbf{x}_i) - c(\mathbf{x}_i)]^2}\right) \tag{5.5}$$

where in fact M and m are the second order moments measured along the axes of maximum and minimum variations and are called canonical moments.

Super and Bovik argue that the frontal point location is given by the coordinates of the minimum of $\sqrt{mM}$ over the image pixels.

As already discussed in chapter 3, using a complete lattice of filters is computationally expensive and is problematic when the texture under analysis does not

have a well defined single dominant LSF. In fact the algorithm for the detection of the frontal point of Super and Bovik, as proposed in [159], is argued by Loh and Kovesi in [114] to be inaccurate, and the authors prefer to consider a manual input.

For the distortion estimation problem in chapter 3, we propose considering the main LSFs of the texture separately. In algorithm 1, chapter 3, different groups of Gabor filters (generally fewer than if using the whole lattice) are used to estimate the selected LSFs at the image pixels.

Then the local spectrum F, and consequently its amplitude A, are taken to be composed of only the well defined selected LSFs:

$$A(\mathbf{u}, \mathbf{x}_i) = \sum_k \delta(\mathbf{u} - \mathbf{u}_k(\mathbf{x}_i)) \ , \tag{5.6}$$

where k indicates the particular LSF among the estimated frequencies, which are obviously functions of the image pixel $\mathbf{x}_i$.

By substituting the delta functions into the continuous second order moment equations 5.1, 5.2 and 5.3, we obtain:

$$a(\mathbf{x}_i) = \sum_k u_k^2(\mathbf{x}_i) \tag{5.7}$$

$$b(\mathbf{x}_i) = \sum_k u_k(\mathbf{x}_i) v_k(\mathbf{x}_i) \tag{5.8}$$

$$c(\mathbf{x}_i) = \sum_k v_k^2(\mathbf{x}_i) \tag{5.9}$$

In particular, for 2 frequencies which we write as $\mathbf{u}' = (u', v')$ and $\mathbf{u}'' = (u'', v'')$, after transforming the moments into their canonical form, the square root of the product of the canonical second order moments is given by

$$\sqrt{mM} = |u_i' v_i'' - u_i'' v_i'| \ . \tag{5.10}$$

It is easy to show that the $\sqrt{mM}$ computed in this way is inversely proportional to the area gradient and directly proportional to the density gradient (see [52] for more details on the two texture gradients), computed as a function of the same two frequencies. This intuitively confirms that the minimum of $\sqrt{mM}$ is

at the frontal point: for a chessboard texture it would be equivalent to assuming that the largest squares, i.e. squares with the biggest area, are frontal.

In fact we use equation 5.10 and 2 LSFs to determine the location of a frontal point, where the $\sqrt{mM}$ is post-smoothed for increased robustness.

Using well defined and estimated LSFs from our multi-scale distortion algorithm ensures a robust frontal point localisation.

We also found that the described method could detect the lack of a frontal point: in this case the minimum value for the term $\sqrt{mM}$ would lie on the border pixels of the image.

Again, drawing the parallel between the area gradient and the chessboard texture, to have the minimum of $\sqrt{mM}$ on the border pixels of the image would mean that the squares with the greatest area are on the border. This could mean that the displayed frontal point is just on the border, or that the frontal point is actually not shown in the image and that the detected value on the border is the most frontal point. In either case we discard the minimum of $\sqrt{mM}$ if it lies close to the borders of the image, and require a manual input as in [114].

5.2 A Frontal View and a Local Spatial Frequency

Here we propose an original algorithm to recover the 3D developable shape of a texture displaying a frontal point or a point of known orientation.

Under the assumptions of either an orthographic or a perspective camera model, the algorithm processes each available LSF separately to produce a 3D surface, and then selects the best reconstruction.

The accuracy of the algorithm is shown by testing it on a semi-synthetic image and on a newly proposed set of real images, with known ground truth.

5.2.1 Projection model extended to developable shapes and orthographic projections

Here we extend the projection model of section 4.1. The model therein presented was based on the viewing geometry illustrated in section 2.3 and on the projection

model of Super and Bovik ([158]). In the present section, we extend the model to deal with non-planar surfaces and to include orthographic relations between surface texture and image plane coordinates and frequencies.

Section 2.3 defines the relevant system coordinates, i.e. world coordinates $\mathbf{x}_w = (x_w, y_w, z_w)$, image coordinates $\mathbf{x}_i = (x_i, y_i)$ and surface texture coordinates $\mathbf{x}_s = (x_s, y_s, z_s)$, and the orientation parameters σ-τ.

The described viewing geometry is used in 4.1 to determine the relation between surface texture and image plane coordinates and frequencies in the perspective case, following the projection model of Super and Bovik ([158]).

A simplification of the projection model of [158] is presented here by considering the same relations under an orthographic camera model assumption. As discussed in section 2.4, orthographic relations only make sense for non-planar shapes, as homogeneity is preserved under affine transformations and provides no information about the plane orientation ([10, 34]).

Let us again consider the change of coordinates between the surface texture and world coordinate systems, already expressed by equation 4.2:

$$\mathbf{x}_w = \begin{bmatrix} \cos\sigma\cos\tau & -\sin\tau & \sin\sigma\cos\tau \\ \cos\sigma\sin\tau & \cos\tau & \sin\sigma\sin\tau \\ -\sin\sigma & 0 & \cos\sigma \end{bmatrix} \mathbf{x}_s + \begin{bmatrix} 0 \\ 0 \\ z_0 \end{bmatrix} , \tag{5.11}$$

Under the orthographic camera model, the image plane coordinates coincide with the world coordinates (along the x_w and y_w axes). In fact the projection from world coordinates to image plane coordinates is expressed by

$$\mathbf{x}_i = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \mathbf{x}_w \ . \tag{5.12}$$

Considering the surface texture coordinates as 2D vectors $\mathbf{x}_s = (x_s, y_s)$ (on the surface texture $z_s = 0$, as shown in section 4.1). Then the orthographic change-of-coordinate transformation between the surface texture and image plane coordinates is obtained by combining equations 5.11 and 5.12:

$$\mathbf{x}_i = \begin{bmatrix} \cos\sigma\cos\tau & -\sin\tau \\ \cos\sigma\sin\tau & \cos\tau \end{bmatrix} \mathbf{x}_s \ . \tag{5.13}$$

And its inverse is

$$\mathbf{x}_s = \frac{1}{\cos\sigma} \begin{bmatrix} \cos\tau & \sin\tau \\ -\cos\sigma\sin\tau & \cos\sigma\cos\tau \end{bmatrix} \mathbf{x}_i \ . \tag{5.14}$$

As in section 4.1, the transpose of the Jacobian (J^t) of equations 5.13 and 5.14 gives the relation between the surface texture frequencies $\mathbf{u}_s = (u_s, v_s)$ and the image plane frequencies $\mathbf{u}_i = (u_i, v_i)$ to a first order approximation.

Therefore, under the orthographic camera model, we have:

$$\mathbf{u}_i = J^t(\mathbf{x}_s, \mathbf{x}_i)\mathbf{u}_s \ , \tag{5.15}$$

where $J^t(\mathbf{x}_s, \mathbf{x}_i)$ is the transpose of the Jacobian of relation 5.14:

$$J^t(\mathbf{x}_s, \mathbf{x}_i) = \begin{bmatrix} \frac{\partial x_s}{\partial x_i} & \frac{\partial y_s}{\partial x_i} \\ \frac{\partial x_s}{\partial y_i} & \frac{\partial y_s}{\partial y_i} \end{bmatrix} = \frac{1}{\cos\sigma} \begin{bmatrix} \cos\tau & -\cos\sigma\sin\tau \\ \sin\tau & \cos\sigma\cos\tau \end{bmatrix} . \tag{5.16}$$

And we have its inverse:

$$\mathbf{u}_s = J^t(\mathbf{x}_i, \mathbf{x}_s)\mathbf{u}_i \ , \tag{5.17}$$

where $J^t(\mathbf{x}_i, \mathbf{x}_s)$ is the transpose of the Jacobian of relation 5.13:

$$J^t(\mathbf{x}_i, \mathbf{x}_s) = \begin{bmatrix} \frac{\partial x_i}{\partial x_s} & \frac{\partial y_i}{\partial x_s} \\ \frac{\partial x_i}{\partial y_s} & \frac{\partial y_i}{\partial y_s} \end{bmatrix} = \begin{bmatrix} \cos\sigma\cos\tau & \cos\sigma\sin\tau \\ -\sin\tau & \cos\tau \end{bmatrix} . \tag{5.18}$$

The above equations for an orthographic projection model 5.15, 5.16, 5.17, 5.18 together with the corresponding equations for a perspective model 4.8, 4.10, 4.11, 4.14 relate the frequencies on the image plane to the surface texture.

Although the relations are defined using planar surfaces, they can easily be extended to general surfaces. Let us assume that the surface is smooth and that at any point it can be locally approximated with the corresponding tangent plane. Then the given equations apply to the tangent plane at the considered point. In this case, the $\mathbf{x}_s$ coordinate is placed, oriented as described in section 2.3, at the considered point.

The relevant notation is unchanged, with the sole exception of variable z_0 in equations 4.2, 4.5, 4.7, 5.11. Formerly the z_w-coordinate of the intersection of the planar surface texture and the optical axis, z_0 now equivalently represents the intersection of the optical axis and the tangent plane at the considered point. This implies that once a tangent plane, i.e. a point, is chosen, then the value of z_0 is a constant.

In particular we are interested in back-projecting the frequency value $\mathbf{u}_i$ for a selected LSF at an image point $\mathbf{x}_i$ back to the texture surface. This is achieved by using equations 5.17, 5.18, 4.11, 4.14, which we re-write below:

$$\mathbf{u}_s = J^t_{\mathbf{x}_i\mathbf{x}_s}(\mathbf{x}_i)\mathbf{u}_i \ , \tag{5.19}$$

where in the orthographic case

$$J^t_{\mathbf{x}_i\mathbf{x}_s}(\mathbf{x}_i) = \begin{bmatrix} \cos\sigma\cos\tau & \cos\sigma\sin\tau \\ -\sin\tau & \cos\tau \end{bmatrix} \ , \tag{5.20}$$

and in the perspective case

$$J^t_{\mathbf{x}_i\mathbf{x}_s}(\mathbf{x}_i) = \frac{\sin\sigma}{z_w}\begin{bmatrix} x_i & y_i \\ 0 & 0 \end{bmatrix} + \frac{f}{z_w}\begin{bmatrix} \cos\sigma\cos\tau & \cos\sigma\sin\tau \\ -\sin\tau & \cos\tau \end{bmatrix} \ , \tag{5.21}$$

with z_w the corresponding coordinate in the $\mathbf{x}_w$ coordinate system, of the surface point which projects to the image point $\mathbf{x}_i$.

Please note the slight change in notation adopted here for the Jacobian. The subscript indicates that the Jacobian is obtained by differentiating the $\mathbf{x}_i$ (as expressed in equations 5.13 and 4.4) with respect to $\mathbf{x}_s$. The $\mathbf{x}_i$ in parentheses indicates that the Jacobian depends on the shape parameters at the particular image point $\mathbf{x}_i$ that we consider. In the orthographic case, $\mathbf{x}_i$ means the orientation parameters (σ, τ) at the image point. In the perspective case, $\mathbf{x}_i$ means the orientation, location and depth parameters (σ, τ, x_i, y_i, z_w) at the image point.

The proposed algorithm relies on the general assumption of 'stochastic' homogeneity of Malik and Rosenholtz ([118]) described in section 2.2. That implies that the relevant LSFs are almost constant over the textured surface under analysis, as already seen in section 4.2.

However the $\mathbf{x}_s$ coordinate system changes from point to point on the surface texture according to the tilt-direction, because the x_s axis is aligned with it. Let us assume that $\mathbf{u}_{i1}$ and $\mathbf{u}_{i2}$ are the frequency values, pertaining to a particular LSF, that we measure at the two image points $\mathbf{x}_{i1}$ and $\mathbf{x}_{i2}$. Let us also assume that $\mathbf{u}_{s1}$ and $\mathbf{u}_{s2}$ are the back-projected frequencies, at the surface texture points $\mathbf{x}_{s1}$ and $\mathbf{x}_{s2}$, corresponding to the measured frequencies $\mathbf{u}_{i1}$ and $\mathbf{u}_{i2}$. The 'stochastic'

homogeneity assumption translates into the following:

$$\mathbf{u}_{s1} = \mathbf{u}_{s2} \tag{5.22}$$

$$J^t_{\mathbf{x}_i\mathbf{x}_s}(\mathbf{x}_{i1})\mathbf{u}_{i1} = \begin{bmatrix} \cos\Delta\tau & \sin\Delta\tau \\ -\sin\Delta\tau & \cos\Delta\tau \end{bmatrix} J^t_{\mathbf{x}_i\mathbf{x}_s}(\mathbf{x}_{i2})\mathbf{u}_{i2} \tag{5.23}$$

where the matrix anti-rotates the $\mathbf{x}_s$ coordinate systems by $\Delta\tau$ to align them. The relation applies to both orthographic and perspective cases.

5.2.2 Computing the 3D coordinates of developable surfaces

The proposed algorithm has two steps. Firstly it processes the multi-scale distortion measure to recover a 3D shape from each LSF. Then it selects the best available reconstruction according to variance. The algorithm assumes that the underlying surface is developable and that the image displays a frontal point or a point with known orientation.

In the orthographic case, substituting for $J^t_{\mathbf{x}_i\mathbf{x}_s}$ from equation 5.20 into equation 5.23 gives two equations in 4 unknowns: σ_1, τ_1, σ_2, τ_2. Assume we know the orientation (σ_1, τ_1) of the tangent plane at point P_1, the surface texture point which projects to $\mathbf{x}_{i1}$. Then we can solve equation 5.23 for σ_2 and τ_2 (the orientation parameters at point P_2, respectively corresponding to $\mathbf{x}_{i2}$).

The equation admits an analytical solution for both σ_2 and τ_2. Such a solution is a novel contribution. τ_2 is determined from the following

$$\tau_2 - \tau_1 = \tan^{-1} \frac{u_{i1}\sin\tau_1 - v_{i1}\cos\tau_1 - u_{i2}\sin\tau_1 + v_{i2}\cos\tau_1}{\cos\sigma_1(u_{i1}\cos\tau_1 + v_{i1}\sin\tau_1) + u_{i2}\cos\tau_1 + v_{i2}\sin\tau_1} = \Delta\tau \ , \tag{5.24}$$

which provides two values in the $[0, 2\pi]$ range, as for the well-known tilt-ambiguity (a detailed explanation can be found in [52]):

$$\tau_2 = \begin{cases} \tau_1 + \Delta\tau \\ \tau_1 + \Delta\tau + \pi \end{cases} . \tag{5.25}$$

σ_2 is given by

$$\sigma_2 = \cos^{-1}\left[\frac{\cos\sigma_1(u_{i1}\cos\tau_1 + v_{i1}\sin\tau_1) + u_{i2}\cos\tau_1 + v_{i2}\sin\tau_1}{\cos(\tau_2 - \tau_1)(u_{i2}\cos\tau_2 + v_{i2}\sin\tau_2)} - 1\right] . \tag{5.26}$$

The perspective equations are obtained by substituting for $J^t_{\mathbf{x}_i\mathbf{x}_s}$ from eq. 5.21 in eq. 5.23. In this case, in addition to the σ and τ's, z_{w1} and z_{w2} are unknowns, i.e. the z_w coordinates at the surface points P_1 and P_2.

As in the orthographic case, we assume z_{w1}, σ_1 and τ_1 are known, i.e. the orientation and depth of P_1 are assumed known. Furthermore z_{w2} is inferred from the assumed orientation parameters (σ_1,τ_1), and then refined once the values of σ_2 and τ_2 are computed. In the first instance z_{w2} is assigned a value as if P_2 lies on the tangent plane at P_1, i.e. the depth z_{w2} of P_2 is computed from z_{w1} using the plane passing through P_1 and oriented according to σ_1 and τ_1. This produces two equations in two unknowns and enables us to compute σ_2 and τ_2 analytically, which are then used to refine z_{w2}.

Experimental results show that assuming P_2 to initially be on the tangent plane at P_1 gives accurate results for σ_2 and τ_2 if P_1 and P_2 correspond to nearby points on the image plane. Our algorithm therefore uses a point P of known depth and orientation to compute the orientation and depth parameters of its neighboring points.

Below is the analytical solution for τ_2 in the perspective case

$$\tau_2 - \tau_1 = \tan^{-1}\left(-\frac{b}{a}\right) = \Delta\tau \ , \tag{5.27}$$

where

$$\begin{aligned} a &= \frac{\sin\sigma_1}{z_{w1}}(u_{i1}x_{i1} + v_{i1}y_{i1}) + \frac{f}{z_{w1}}\cos\sigma_1(u_{i1}\cos\tau_1 + v_{i1}\sin\tau_1) \\ &\qquad + \frac{f}{z_{w2}}(u_{i2}\cos\tau_1 + v_{i2}\sin\tau_1) \ , \qquad (5.28) \\ b &= \frac{f}{z_{w1}}(v_{i1}\cos\tau_1 - u_{i1}\sin\tau_1) + \frac{f}{z_{w2}}(u_{i2}\sin\tau_1 - v_{i2}\cos\tau_1) \ , \qquad (5.29) \end{aligned}$$

giving two solutions, as indicated by equation 5.25:

$$\tau_2 = \begin{cases} \tau_1 + \Delta\tau \\ \tau_1 + \Delta\tau + \pi \end{cases} \ . \tag{5.30}$$

The corresponding σ_2 is then given by

$$\sigma_2 = \tan^{-1}\left(\frac{c}{d}\right) + \cos^{-1}\left(\frac{|b+d|}{\sqrt{c^2+d^2}}\right) \ , \tag{5.31}$$

where b is as in equation 5.29,

$$c = \frac{1}{z_{w2}} \sin(\tau_2 - \tau_1)(u_{i2}x_{i2} + v_{i2}y_{i2}) \ , \tag{5.32}$$

$$d = \frac{f}{z_{w2}} \sin(\tau_2 - \tau_1)(u_{i2}\cos\tau_2 + v_{i2}\sin\tau_2) \ , \tag{5.33}$$

and where the quadrant of $\alpha = \tan^{-1}(\frac{c}{d})$ is given by knowing that

$$\begin{cases} \cos\alpha = \frac{d}{R} \\ \sin\alpha = \frac{c}{R} \end{cases} \quad \text{with} \quad R = \sqrt{c^2 + d^2} \ . \tag{5.34}$$

Equations 5.30 and 5.31 show an ambiguity in the tilt-direction even in the perspective case, although the solution should be unique. We explain this as due to solving a not-well-constrained problem, where the depth values, z_w, are inferred from the orientation parameters, rather than expressed as functions of those parameters and considered in the calculations.

The structure of the proposed algorithm follows naturally from equations 5.24, 5.26, 5.27, 5.31: a single LSF is used to propagate the orientation of a known point to the neighbouring image points and iteratively to all image points.

This structure raises the problem of determining a starting point of known orientation. A possible solution to the problem would be to assume that we know the orientation at a point, e.g. it has been given as input or it has been computed by using another visual process (e.g. stereo, motion, shading). Alternatively we assume that the texture image is showing a point which is frontal, i.e. parallel to the image plane and therefore of known orientation ($\sigma = 0$). In this case, the frontal point can be localised by using the algorithm described in section 5.1 and used as the starting point to this algorithm. In both cases the orientation of the starting point is used to compute the orientation of neighbouring points, and subsequently of all image pixels.

The estimated orientations σ's and τ's at all image points are integrated to recover the actual shape, i.e. z_w, by using the method for integration of Frankot and Chellappa ([50]). The method, introduced in 1988, is elegant, simple to implement and computationally efficient, and is therefore used in many surface-reconstruction algorithms, as described in section 2.5.

The described algorithm reconstructs a shape for each LSF. By using equations 5.23, 5.20 and 5.21 and the determined values σ, τ, z_w at the image pixels,

we can back-project the frequencies $\mathbf{u}_i$ to the surface texture and align them. The adopted homogeneity assumption (described in section 2.2) states that these back-projected frequencies belonging to the same LSF should be almost the same on the surface texture. As said in section 4.2, the assumption includes as a corollary that the variances of the back-projected and aligned frequencies, measured over the surface texture $\mathbf{x}_s$, are nearly zero. Hence we choose the shape associated with the lowest variance, assuming that lower values, closer to the ideal zero value, correspond to better estimates.

We summarize the general structure in algorithm 3.

Algorithm 3 Structure of the proposed algorithm to recover the shape of a developable surface from multiple estimated LSFs by using one LSF at a time, given a frontal point or a point of known orientation

- determine the coordinates of a frontal point, as described in section 5.1, or manually input the orientation at a point, if a frontal point is not available

for each estimated LSF **do**

- propagate the orientation estimation from the selected starting point to all image pixels (with equations 5.24, 5.26, 5.27, 5.31)
- integrate the orientations at the image pixels by using the algorithm of [50]
- back-project and align the frequency values for the considered LSF according to the computed orientations and shape (with equations 5.23, 5.20, 5.21)
- compute the variance of back-projected frequencies on the surface texture

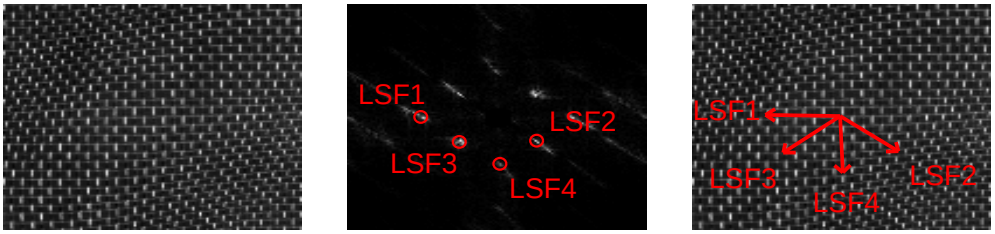
end for

- choose the best shape reconstruction estimation by selecting the LSF which gives the smallest variance

5.2.3 Results

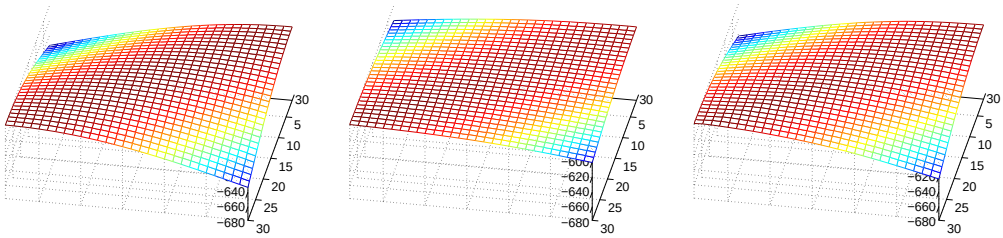
The proposed method, described in algorithm 3, is here applied to a semi-synthetic image and to a set of real images.

Firstly, we consider the semi-synthetic image, already presented in figure 3.10, obtained by mapping the Brodatz texture D6 ([18]) onto a cosine shape and re-



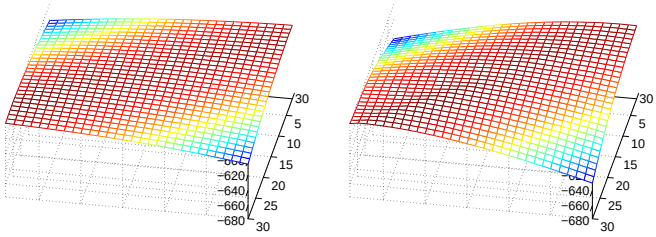
(a) (b) (c)

Figure 5.1: (a) 128x128 semi-synthetic picture obtained by rendering Brodatz texture D6 on a cosine surface, same as 3.9(b); (b) its spectrum with labels indicating the selected LSFs; (c) arrows indicating the respective spectral directions of the selected LSFs at the central pixel superposed on (a)



(a) (b) (c)

Figure 5.2: (a) Ground truth shape of the cosine surface in fig. 5.1(a), central 64x64 part; (b) orthographic and (c) perspective reconstructions achieved from the image in fig. 5.1(a) with algorithm 3 and LSF 2, central 64x64 part



(a) (b)

Figure 5.3: (a) Orthographic and (b) perspective reconstructions achieved from the image in fig. 5.1(a) with algorithm 3 and LSF 4, central 64x64 part

rendering it as a new image. Figures 5.1(a) and 5.1(b) show this image and its spectrum, on which we have labeled the main selected LSFs. Figure 5.1(c) indicates the corresponding spectral directions of the LSFs at the central pixel, superposed on the image, while the estimated values for all the selected LSFs have already been presented in figure 3.11.

Figure 5.2 shows the ground truth shape of the underlying cosine, and the best reconstructions (in the respective orthographic and perspective cases) that we achieved by applying the proposed algorithm 3 to LSF 2. Furthermore, figure 5.3 shows another pair of orthographic and perspective reconstructions, achieved by using a different LSF estimate, i.e. LSF 4.

The accuracy of the algorithm is demonstrated by both the figures and by the quantitative results in table 5.1, where the average errors for the orientation parameters (σ,τ) are reported for the central part of the reconstructed shape.

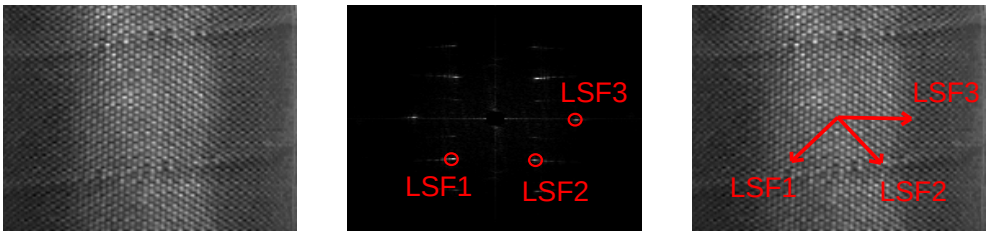
Please note that the frontal point used to achieve these results was determined automatically with the algorithm proposed in section 5.1. In this semi-synthetic case, as in all cases in the remainder of the chapter, the algorithm succeeded to detect and compute the frontal points. For the results presented here, no user interaction was therefore ever required.

A second set of images with known ground truth comprises real pictures that we acquired in the lab with a Pulnix TM-6EX camera, at a resolution of 640x480 with 256 grey levels (the camera also used for the planar pictures in figures 4.3 and 4.4).

Although different sets of real images with known ground truth have been produced in the case of planar surfaces (e.g. the set of images of Super and Bovik [158], of Massot and Hérault [121], of Ribeiro and Hancock [139]), there are no other sets of readily available real images with known ground truth of non-planar shapes.

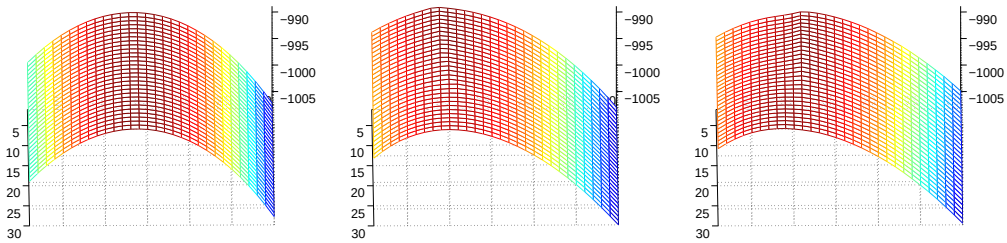
In the real pictures that we acquired, we obtained the ground truth by laying the textures on cylinders of different known diameters, which we arranged in front of the camera.

Figures 5.4(a) and 5.4(b), previously illustrated in figures 3.13(b) and 3.13(c), present the 180x180 image (extracted from the 640x480 picture in figure 3.13(a)) of a rubber rug laid on a metallic cylinder and its spectrum, where the three



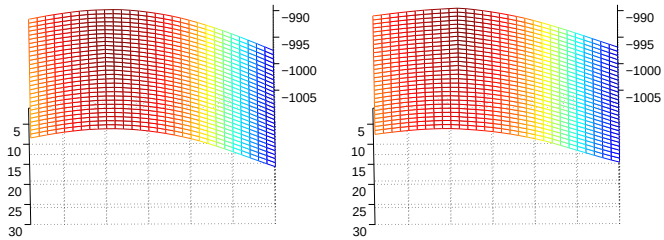
(a) (b) (c)

Figure 5.4: (a) 180x180 real image of a rubber rug laid on a cylinder extracted from 3.13(a), as in fig. 3.13(b); (b) its spectrum with labels indicating the selected LSFs; (c) arrows indicating the respective spectral directions of the selected LSFs at the central pixel superposed on (a)



(a) (b) (c)

Figure 5.5: (a) Ground truth shape of the cylinder beneath the rubber rug in fig. 5.4(a), central 90x90 part; (b) orthographic and (c) perspective reconstructions achieved from the image in fig. 5.4(a) with algorithm 3 and LSF 1, central 90x90 part



(a) (b)

Figure 5.6: (a) Orthographic and (b) perspective reconstructions achieved from the image in fig. 5.4(a) with algorithm 3 and LSF 3, central 90x90 part

(a) (b) (c)

Figure 5.7: (a) 640x480 real picture of a pair of trousers laid on a cylinder; (b) 128x128 real image extracted from (a); (c) spectrum of (b) with labels indicating the selected LSFs

main selected LSFs have been labeled. As for the case of texture D6, figure 5.4(c) illustrates the associated spectral directions at the central point superposed on the image, while the estimated LSFs have already been presented in figure 3.14.

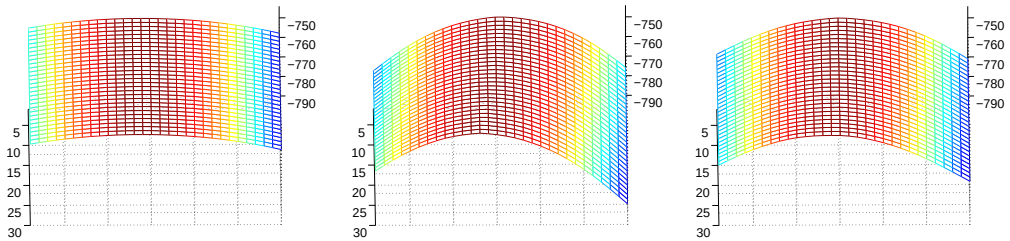
Figure 5.5 shows the ground truth shape of the underlying cylinder, and the best 3D reconstructions that we have achieved with the orthographic and perspective equations by using algorithm 3 and LSF 1.

As for texture D6, a second reconstruction (using both orthographic and perspective equations) is also given in figure 5.6 to confirm that different estimates of LSFs are often usable. Table 5.1 presents the average errors for the orientation parameters (σ,τ), confirming in both cases the accuracy of the algorithm.

The reconstruction is valid although the image contains reflections coming from the underlying metallic cylinder, and although the rug is an actual 3D structure with imperfections.

Figure 5.7(a) illustrates another example of a textured object, here a pair of trousers, laid on a cylinder. Figure 5.7(b) presents the analysed 128x128 central part of the original picture, while figure 5.7(c) shows its spectrum, with labels on the two selected LSFs.

The best reconstructions for this example, using orthographic and perspective equations, is obtained with LSF 2. Figure 5.8 illustrates the recovered shapes next to the true shape of the underlying cylinder, and a quantitative measure of the accuracy of the algorithm is given in table 5.1.

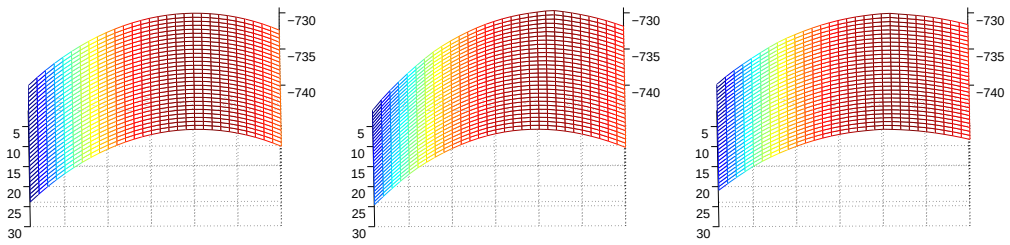


(a) (b) (c)

Figure 5.8: (a) Ground truth shape of the cylinder beneath the trousers in fig. 5.7(b), central 64x64 part; (b) orthographic and (c) perspective reconstructions achieved from the image in fig. 5.7(b) with algorithm 3 and LSF 2, central 64x64 part

(a) (b) (c)

Figure 5.9: (a) 640x480 real picture of a sponge laid on a cylinder; (b) 128x128 real image extracted from (a); (c) spectrum of (b) with labels indicating the selected LSFs



(a) (b) (c)

Figure 5.10: (a) Ground truth shape of the cylinder beneath the sponge in fig. 5.9(b), central 64x64 part; (b) orthographic and (c) perspective reconstructions achieved from the image in fig. 5.9(b) with algorithm 3 and LSF 2, central 64x64 part

Finally figure 5.9 presents the case of a sponge laid on another cylinder. As above we show the full 640x480 picture (5.9(a)), from which a 128x128 image is extracted (5.9(b)), and the image spectrum with labels indicating the selected LSFs (5.9(c)).

The qualitative and quantitative proof of the accuracy of algorithm is given both in figure 5.10, where the orthographic and perspective reconstructions can be compared with the ground truth, and in table 5.1.

Table 5.1 gathers all the average errors for the estimated σ and τ's of the presented images. As already noted, the small errors show the accuracy of the proposed algorithm (and of the LSF estimates), which is in fact in line with the present state of the art of SFT in [121], notwithstanding reflections and imperfections (e.g. figure 5.4(a)), variations in illumination and self-shadowing (e.g. figure 5.9(b)).

Furthermore, although the reconstructions in the figures are all close to the ground truth shape, it can be noted from the table how the perspective reconstructions are clearly more accurate than the orthographic reconstructions when the perspective effects are more evident, as in the case of the texture in figure 5.1(a).

On another note, the tilt-ambiguity is solved in the given cases by simply assuming convexity. However it is possible to use the algorithm on more complex

	Average error (degrees)			
	Orthographic case		**Perspective case**	
Image (utilised LSF)	$\lvert\epsilon_\sigma\rvert$	$\lvert\epsilon_\tau\rvert$	$\lvert\epsilon_\sigma\rvert$	$\lvert\epsilon_\tau\rvert$
D6 (LSF2)	4.92	0.72	0.90	0.70
D6 (LSF4)	8.49	0.49	2.49	0.37
rubber rug (LSF1)	4.24	3.86	4.41	4.09
rubber rug (LSF3)	7.56	4.20	9.34	4.17
trousers (LSF2)	1.46	5.87	2.83	5.71
sponge (LSF2)	1.28	4.21	2.38	4.06

Table 5.1: Average estimation errors for σ and τ of the reconstructions illustrated in figs.5.2, 5.3, 5.5, 5.6, 5.8, 5.10, achieved using algorithm 3 in the orthographic and perspective cases

developable shapes and implement an EM algorithm to disambiguate the tilt, as done for example by Lobay and Forsyth in [110].

Finally it is worth emphasising, as for the results presented in chapter 4, that the images were not retouched before processing.

5.2.4 Remarks on the algorithm

The proposed algorithm has the virtue of estimating a shape accurately with just a single LSF. However, it also has the flaws of only being applicable to developable surfaces and of propagating the orientation estimation using neighbouring points. If the orientation is misestimated at an image point, it is likely to be propagated to neighbouring points and will thus spoil the shape reconstruction.

Also, computing two unknowns (or more in the perspective case) with two equations means that no redundancy is considered, which may increase the chance of misestimates in the case of a noisy image.

The limitation to developable surfaces and the flaws due to the propagation process and the lack of redundancy are overcome by the second algorithm which we describe in section 5.3

5.3 A Frontal View and Two Local Spatial Frequencies

In this section, a second algorithm for SFT is proposed which recovers the general shape, either developable or non-developable, for each pair of LSFs, and then selects the best available reconstruction.

As for the previous algorithm of section 5.2, the algorithm presented here takes into account both orthographic and perspective camera models and requires that the image display a frontal point or a point of known orientation.

Accurate results are shown on the set of images presented in section 5.2, while an improved robustness due to the use of SVD and an effective redundancy is demonstrated on a new real video sequence.

This second algorithm draws on the original 3D reconstruction technique of Dugué and Elghadi ([42]). The technique, defined by the authors in the case of an orthographic camera model, is here extended to also consider perspective projections, and the advantages of the extension are discussed.

5.3.1 Viewing geometry and projection model revisited

Recall the viewing geometry of [158]: world coordinates $\mathbf{x}_w = (x_w, y_w, z_w)$, image coordinates $\mathbf{x}_i = (x_i, y_i)$, surface texture coordinates $\mathbf{x}_s = (x_s, y_s, z_s)$ and orientation parameters σ-τ. Originally defined for planar surfaces, it has been shown to easily adapt to the case of general shapes by using tangent planes in section 5.2.1. As therein noted, an assumption of smoothness is however required. Furthermore the $\mathbf{x}_s$ coordinates are defined at each surface texture point, on the tangent planes, according to the local orientation.

As seen in section 2.2, the chosen homogeneity assumption implies that the relevant LSFs are almost constant over the textured surface under analysis. However the dependency of the $\mathbf{x}_s$ coordinates on the tilt-direction, described in sections 4.1 and 5.2.1, complicates the use of the homogeneity assumption.

In order to compare the LSFs, in section 5.2.1 we aligned two back-projected values of a selected LSF at two points with tilts τ_1 and τ_2 by counter-rotating one by the difference of tilt angles $\Delta\tau = \tau_2 - \tau_1$.

Here we want to integrate the counter-rotation into the viewing geometry itself, so as to be able to compare the back-projected values of a LSF without any need to further align them.

This was proposed for the first time by Dugué and Elghadi under an orthographic projection camera model. In [42], they define an $\mathbf{x}_s$ coordinate system which is consistent on a virtually unfolded surface texture. Here we revise the viewing geometry and projection models already seen in sections 2.3, 4.1 and 5.2.1. We then present the orthographic relations between surface texture and image plane coordinates and frequencies of [42]. Finally we derive the respective relations under a perspective projection.

Given its previous definition in 2.3, an alignment for $\mathbf{x}_s$ independent from the tilt-direction is obtained by counter-rotating the system coordinates by the angle $-\tau$ on the x_s-y_s plane.

The new transformation matrix between the revised $\mathbf{x}_s$ and the $\mathbf{x}_w$ coordinate system is expressed by

$$\mathbf{x}_w = \begin{bmatrix} \cos\sigma\cos\tau & -\sin\tau & \sin\sigma\cos\tau \\ \cos\sigma\sin\tau & \cos\tau & \sin\sigma\sin\tau \\ -\sin\sigma & 0 & \cos\sigma \end{bmatrix} \begin{bmatrix} \cos\tau & \sin\tau & 0 \\ -\sin\tau & \cos\tau & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{x}_s + \begin{bmatrix} 0 \\ 0 \\ z_0 \end{bmatrix} , \tag{5.35}$$

where we now have an additional rotation matrix (compare equation 5.35 with equation 5.11).

Equation 5.35 can then be combined with the equation for perspective projection 4.3 and with the equation for orthographic projection 5.12 to get the respective relations between the image plane $\mathbf{x}_i$ and the surface texture $\mathbf{x}_s$ coordinates systems.

In particular, taking $\mathbf{x}_s$ to be (x_s, y_s) (since on the surface texture $z_s = 0$), we have in the perspective case

$$\mathbf{x}_i = \frac{f}{z_w} R(\tau)P(\sigma)R(-\tau)\mathbf{x}_s \ , \quad \text{with} \quad z_w = z_0 - \sin\sigma(x_s\cos\tau + y_s\sin\tau) \ , \tag{5.36}$$

$$R(\tau) = \begin{bmatrix} \cos\tau & -\sin\tau \\ \sin\tau & \cos\tau \end{bmatrix} \quad \text{and} \quad P(\sigma) = \begin{bmatrix} \cos\sigma & 0 \\ 0 & 1 \end{bmatrix} , \tag{5.37}$$

and its inverse

$$\mathbf{x}_s = \frac{z_w}{f} R(\tau) P^{-1}(\sigma) R(-\tau) \mathbf{x}_i \ , \tag{5.38}$$

$$\text{with} \quad z_w = \frac{z_0 f \cos\sigma}{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma} \ . \tag{5.39}$$

$R(\tau)$ and $P(\sigma)$ are respectively responsible for rotating the system of coordinates by τ and foreshortening the distances parallel to the tilt direction by a coefficient $\cos\sigma$. z_0, as explained in section 5.2.1, is the z_w coordinate of the intersection of the tangent plane at the considered point and the optical axis.

In the orthographic case, as shown by [42], we have

$$\mathbf{x}_i = R(\tau) P(\sigma) R(-\tau) \mathbf{x}_s \ , \tag{5.40}$$

and its inverse

$$\mathbf{x}_s = R(\tau) P^{-1}(\sigma) R(-\tau) \mathbf{x}_i \ , \tag{5.41}$$

where the rotation $R(\tau)$ and foreshortening $P(\sigma)$ matrices are the same as in equations 5.37.

As seen in sections 4.1 and 5.2.1, the transpose of the Jacobian (J^t) of equations 5.36 and 5.38 gives us the relation between the surface texture frequencies $\mathbf{u}_s = (u_s, v_s)$ and the image plane frequencies $\mathbf{u}_i = (u_i, v_i)$ in the perspective case under a first order approximation:

$$\mathbf{u}_s = J^t(\mathbf{x}_i, \mathbf{x}_s) \mathbf{u}_i \tag{5.42}$$

with

$$\begin{aligned} J^t(\mathbf{x}_i, \mathbf{x}_s) &= \begin{bmatrix} \frac{\partial x_i}{\partial x_s} & \frac{\partial y_i}{\partial x_s} \\ \frac{\partial x_i}{\partial y_s} & \frac{\partial y_i}{\partial y_s} \end{bmatrix} \\ &= \frac{\sin\sigma}{z_w} \begin{bmatrix} x_i \cos\tau & y_i \cos\tau \\ x_i \sin\tau & y_i \sin\tau \end{bmatrix} + \frac{f}{z_w} R(\tau) P(\sigma) R(-\tau) \ , \end{aligned} \tag{5.43}$$

and

$$\mathbf{u}_i = J^t(\mathbf{x}_s, \mathbf{x}_i) \mathbf{u}_s \tag{5.44}$$

with

$$J^t(\mathbf{x}_s, \mathbf{x}_i) = \begin{bmatrix} \frac{\partial x_s}{\partial x_i} & \frac{\partial y_s}{\partial x_i} \\ \frac{\partial x_s}{\partial y_i} & \frac{\partial y_s}{\partial y_i} \end{bmatrix}$$
$$= \frac{z_w}{f} \left\{ -\frac{\sin\sigma \begin{bmatrix} x_i \cos\tau & y_i \cos\tau \\ x_i \sin\tau & y_i \sin\tau \end{bmatrix}}{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma} + I \right\} R(\tau)P^{-1}(\sigma)R(-\tau) \ , \quad (5.45)$$

where z_w is the corresponding coordinate of the surface point which projects to the image point $\mathbf{x}_i$.

Equations 5.44 and 5.45 coincide with the relationship between image plane and surface texture frequencies found by Massot and Hérault in [121] (equation 20 in [121]), which confirms the result under a first order approximation.

In the orthographic case, as shown by [42], equations 5.42 and 5.44 are still valid, but the respective transpose Jacobians are computed from equations 5.40 and 5.41, resulting in the following:

$$J^t(\mathbf{x}_i, \mathbf{x}_s) = R(\tau)P(\sigma)R(-\tau) \quad (5.46)$$
$$J^t(\mathbf{x}_s, \mathbf{x}_i) = R(\tau)P^{-1}(\sigma)R(-\tau) \ . \quad (5.47)$$

On a final note, having aligned the $\mathbf{x}_s$ coordinates on a virtually unfolded surface texture, we can write the homogeneity assumption (compare equation 5.23) as:

$$\mathbf{u}_{s1} = \mathbf{u}_{s2} \quad (5.48)$$
$$J^t_{\mathbf{x}_i\mathbf{x}_s}(\mathbf{x}_{i1})\mathbf{u}_{i1} = J^t_{\mathbf{x}_i\mathbf{x}_s}(\mathbf{x}_{i2})\mathbf{u}_{i2} \quad (5.49)$$

where $\mathbf{u}_{s1}$ and $\mathbf{u}_{s2}$, the surface texture frequencies at the surface texture points $\mathbf{x}_{s1}$ and $\mathbf{x}_{s2}$, are the back-projections of $\mathbf{u}_{i1}$ and $\mathbf{u}_{i2}$, the image plane frequencies belonging to the same LSF measured at two distinct image plane points $\mathbf{x}_{i1}$ and $\mathbf{x}_{i2}$. As in equation 5.23, the slight difference of notation for the Jacobian indicates that it is obtained by differentiating the $\mathbf{x}_i$ (as expressed in equations 5.36 and 5.40) with respect to $\mathbf{x}_s$ and that it is a function of the orientation parameters at the particular image point $\mathbf{x}_i$.

5.3.2 Computing the 3D coordinates with singular value decomposition

The algorithm uses the revisited viewing geometry and projection model of section 5.3.1 to elegantly compute the orientation locally by using SVD. The method, originally proposed by [42] in the orthographic case, is here extended to perspective projections to enhance performance in cases where perspective effects are evident.

The algorithm explained here assumes that the image displays a frontal point or a point with known orientation. The redundancy given by using two LSFs provides robustness and applicability to non-developable surfaces.

Let us consider equation 5.42 for two different LSFs, $\mathbf{u}'_i = (u'_i, v'_i)$ and $\mathbf{u}''_i = (u''_i, v''_i)$, measured at the same image point $\mathbf{x}_i$:

$$\mathbf{u}'_s = J^t(\mathbf{x}_i, \mathbf{x}_s)\mathbf{u}'_i \quad \text{and} \quad \mathbf{u}''_s = J^t(\mathbf{x}_i, \mathbf{x}_s)\mathbf{u}''_i \ . \tag{5.50}$$

Equations 5.50 can be combined to write

$$U_s = J^t(\mathbf{x}_i, \mathbf{x}_s)U_i \quad \text{where} \quad U_s = \begin{bmatrix} u'_s & u''_s \\ v'_s & v''_s \end{bmatrix} \quad \text{and} \quad U_i = \begin{bmatrix} u'_i & u''_i \\ v'_i & v''_i \end{bmatrix} \ . \tag{5.51}$$

Here the 2x2 matrix U_s contains the surface texture frequencies corresponding to the two LSFs, constant over the surface texture according to the homogeneity assumption, while the 2x2 matrix U_i contains the image frequencies for the two LSFs, which are functions of the particular point $\mathbf{x}_i$.

In the orthographic case we substitute for $J^t(\mathbf{x}_i, \mathbf{x}_s)$ from 5.46 in the equation 5.51:

$$U_i U_s^{-1} = R(\tau)P^{-1}(\sigma)R(-\tau) = \begin{bmatrix} \cos\tau & -\sin\tau \\ \sin\tau & \cos\tau \end{bmatrix} \begin{bmatrix} \frac{1}{\cos\sigma} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\tau & \sin\tau \\ -\sin\tau & \cos\tau \end{bmatrix} . \tag{5.52}$$

The above system of four equations in the two unknowns (σ,τ) holds for each pixel of the image.

In the particular case of a frontal point ($\sigma = 0$), i.e. when the orientation of the tangent plane is parallel to the image plane, we have

$$U_s = U_i|_{\sigma=0} \ . \tag{5.53}$$

In other words the U_s matrix can be computed from the value of the two measured LSFs in U_i at a frontal point. In fact the location of the frontal point on the image plane may be estimated by using the same two LSFs, as indicated in section 5.1.

Alteratively, when the image does not display a frontal point (see discussion in section 5.1), the U_s can be computed by the manual input of the orientation of a particular point and then using equation 5.51. The value of the two considered texture frequencies can also be measured on a separately provided frontal patch of the same texture.

Since R is orthonormal and P^{-1} is diagonal, equation 5.52 can be compared with the SVD of the left hand side of the equation, according to which we write

$$U_i U_s^{-1} = V D V^{-1} = [\mathbf{v}_M \mathbf{v}_m] \begin{bmatrix} \lambda_M & 0 \\ 0 & \lambda_m \end{bmatrix} [\mathbf{v}_M \mathbf{v}_m]^{-1} \ , \tag{5.54}$$

with V the orthogonal matrix of the eigenvectors $(\mathbf{v}_M, \mathbf{v}_m)$ of $U_i U_s^{-1}$, and D the diagonal matrix of the corresponding eigenvalues $(|\lambda_M| > |\lambda_m|)$. Therefore we have

$$\sigma = \arccos\left(\frac{1}{\lambda_M}\right) \tag{5.55}$$

$$\tau = \angle(\mathbf{v}_M) \ . \tag{5.56}$$

In other words σ is computed from the bigger eigenvalue, while τ is oriented in the (unsigned, as for the tilt-ambiguity) direction of the corresponding eigenvector. This is also approximately valid when the surface is not completely unfoldable, in which case the term D_{22} of the diagonal matrix is not exactly 1, as explained in [42].

In the perspective case we substitute for $J^t(\mathbf{x}_i, \mathbf{x}_s)$ from equation 5.43 in equation 5.51:

$$\frac{f}{z_w}\left(I - Y(\sigma, \tau, z_w)\right)^{-1} U_i U_s^{-1} = \begin{bmatrix} \cos\tau & -\sin\tau \\ \sin\tau & \cos\tau \end{bmatrix} \begin{bmatrix} \frac{1}{\cos\sigma} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\tau & \sin\tau \\ -\sin\tau & \cos\tau \end{bmatrix} \tag{5.57}$$

where

$$Y(\sigma, \tau, z_w) = \frac{\sin\sigma}{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma} \begin{bmatrix} x_i \cos\tau & y_i \cos\tau \\ x_i \sin\tau & y_i \sin\tau \end{bmatrix} \ . \tag{5.58}$$

In the perspective case the matrix U_s can be easily computed from the matrix U_i of image plane frequencies at a frontal point (compare equation 5.53):

$$U_s = \frac{z_0}{f} U_i|_{\sigma=0} \ , \tag{5.59}$$

where f is the focal length and z_0 is the z_w coordinate of the frontal point.

As in the orthographic case, assuming we know the value of the left hand side of equation 5.57, we can decompose it by using the SVD:

$$\frac{f}{z_w}\left(I - Y(\sigma,\tau,z_w)\right)^{-1} U_i U_s^{-1} = VDV^{-1} = [\mathbf{v}_M \mathbf{v}_m] \begin{bmatrix} \lambda_M & 0 \\ 0 & \lambda_m \end{bmatrix} [\mathbf{v}_M \mathbf{v}_m]^{-1} \ , \tag{5.60}$$

from which we compute (σ,τ) using equations 5.55 and 5.56. However, since Y actually depends on (σ, τ, z_w), we propose the iterative solution illustrated in algorithm 4.

Algorithm 4 Proposed iterative algorithm to estimate the local orientations and shape in the perspective case by using SVD

- the slant σ and tilt τ are estimated with the orthographic equations (5.54,5.55 and 5.56)

repeat

- the computed σ's and τ's are integrated to recover the 3D shape, i.e. the z_w, with the method of [50] (see section 2.5)
- the matrix $Y(\sigma, \tau, z_w)$ of equation 5.58 and the left hand side of equation 5.57 are computed
- σ's and τ's are updated by decomposing the left hand side of equation 5.57 via SVD (equations 5.60, 5.55 and 5.56)

until σ's and τ's converge

The convergence of the algorithm has been verified empirically, furthermore we observe that matrix Y is usually small with respect to the unitary matrix I. In fact, a few iterations are usually enough to reach convergence, and this is achieved even when initializing the algorithm with the wrong (σ,τ,z_w) estimate.

Finally, for both the orthographic and perspective cases, the shape is recovered from the estimated τ's and σ's at the image pixels by using the integration method of [50], as discussed in section 2.5.

A reconstructed shape is therefore obtained for each pair of LSFs. By using equation 5.51 and the determined (σ,τ,z_w), we can back-project the frequencies $\mathbf{u}_i$ to the surface texture. Using the homogeneity assumption (see section 2.2) and the theoretical zero value of the variance of the back-projected frequencies (as in section 5.2.2), we choose the best reconstructed shape according to the lowest variance.

We summarize the general proposed structure in algorithm 5.

Algorithm 5 Structure of the proposed algorithm to recover the 3D shape from multiple estimated LSFs by using pairs of LSFs via SVD, given a frontal point or a point of known orientation or an additional frontal patch of the texture

for each pair of estimated LSFs **do**

- compute the surface texture values of the respective frequencies in U_s by using a frontal point (determined as indicated in section 5.1) and equations 5.53 or 5.59, by using a point of known orientation and equation 5.51, or directly by measuring the frequencies on a additionally input frontal patch of the same texture
- compute the orientation parameters (σ,τ) at each pixels by using SVD and equations 5.54, 5.55 and 5.56 and integrate the orientations with the method of [50] to recover the 3D shape z_w, iterating as indicated by algorithm 4 in the perspective case;
- back-project the LSFs to the surface texture by using the computed orientations (σ,τ) and shape (z_w) and equation 5.51
- compute the variance of back-projected frequencies on the surface texture

end for

- choose the best shape reconstruction estimation by selecting the pair of LSFs which determines the smallest variance

Using two LSFs does not reduce the applicability of the proposed method, because most real textures have multiple frequencies, as already argued in chapter 2.

However, the over-complete systems of equations and the mathematical structure imposed by the SVD provide increased robustness to noise in the orientation estimation at each pixel, as shown section 5.3.3.

	Average error (degrees)			
	Orthographic case		**Perspective case**	
Image (utilised LSF)	$\lvert\epsilon_\sigma\rvert$	$\lvert\epsilon_\tau\rvert$	$\lvert\epsilon_\sigma\rvert$	$\lvert\epsilon_\tau\rvert$
D6 (LSFs2-4)	6.07	9.65	1.78	6.56
rubber rug (LSFs1-3)	3.28	5.23	2.92	5.45
trousers (LSFs1-2)	2.19	5.14	2.14	5.50
sponge (LSFs1-2)	1.11	4.26	1.07	4.48

Table 5.2: Average estimation errors for σ and τ of the reconstructions illustrated in fig. 5.11, achieved using the algorithm of 5 in the orthographic and perspective cases

In addition to this, the orientation estimation is achieved at each pixel independently, and therefore homogeneity can be assumed only locally, allowing us to reconstruct non-developable shapes, as argued in [42] and demonstrated by the results of section 5.3.3.

Independent estimates prevent single errors from spoiling the whole reconstruction (compare with section 5.2.4), while the enforced integrability of [50], used to recover the shape, generally discards single errors. In this way the flaws due to orientation error propagation are also solved and the algorithm can achieve more challenging reconstructions robustly.

We also note that the extension of the algorithm to consider perspective cases not only improves the results when perspective effects are evident in the considered image, but also allows us to generalize the applicability of the algorithm to planar textures. As explained in section 2.4, orthographic projections provide no information on the orientation of a planar surface. However considering perspective effects the method should also apply to planar surfaces, provided that the orientation of an initial point is given as input (here a frontal point clearly cannot be used).

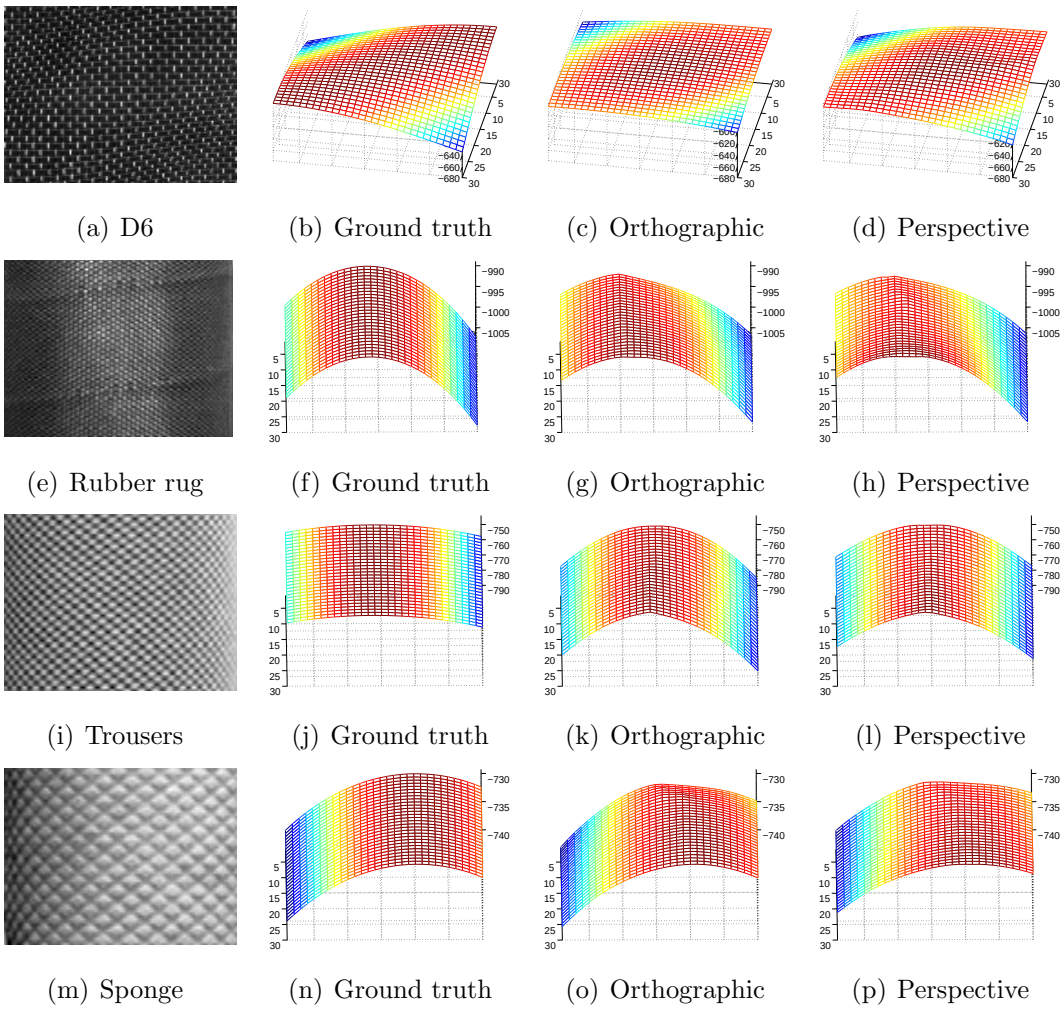


Figure 5.11: Semi-synthetic and real images of figs.5.1(a), 5.4(a), 5.7(b), 5.9(b), respective ground truth and reconstructed shapes achieved by using algorithm 5 in orthographic and perspective cases, with LSFs 2-4 ((c), (d)), 1-3((g), (h)), 1-2((k), (l)), 1-2((o), (p)). As in section 5.2.3, the central parts of the ground truth and reconstructed shapes are represented

5.3.3 Results and comparison

The proposed method, described by algorithm 5, is applied here to the semi-synthetic and real images presented in section 5.2.3.

Figure 5.11 shows the images and ground truth shapes, already illustrated in section 5.2.3, and the 3D reconstructions obtained with the proposed algorithm 5, using the orthographic and perspective equations (as indicated in figure).

These results can be directly compared with the reconstructions of section 5.2.3, because the labels for the LSFs are the same (they are indicated in figures 5.1, 5.4, 5.7, 5.9) and because the figures represent the same central portion of the 3D shapes, with the coordinate axes equally scaled and with the same camera poses. Furthermore, as noted in the cited result section, the frontal points have been automatically estimated for all the cases in this chapter with the algorithm of section 5.1.

As observed for the reconstructions of section 5.2.3, the reconstructions of figure 5.11 also show a good accuracy, notwithstanding reflections and imperfections (e.g. figure 5.11(e)), variations in illumination and self-shadowing (e.g. figure 5.11(m)).

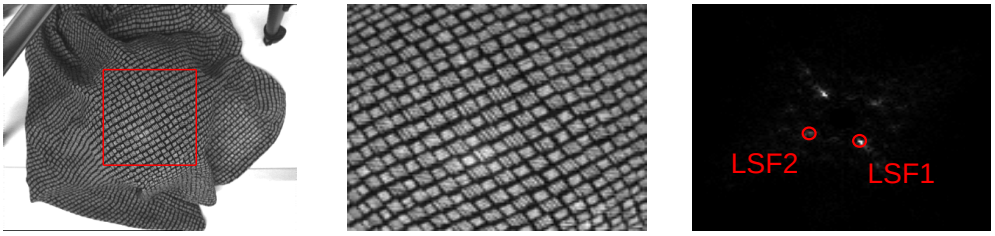
This is confirmed by table 5.2, which shows the average errors in the estimation of the orientation parameters (σ,τ), achieved using algorithm 5 and the orthographic and perspective equations.

As for algorithm 3, the results obtained by using algorithm 5 with the perspective equations are clearly more accurate than the results with the orthographic equations when the perspective effects are more evident, as in the case of the texture in figure 5.11(a).

The results of table 5.2 are similar to those presented in table 5.1, and in line with the present state of the art of SFT in [121].

The higher error in table 5.2, compared to table 5.1, is ameliorated by an increased robustness and by the applicability of algorithm 5 to non-developable shapes.

Next we introduce a new set of images, which we extract from a video sequence of a textured cloth laid on a slowly deflating balloon.



(a) (b) (c)

Figure 5.12: (a) Frame 1 extracted from a 400x300 real video sequence of a textured cloth laid on a deflating balloon; (b) 128x128 real image extracted from (a); (c) spectrum of (b) with labels indicating the two selected LSFs

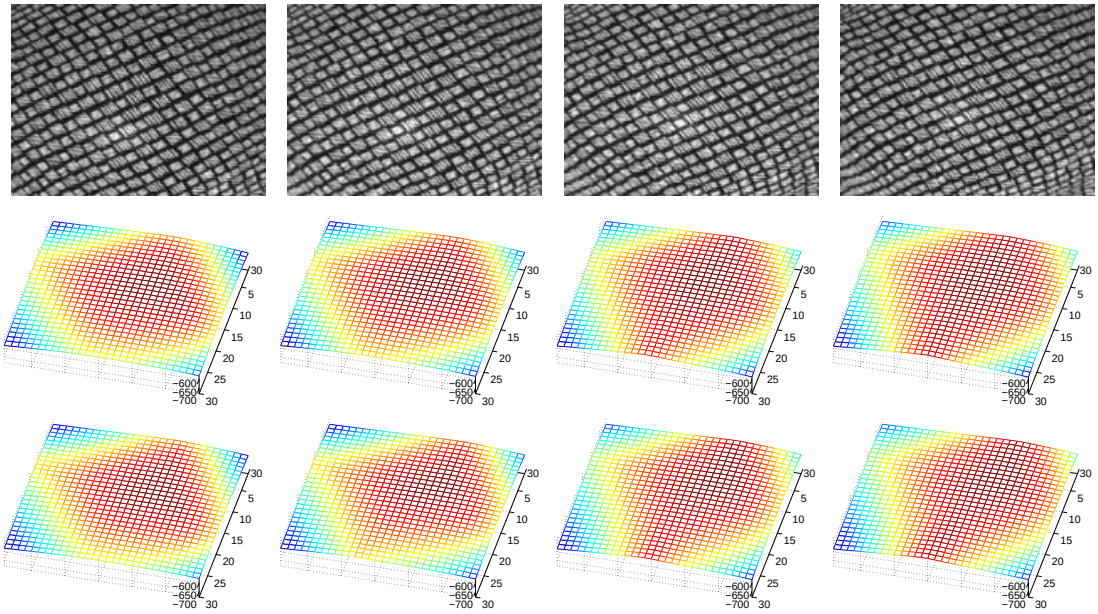


Figure 5.13: 128x128 real images extracted from the frames of a video sequence of a textured cloth laid on a deflating balloon. Top row: frames 1,100,200,300; Mid row: respective orthographic reconstructions using algorithm 5 on LSFs 1 and 2; Bottom row: corresponding perspective reconstructions using algorithm 5 on LSFs 1 and 2

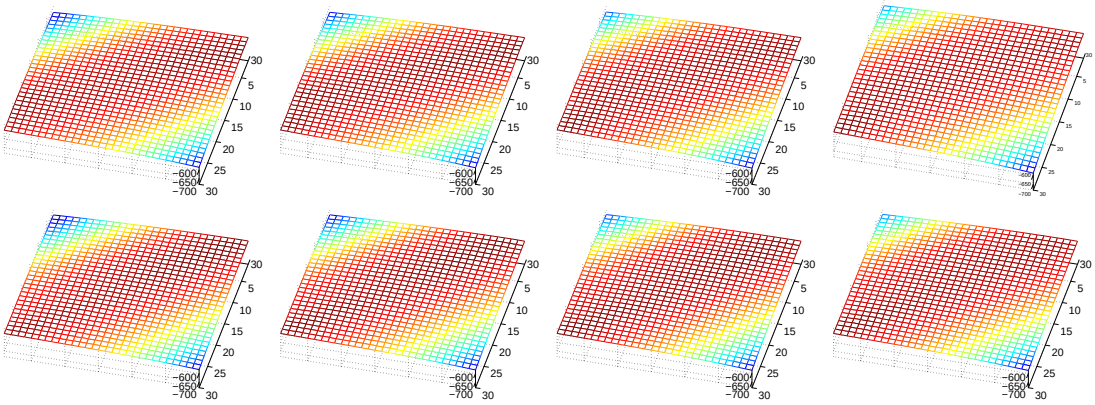


Figure 5.14: First row: respective orthographic reconstructions of the frames (1,100,200,300) in figure 5.13 using algorithm 3 on LSF 1; Second row: respective perspective reconstructions using algorithm 3 on LSF 1

The video was shot using the same Pulnix TM-6EX camera, at a resolution of 400x300 with 256 grey levels, and figures 3.15 and 3.16 already presented the first frame of the real video sequence, its spectrum and the estimates of the two main LSFs.

Figure 5.12 re-illustrates here the whole 400x300 picture of the first frame of the sequence (figure 5.12(a)), the 128x128 analysed image extracted from its central part (figure 5.12(b)), and the spectrum of this image (figure 5.12(c)) with the two main LSFs labeled.

Figure 3.16 showed the estimates of the two main LSFs at the image pixels of the first frame, which appeared to correspond well to the expected LSF changes observable in the image sequence. However this new set of images appears to be particularly challenging for a number of reasons:

- the image quality is non-optimal and looks granular, possibly because of some sampling and quantization effects, which add to the intrinsic noise always present in real images;

- the textured cloth, therefore an intrinsically developable surface, is laid on a balloon, therefore on a mostly spherical, non-developable shape. As a consequence of laying the planar cloth on the sphere, we expect some

stretching, due to the nature of the textile, and some creases. In addition, the changing shape due to deflation should alter the stretches and creases during the deflation process;

- a sphere is the perfect example of a shape for which the tilt and slant change from point to point taking a wide range of values. In particular, the tilt takes all values from $-180°$ and $180°$, as it is directed from the frontal point to the edges.

As a result of these factors, we expect the estimated distortion measure, i.e. the LSFs, to be noisy, respecting the homogeneity assumption only locally, and therefore difficult to analyse for any reconstruction algorithm.

Every frame from the video sequence was processed independently from the others, i.e. as a single image. For each frame the LSFs were estimated (with the method described in chapter 3) and the 3D shapes were reconstructed by using algorithm 5 and the orthographic and perspective equations on the two available LSFs. An output video was produced with the recovered 3D shapes (please see the DVD accompanying the thesis).

Figure 5.13 presents four of the analysed frames (1,100,200,300) in the top row, and the respective reconstructions achieved by using the orthographic equations (second row) and the perspective equations (third row).

Despite the difficulties described, the reconstructions are plausible and succeed in adequately representing the shape in each picture. Moreover both our LSF estimation algorithm (described in chapter 3) and the reconstruction method of algorithm 5 are able to detect the small changes in the shape from frame to frame.

Figure 5.14 shows the reconstructions of the same four frames obtained by using the single-LSF-algorithm 3 and the orthographic (first row) and perspective equations (second row). As we see from the figure, the algorithm is not suitable for dealing with non-developable shapes, as it relies on the propagation of the orientation estimation from an initial point.

Furthermore, as already mentioned in section 5.2.4, algorithm 3 does not consider any redundancy the use of which might increase the robustness against noise. In addition to this, the propagation of the orientation parameters from an initial point implies the risk of propagating errors, which may spoil the whole

reconstruction. In fact algorithm 3 could not provide a reconstruction for all the frames of the video sequence.

For algorithm 5, as well as for algorithm 3, the well known tilt ambiguity is solved assuming convexity. However, as mentioned in section 5.2.3, it is possible to implement an EM algorithm to disambiguate the tilt and extend the algorithm to more complex shapes, as done for example by Lobay and Forsyth in [110].

As a final note, we again underline that each frame has been processed independently. However the changes between close frames are consistent and smooth, and describe well the small shape changes due to the deflation of the balloon in the video sequence.

5.4 Final Remarks

This chapter has presented two algorithms, both of which use a multi-scale distortion based on LSFs to recover the 3D coordinates of textured images, under the assumption that the image displays a frontal point or a point of known orientation or that a separate frontal patch of the same texture is available.

The first algorithm has the advantage of working with a single LSF. However it has the flaws of propagating the orientation estimation through neighbouring points, not using a redundant LSF in the estimation process, and only being applicable to developable surfaces, as already remarked on in section 5.2.4.

The second algorithm can effectively consider pairs of LSFs, using redundancy and SVD to achieve better robustness to noise. It estimates the orientation of each pixel independently and is applicable to non-developable shapes.

The two algorithms are presented in forms using both the orthographic and the more general perspective camera models.

In order to test the algorithm with real non-planar images of known ground truth, we have acquired pictures of various textured objects laid on different cylinders with known diameter. The new dataset fills in a gap in the SFT literature, providing real images of non-planar surfaces with known ground truth.

Both the algorithms are accurate, with accuracy in line with the present state of the art of SFT, and robust against variations in illumination, self shadowing, reflections and imperfections in the texture pattern.

In addition, the second algorithm, also tested on a real video sequence of a textured cloth laid on a balloon, showed robustness against creases and described well the small shape changes due to the deflation of the balloon. As already mentioned in section 1, this may open the way to using laid-on textures as a non-invasive way of studying shape changes of objects.

However, both the algorithms suffer from their reliance on the accuracy with which a frontal point is determined, which influences all the orientation estimates.

The requirement of a frontal point or a point of known orientation is removed in the next chapter where we also explain why the 3D reconstructions in figure 5.14 seem to be so strongly influenced by the spectral direction of the LSF (LSF 1) used.

Chapter 6

Local 3D Shape Reconstruction from Local Frequency Variations

This chapter presents the recovery of Shape-From-Texture using multi-scale distortion estimation based on LSFs in the case of general shapes.

We introduce a 3D reconstruction technique, based on local frequency variations and eigendecomposition, which uses pairs of LSFs and then selects the best available reconstruction.

The algorithm does not require any further assumption about the shape underlying the texture, and is applicable to developable and non-developable, planar and curved surfaces.

Inspiration was drawn from the work of Massot and Hérault ([121]), who estimated and used a single LSF. This chapter shows our original derivation of general perspective equations which apply to two LSFs.

The proposed algorithm is tested on a semi-synthetic image, on a set of real images and on a real video sequence.

Through the examples, the accuracy and robustness of the algorithm is shown. In particular the accuracy of the results is in line with the state of the art of SFT, while the applicability is a considerable advance in the current SFT literature.

6.1 Computing Local 3D Coordinates from Local Frequency Variations

This section illustrates how the proposed algorithm processes the texture multi-scale distortion to recover the 3D shape from each pair of LSFs and then selects the best available reconstruction.

No further assumption is made apart from the 'stochastic' homogeneity of Malik and Rosenholtz ([118]) described in section 2.2, and the shape is recovered locally by the use of local frequency variations of the two considered LSFs. In fact the use of eigendecomposition and the redundancy given by two LSFs provide robustness and applicability to general surfaces, developable and non-developable, planar and curved.

Two equivalent techniques are described to compute the orientation parameters (σ,τ) at a pixel and two other equivalent techniques are given in appendix A. Finally a discussion of the techniques and the general structure of the proposed algorithm is presented.

6.1.1 An equation for local frequency variations

The algorithm relies on the revisited viewing geometry and projection model of section 5.3.1:

- the viewing geometry of [158], with world coordinates $\mathbf{x}_w = (x_w, y_w, z_w)$, image coordinates $\mathbf{x}_i = (x_i, y_i)$, surface texture coordinates $\mathbf{x}_s = (x_s, y_s)$ and orientation parameters σ,τ;
- adapted to deal with non-planar surfaces, by approximating them with the tangent plane at each considered point;
- with an integrated counter-rotation as indicated by [42], so as to have the $\mathbf{x}_s$ coordinate system consistent on a virtually unfolded surface texture.

In this viewing geometry, under the stochastic homogeneity assumption, frequencies pertaining to the same LSF back-projected onto the surface texture are in theory equal (see equations 5.48 and 5.49).

As seen in equation 5.50, we can write the equality relation for two different LSFs, $\mathbf{u}'_i = (u'_i, v'_i)$ and $\mathbf{u}''_i = (u''_i, v''_i)$, and then combine the relations into one matrix equation. This was shown in equation 5.51, and we repeat it below:

$$U_s = J^t(\mathbf{x}_i, \mathbf{x}_s)U_i \quad \text{where} \quad U_s = \begin{bmatrix} u'_s & u''_s \\ v'_s & v''_s \end{bmatrix} \quad \text{and} \quad U_i = \begin{bmatrix} u'_i & u''_i \\ v'_i & v''_i \end{bmatrix} . \tag{6.1}$$

In this equation, the 2x2 matrix U_s contains the surface texture frequencies corresponding to the two LSFs, which are constant over the surface texture according to the homogeneity assumption. The 2x2 matrix U_i contains the image frequencies for the two LSFs, which are functions of the particular point $\mathbf{x}_i$.

Let us consider the value of the Jacobian, as expressed in the general perspective case by equation 5.43. By substituting it in equation 6.1 we have

$$U_s = \frac{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma}{z_0 f \cos\sigma} \left\{ \sin\sigma \begin{bmatrix} \cos\tau \\ \sin\tau \end{bmatrix} \begin{bmatrix} x_i & y_i \end{bmatrix} + fR(\tau)P(\sigma)R(-\tau) \right\} U_i \ . \tag{6.2}$$

Note that we have also substituted for z_w from equation 5.39, so as to have the relation expressed as a function of z_0. In fact, z_0, as explained in section 5.2.1, is the z_w coordinate of the intersection of the tangent plane at the considered surface texture point and the optical axis. z_0 is therefore constant when we consider the first order approximation at a given surface texture point, i.e. z_0 is constant once the particular tangent plane is considered.

In the remaining part of the section, we show how to derive the relation between the local shape orientation and the local frequency variations. The result is given at equation 6.8, where the local orientation (τ and σ at the pixel) is related to the image frequencies (U_i) and to its first derivatives (indicated with $\partial_{\mathbf{s}} U_i$). Before we conclude the section, we also show an alternative derivation of equation 6.8, which uses equation 5.44 instead of equation 5.51.

The derivation is based on differentiating both the left and right hand sides of equation 6.2. To this purpose, let us indicate the general direction of differentiation with the direction of an arbitrary vector $\mathbf{s}$.

However, before we proceed, let us note that equation 6.2 was derived by using the Jacobian $J^t(\mathbf{x}_i, \mathbf{x}_s)$ of the coordinate transformation in equation 5.36. As

previously described, using of the Jacobian corresponds to approximating to first order. In other words, the surface texture at the considered surface texture point is approximated with the tangent plane at the point. This implies that in the further differentiation we can consider the variables τ,σ,z_0 and f as constants, as they do not vary on the tangent plane at the surface texture point under analysis.

We also note that $\partial_{\mathbf{s}} U_s = 0$, because U_s contains surface texture frequencies which are constant over the surface according to the homogeneity assumption of section 2.2 and to the alignment described in section 5.3.1.

Finally, in our calculations we define the expression $\partial_{\mathbf{s}} U_i$ to indicate the derivative in the general direction, indicated by the vector $\mathbf{s}$, of every element of the matrix U_i:

$$\partial_{\mathbf{s}} U_i = \begin{bmatrix} \partial_{\mathbf{s}} u_i' & \partial_{\mathbf{s}} u_i'' \\ \partial_{\mathbf{s}} v_i' & \partial_{\mathbf{s}} v_i'' \end{bmatrix} \quad \text{or equivalently} \quad \partial_{\mathbf{s}} U_i = \begin{bmatrix} \mathbf{s} \cdot \nabla u_i' & \mathbf{s} \cdot \nabla u_i'' \\ \mathbf{s} \cdot \nabla v_i' & \mathbf{s} \cdot \nabla v_i'' \end{bmatrix} . \tag{6.3}$$

The differentiation is given explicitly below:

$$\begin{aligned} 0 = {} & \frac{\sin\sigma(\partial_{\mathbf{s}} x_i \cos\tau + \partial_{\mathbf{s}} y_i \sin\tau)}{z_0 f \cos\sigma} \\ & \left\{ \sin\sigma \begin{bmatrix} \cos\tau \\ \sin\tau \end{bmatrix} \begin{bmatrix} x_i & y_i \end{bmatrix} + f R(\tau) P(\sigma) R(-\tau) \right\} U_i \\ & + \frac{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma}{z_0 f \cos\sigma} \sin\sigma \begin{bmatrix} \cos\tau \\ \sin\tau \end{bmatrix} \begin{bmatrix} \partial_{\mathbf{s}} x_i & \partial_{\mathbf{s}} y_i \end{bmatrix} U_i \\ & + \frac{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma}{z_0 f \cos\sigma} \\ & \left\{ \sin\sigma \begin{bmatrix} \cos\tau \\ \sin\tau \end{bmatrix} \begin{bmatrix} x_i & y_i \end{bmatrix} + f R(\tau) P(\sigma) R(-\tau) \right\} \partial_{\mathbf{s}} U_i . \end{aligned} \tag{6.4}$$

We note that

$$R(\tau) P^{-1}(\sigma) R(-\tau) \begin{bmatrix} \cos\tau \\ \sin\tau \end{bmatrix} = \frac{1}{\cos\sigma} \begin{bmatrix} \cos\tau \\ \sin\tau \end{bmatrix} \tag{6.5}$$

and multiply equation 6.4 on the left by $z_w R(\tau) P^{-1}(\sigma) R(-\tau)$:

$$\begin{aligned} 0 = {} & \frac{\sin\sigma(\partial_{\mathbf{s}} x_i \cos\tau + \partial_{\mathbf{s}} y_i \sin\tau)}{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma} \left\{ \tan\sigma \begin{bmatrix} \cos\tau \\ \sin\tau \end{bmatrix} \begin{bmatrix} x_i & y_i \end{bmatrix} + f I \right\} U_i \\ & + \tan\sigma \begin{bmatrix} \cos\tau \\ \sin\tau \end{bmatrix} \begin{bmatrix} \partial_{\mathbf{s}} x_i & \partial_{\mathbf{s}} y_i \end{bmatrix} U_i + f \left\{ \frac{\tan\sigma}{f} \begin{bmatrix} \cos\tau \\ \sin\tau \end{bmatrix} \begin{bmatrix} x_i & y_i \end{bmatrix} + I \right\} \partial_{\mathbf{s}} U_i . \end{aligned} \tag{6.6}$$

Again we multiply on the left by the following expression

$$\left[\frac{\tan\sigma}{f}\begin{bmatrix}\cos\tau\\ \sin\tau\end{bmatrix}\begin{bmatrix}x_i & y_i\end{bmatrix}+I\right]^{-1}$$
$$=\frac{f\cos\sigma}{\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma}\left\{\frac{\tan\sigma}{f}\begin{bmatrix}y_i\\ -x_i\end{bmatrix}\begin{bmatrix}\sin\tau & -\cos\tau\end{bmatrix}+I\right\}, \tag{6.7}$$

and we finally obtain the expression for local frequency variation

$$\partial_{\mathbf{s}}U_i=-\frac{\sin\sigma}{\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma}$$
$$\left\{(\partial_{\mathbf{s}}x_i\cos\tau+\partial_{\mathbf{s}}y_i\sin\tau)I+\begin{bmatrix}\cos\tau\\ \sin\tau\end{bmatrix}\begin{bmatrix}\partial_{\mathbf{s}}x_i & \partial_{\mathbf{s}}y_i\end{bmatrix}\right\}U_i\ . \tag{6.8}$$

As previously said, the same result can be obtained by considering equation 5.44 for the differentiation.

Let us consider equation 5.44 for two different LSFs $\mathbf{u}'_i=(u'_i,v'_i)$ and $\mathbf{u}''_i=(u''_i,v''_i)$ measured at the same image point $\mathbf{x}_i$:

$$\mathbf{u}'_i=J^t(\mathbf{x}_s,\mathbf{x}_i)\mathbf{u}'_s \quad\text{and}\quad \mathbf{u}''_i=J^t(\mathbf{x}_s,\mathbf{x}_i)\mathbf{u}''_s\ , \tag{6.9}$$

which we combine to write

$$U_i=J^t(\mathbf{x}_s,\mathbf{x}_i)U_s\ , \tag{6.10}$$

where U_i and U_s, as defined in equation 6.1, are the respective matrices of image frequencies and surface texture frequencies.

By substituting into equation 6.10 for the value of the Jacobian from equation 5.45 we have

$$U_i=\frac{z_0\cos\sigma}{\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma}$$
$$\left\{-\frac{\sin\sigma\begin{bmatrix}\cos\tau\\ \sin\tau\end{bmatrix}\begin{bmatrix}x_i & y_i\end{bmatrix}}{\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma}+I\right\}R(\tau)P^{-1}(\sigma)R(-\tau)U_s\ , \tag{6.11}$$

where, as for equation 6.2, we also have substituted for z_w from equation 5.39, so as to have the relation expressed as a function of z_0 (constant under a first order approximation).

As for equation 6.2, both the left and right hand sides of equation 6.11 are differentiated in a general direction, which we take to be the direction of the arbitrary vector $\mathbf{s}$.

As noted before, in the differentiation the variables τ,σ,z_0 and f can be considered constant, $\partial_{\mathbf{s}} U_s = 0$, and $\partial_{\mathbf{s}} U_i$ indicates the differentiation of each element of the matrix (see equation 6.3).

For the sake of clarity, we first show the differentiation of the first two factors of the right hand side of equation 6.11:

$$\begin{aligned}
&\partial_{\mathbf{s}} \left[\frac{z_0 \cos\sigma}{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma} \right] \\
&= \left\{ -\frac{\sin\sigma(\partial_{\mathbf{s}} x_i \cos\tau + \partial_{\mathbf{s}} y_i \sin\tau)}{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma} \right\} \frac{z_0 \cos\sigma}{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma}
\end{aligned} \tag{6.12}$$

and

$$\begin{aligned}
&\partial_{\mathbf{s}} \left[-\frac{\sin\sigma \begin{bmatrix} \cos\tau \\ \sin\tau \end{bmatrix} \begin{bmatrix} x_i & y_i \end{bmatrix}}{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma} + I \right] \\
&\qquad = \frac{\sin^2(\sigma)(\partial_{\mathbf{s}} x_i \cos\tau + \partial_{\mathbf{s}} y_i \sin\tau) \begin{bmatrix} \cos\tau \\ \sin\tau \end{bmatrix} \begin{bmatrix} x_i & y_i \end{bmatrix}}{\left[\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma\right]^2} \\
&\qquad - \frac{\sin(\sigma)(\sin(\sigma)(x_i \cos\tau + y_i \sin\tau) + f\cos(\sigma)) \begin{bmatrix} \cos\tau \\ \sin\tau \end{bmatrix} \begin{bmatrix} \partial_{\mathbf{s}} x_i & \partial_{\mathbf{s}} y_i \end{bmatrix}}{\left[\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma\right]^2} \\
&\qquad = \left\{ -\frac{\sin\sigma \begin{bmatrix} \cos\tau \\ \sin\tau \end{bmatrix} \begin{bmatrix} \partial_{\mathbf{s}} x_i & \partial_{\mathbf{s}} y_i \end{bmatrix}}{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma} \right\} \\
&\qquad\qquad \left\{ -\frac{\sin\sigma \begin{bmatrix} \cos\tau \\ \sin\tau \end{bmatrix} \begin{bmatrix} x_i & y_i \end{bmatrix}}{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma} + I \right\} .
\end{aligned} \tag{6.13}$$

We then combine $\partial_{\mathbf{s}} U_s = 0$ and the two above expressions 6.12 and 6.13 to finally obtain the equation for local frequency variation:

$$\partial_{\mathbf{s}} U_i = -\frac{\sin\sigma}{\sin\sigma(x_i\cos\tau + y_i\sin\tau) + f\cos\sigma} \left\{ (\partial_{\mathbf{s}} x_i \cos\tau + \partial_{\mathbf{s}} y_i \sin\tau) I + \begin{bmatrix} \cos\tau \\ \sin\tau \end{bmatrix} \begin{bmatrix} \partial_{\mathbf{s}} x_i & \partial_{\mathbf{s}} y_i \end{bmatrix} \right\} U_i \,. \quad (6.14)$$

where we have back-substituted for the value of U_i as given by equation 6.11.

As expected, equation 6.14 is the same as equation 6.8. These equations coincide with equation 21 in [121], relating the frequency measured on the image plane and its derivative to the orientation parameters. In particular, a_i in [121] is expressed according to our coordinate system and f_i, a single frequency, is replaced by U_i, a matrix with two LSFs.

We write equations 6.14 and 6.8 as

$$-\left(\partial_{\mathbf{s}} U_i\right) U_i^{-1} = \frac{\sin\sigma}{\sin\sigma(x_i\cos\tau + y_i\sin\tau) + f\cos\sigma} \begin{bmatrix} 2\partial_{\mathbf{s}} x_i \cos\tau + \partial_{\mathbf{s}} y_i \sin\tau & \partial_{\mathbf{s}} y_i \cos\tau \\ \partial_{\mathbf{s}} x_i \sin\tau & \partial_{\mathbf{s}} x_i \cos\tau + 2\partial_{\mathbf{s}} y_i \sin\tau \end{bmatrix} \,. \quad (6.15)$$

Equation 6.15 relates the orientation parameters $(\sigma,\ \tau)$ at a point $\mathbf{x}_i$ to the local frequency variation, expressed by the matrix U_i and by its derivative $\partial_{\mathbf{s}} U_i$. It is solved for the local orientation parameters $(\sigma,\ \tau)$ in the following section 6.1.2.

6.1.2 Solving for slant and tilt locally

Here we solve for the value of the slant σ and tilt τ at a given image point $\mathbf{x}_i$, in terms of the local values of two LSFs (expressed by U_i) and of their local variations (expressed by $\partial_{\mathbf{s}} U_i$).

In particular, here we illustrate two methods for solving equation 6.15. Appendix A illustrates two equivalent methods, which solve for σ and τ using equation 6.11.

6.1.2.1 With eigenvalues and eigenvectors

First we solve equation 6.15 by using eigendecomposition.

The eigenvalues of the expression on the right hand side of equation 6.15 are

$$\lambda_1 = \frac{\sin\sigma(2\partial_{\mathbf{s}}x_i\cos\tau + 2\partial_{\mathbf{s}}y_i\sin\tau)}{\sin\sigma(x_i\cos\tau + y_i\sin\tau) + f\cos\sigma} \tag{6.16}$$

$$\lambda_2 = \frac{\sin\sigma(\partial_{\mathbf{s}}x_i\cos\tau + \partial_{\mathbf{s}}y_i\sin\tau)}{\sin\sigma(x_i\cos\tau + y_i\sin\tau) + f\cos\sigma}\ , \tag{6.17}$$

and the eigenvectors are

$$\mathbf{w}_1 = \begin{pmatrix} \cos\tau \\ \sin\tau \end{pmatrix} \tag{6.18}$$

$$\mathbf{w}_2 = \begin{pmatrix} -\partial_{\mathbf{s}}y_i \\ \partial_{\mathbf{s}}x_i \end{pmatrix}\ . \tag{6.19}$$

Therefore equation 6.15 can be written as

$$-\left(\partial_{\mathbf{s}}U_i\right)U_i^{-1} = \frac{\sin\sigma}{\sin\sigma(x_i\cos\tau + y_i\sin\tau) + f\cos\sigma}\begin{bmatrix} \cos\tau & -\partial_{\mathbf{s}}y_i \\ \sin\tau & \partial_{\mathbf{s}}x_i \end{bmatrix}$$
$$\begin{bmatrix} 2\partial_{\mathbf{s}}x_i\cos\tau + 2\partial_{\mathbf{s}}y_i\sin\tau & 0 \\ 0 & \partial_{\mathbf{s}}x_i\cos\tau + \partial_{\mathbf{s}}y_i\sin\tau \end{bmatrix}\begin{bmatrix} \cos\tau & -\partial_{\mathbf{s}}y_i \\ \sin\tau & \partial_{\mathbf{s}}x_i \end{bmatrix}^{-1}\ . \tag{6.20}$$

On the other hand, the left hand side of equation 6.20 can be written as a function of the two image plane frequencies considered, $\mathbf{u}'_i = (u'_i, v'_i)$ and $\mathbf{u}''_i = (u''_i, v''_i)$:

$$-\left(\partial_{\mathbf{s}}U_i\right)U_i^{-1} = \frac{1}{u''_iv'_i - u'_iv''_i}\begin{bmatrix} \partial_{\mathbf{s}}u'_iv''_i - \partial_{\mathbf{s}}u''_iv'_i & -\partial_{\mathbf{s}}u'_iu''_i + \partial_{\mathbf{s}}u''_iu'_i \\ \partial_{\mathbf{s}}v'_iv''_i - \partial_{\mathbf{s}}v''_iv'_i & -\partial_{\mathbf{s}}v'_iu''_i + \partial_{\mathbf{s}}v''_iu'_i \end{bmatrix}\ . \tag{6.21}$$

The sum of the eigenvalues of the above expression is

$$\lambda_1 + \lambda_2 = \frac{\partial_{\mathbf{s}}u'_iv''_i - \partial_{\mathbf{s}}u''_iv'_i - \partial_{\mathbf{s}}v'_iu''_i + \partial_{\mathbf{s}}v''_iu'_i}{u''_iv'_i - u'_iv''_i} = -\partial_{\mathbf{s}}\left[\log\left|(u'_iv''_i - u''_iv'_i)\right|\right]\ . \tag{6.22}$$

We now set

$$r = \left|u'_iv''_i - u''_iv'_i\right|\ . \tag{6.23}$$

Alternatively, considering the two frequencies $\mathbf{u}'_i$ and $\mathbf{u}''_i$ as vectors, we could write

$$r = \left|\ \ |\mathbf{u}'_i|\,|\mathbf{u}''_i|\sin(\angle\mathbf{u}''_i - \angle\mathbf{u}'_i)\ \ \right| \tag{6.24}$$

and, considering $\mathbf{u}'_i$ and $\mathbf{u}''_i$ as vectors on the frequency spectrum plane in a 3D space, we could define a vector $\mathbf{r}$ using the outer product

$$\mathbf{r} = \mathbf{u}'_i \wedge \mathbf{u}''_i \ . \tag{6.25}$$

In this case r would be the modulus of the above vector $\mathbf{r}$, i.e. the modulus of the outer products of the frequency vectors:

$$r = |\mathbf{u}'_i \wedge \mathbf{u}''_i| = |\mathbf{r}| \ . \tag{6.26}$$

Equating the sums of eigenvalues of the right and left hand sides of equation 6.20, we get

$$-\partial_{\mathbf{s}} \left[\log(r)\right] = 3\frac{\sin\sigma(\partial_{\mathbf{s}} x_i \cos\tau + \partial_{\mathbf{s}} y_i \sin\tau)}{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma} \ . \tag{6.27}$$

Equation 6.27 is used to solve for the orientation parameters $(\sigma,\ \tau)$ at each point of the image plane. First of all we note that the left hand side is minimal when the direction of differentiation $\angle\mathbf{s}$ is aligned with the direction of the gradient of $\log(r)$, i.e. when the differentiation is done in the direction of the gradient of r (i.e. $\angle\nabla r$). By choosing $\mathbf{s}$ aligned with ∇r, the left hand side of 6.27 takes its minimum value $-|\nabla \log(r)|$.

The minimum value of the right hand side of 6.27 is determined according to the sign of the expression

$$\frac{\sin\sigma}{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma} \ : \tag{6.28}$$

- when 6.28 is negative, the right hand side of 6.27 is minimum for the tilt direction aligned with the direction of differentiation ($\tau = \angle\mathbf{s}$);
- when 6.28 is positive, the right hand side of 6.27 is minimum for the tilt direction opposite to the direction of differentiation ($\tau = \angle\mathbf{s} + \pi$).

In other words, depending on the sign of expression 6.28, the tilt direction is aligned with the direction of the gradient of r ($\tau = \angle\nabla(r)$ when negative), or with the direction opposite to it ($\tau = \angle\nabla(r) + \pi$ when positive).

In our coordinate system the first option is usually the case, so we first assume expression 6.28 has negative sign and we solve for $(\sigma,\ \tau)$. Equating the minima

of the right and left hand sides of equation 6.27 under this assumption, the direction of differentiation $\angle\mathbf{s}$ is the same as the direction of ∇r ($\angle\mathbf{s} = \angle\nabla r$), the tilt direction is aligned with $\mathbf{s}$ ($\tau = \angle\mathbf{s}$) and equation 6.27 becomes

$$-|\nabla(\log r)| = 3\frac{\sin\sigma}{\sin\sigma(x_i\cos\tau + y_i\sin\tau) + f\cos\sigma}\ , \tag{6.29}$$

from which we get the slant σ

$$\tan\sigma = \frac{-|\nabla(\log r)|f}{3 + |\nabla(\log r)|(x_i\cos\tau + y_i\sin\tau)}\ . \tag{6.30}$$

In the other case, i.e. when expression 6.28 is positive, equating the minima of the right and left side of equation 6.27 yields that the direction of differentiation $\angle\mathbf{s}$ is again aligned with the gradient of r ($\angle\mathbf{s} = \angle\nabla r$). However, in this case, the tilt direction is opposite to the direction of $\mathbf{s}$ ($\tau = \angle\mathbf{s} + \pi$), so $\tau = \angle\nabla(r) + \pi$ and equation 6.27 becomes

$$-|\nabla(\log r)| = -3\frac{\sin\sigma}{\sin\sigma(x_i\cos\tau + y_i\sin\tau) + f\cos\sigma}\ , \tag{6.31}$$

from which the corresponding slant σ

$$\tan\sigma = \frac{|\nabla(\log r)|f}{3 - |\nabla(\log r)|(x_i\cos\tau + y_i\sin\tau)}\ . \tag{6.32}$$

In this second case $\angle\mathbf{s}$ is the same, while the tilt is in the opposite direction, i.e. τ is shifted by π radians. This implies that $\cos\tau$, $\sin\tau$ and therefore $(x_i\cos\tau + y_i\sin\tau)$ all change sign. As a result, the sign of the σ computed in 6.32 is opposite to the sign of the σ computed in 6.30. However in our coordinate system, shifting τ by π radians and computing $-\sigma$ means getting the same orientation. The two hypotheses therefore produce the same solution:

$$\tau = \angle(\nabla r) \tag{6.33}$$

$$\sigma = \arctan\frac{-|\nabla(\log r)|f}{3 + |\nabla(\log r)|(x_i\cos\tau + y_i\sin\tau)}\ . \tag{6.34}$$

6.1.2.2 Via integration

Here we show another solution for equation 6.15 which uses integration.

Let us again consider equation 6.27. We note that the numerator of the right hand side is actually the first derivative of the corresponding denominator. Therefore

$$3\frac{\sin\sigma(\partial_{\mathbf{s}}x_i\cos\tau+\partial_{\mathbf{s}}y_i\sin\tau)}{\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma} = 3\partial_{\mathbf{s}}\left[\log(|\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma|)\right] \ , \quad (6.35)$$

and equation 6.27 can therefore be written using the first derivatives:

$$-\partial_{\mathbf{s}}\left[\log(r)\right] = 3\partial_{\mathbf{s}}\left[\log|\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma|\right] \ . \quad (6.36)$$

We now integrate both sides of the above equation 6.36 in the direction of the differentiation vector $\mathbf{s}$, between the value at the image point $\mathbf{x}_i=(x_i,y_i)$ and the value at the nearby image point $(\mathbf{x}_i+\mathbf{s})=(x_i+s\cos\phi, y_i+s\sin\phi)$. $\mathbf{s}$ is therefore the vector representing the increment between the two points: the angle $\phi=\angle\mathbf{s}$ represents the direction of differentiation and integration, the modulus $s=|\mathbf{s}|$ represents the spatial shift, i.e. the distance between the considered image plane coordinate points. The vector $\mathbf{s}$ can therefore be written as

$$\mathbf{s}=|\mathbf{s}|\begin{pmatrix}\cos(\angle\mathbf{s})\\ \sin(\angle\mathbf{s})\end{pmatrix}=\begin{pmatrix}s\cos\phi\\ s\sin\phi\end{pmatrix} \ . \quad (6.37)$$

Below we give the result of integrating 6.36 as described:

$$\sqrt[3]{\frac{r(\mathbf{x}_i)}{r(\mathbf{x}_i+\mathbf{s})}} = \frac{|\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma+\sin\sigma s(\cos\phi\cos\tau+\sin\phi\sin\tau)|}{|\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma|} \ . \quad (6.38)$$

We can easily get rid of the absolute values in the equation above by showing that the numerator and the denominator of the right hand side have the same sign. In fact as $\mathbf{s}$ is an arbitrary increment, we can always choose its modulus s small enough to have the numerator and denominator of the same sign. Hence equation 6.38 is simplified to

$$\sqrt[3]{\frac{r(\mathbf{x}_i)}{r(\mathbf{x}_i+\mathbf{s})}}-1=\frac{\sin\sigma|\mathbf{s}|(\cos\phi\cos\tau+\sin\phi\sin\tau)}{\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma} = \frac{\sin\sigma\ \ \mathbf{s}\cdot\hat{\tau}}{\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma} \ , \quad (6.39)$$

where we have indicated with $\hat{\tau}$ the unitary vector in the direction indicated by τ.

Equation 6.39 can then be solved for the orientation parameters (σ, τ) at each point of the image plane as previously done with equation 6.27.

Let us consider small increments $\mathbf{s}$ (in modulus) and approximate $r(\mathbf{x}_i + \mathbf{s})$ to first order:

$$\sqrt[3]{\frac{r(\mathbf{x}_i)}{r(\mathbf{x}_i + \mathbf{s})}} - 1 = \sqrt[3]{\frac{r(\mathbf{x}_i)}{r(\mathbf{x}_i) + \nabla r \cdot \mathbf{s}}} - 1 \ . \tag{6.40}$$

The above equation 6.40 shows that the left hand side of equation 6.39 is minimal when the direction of differentiation $\angle\mathbf{s}$ is aligned with the direction of the gradient of r, i.e. when the differentiation is done in the direction of the gradient of r ($\angle\mathbf{s} = \angle\nabla r$). We denote the value of the minimum by a_i:

$$a_i = \sqrt[3]{\frac{r(\mathbf{x}_i)}{r(\mathbf{x}_i) + |\nabla r| s}} - 1 \ . \tag{6.41}$$

The minimal value of the right hand side of 6.39 is determined according to the sign of the expression

$$\frac{\sin\sigma}{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma} \ : \tag{6.42}$$

- when 6.42 is negative, the right hand side of 6.39 is minimum for the tilt direction aligned with the direction of differentiation ($\tau = \angle\mathbf{s}$);
- when 6.42 is positive, the right hand side of 6.39 is minimum for the tilt direction opposite to the direction of differentiation ($\tau = \angle\mathbf{s} + \pi$).

In other words, depending on the sign of expression 6.42, the tilt direction is aligned with the direction of the gradient of r ($\tau = \angle\nabla(r)$ when negative), or with the direction opposite to it ($\tau = \angle\nabla(r) + \pi$ when positive).

In our coordinate system the first is usually the case, so we first assume expression 6.42 has negative sign and we solve for (σ, τ). Using this assumption and equating the minima of the right and left hand sides of equation 6.39, the direction of differentiation is the same as the direction of ∇r ($\angle\mathbf{s} = \angle\nabla r$), the tilt direction is aligned with $\mathbf{s}$ ($\tau = \angle\mathbf{s}$) and equation 6.39 becomes

$$a_i = \frac{s\sin\sigma}{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma} \ , \tag{6.43}$$

from which we get the slant σ

$$\sigma = \arctan \frac{a_i f}{s - a_i(x_i \cos\tau + y_i \sin\tau)} \ . \tag{6.44}$$

In the other case, when the expression 6.42 is positive, for equating the minima of the right and left side of equation 6.39, we again choose the direction of differentiation $\angle\mathbf{s}$ aligned with the gradient of r ($\angle\mathbf{s} = \angle\nabla r$), but then the tilt direction is opposite to the direction of $\mathbf{s}$ ($\tau = \angle\mathbf{s} + \pi$). So $\tau = \angle\nabla(r) + \pi$ and equation 6.39 becomes

$$a_i = \frac{-s \sin\sigma}{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f \cos\sigma} \ , \tag{6.45}$$

from which the corresponding slant σ

$$\sigma = \arctan \frac{-a_i f}{s + a_i(x_i \cos\tau + y_i \sin\tau)} \ . \tag{6.46}$$

In this second case the vector $\mathbf{s}$ is the same, while the tilt is in the opposite direction, i.e. τ is shifted by π radians. This implies that $\cos\tau$, $\sin\tau$ and $(x_i \cos\tau + y_i \sin\tau)$ all change sign. As a result, the sign of the σ computed in 6.46 is opposite to the sign of the σ computed in 6.44. However in our coordinate system, shifting τ by π radians and computing $-\sigma$ means getting the same orientation. The two hypotheses therefore produce the same solution:

$$\tau = \angle(\nabla r) \tag{6.47}$$

$$\sigma = \arctan \frac{a_i f}{s - a_i(x_i \cos\tau + y_i \sin\tau)} \ . \tag{6.48}$$

6.1.2.3 Remarks on the solutions

The two methods presented (in sections 6.1.2.1 and 6.1.2.2) for solving equation 6.15 give the same solution for the estimates of the orientation parameters (τ, σ) at each considered image point $\mathbf{x}_i$ (and the same solution is also obtained using the two equivalent methods which we illustrate in appendix A).

Equation 6.15 is written as a function of the variables τ, σ, z_0, f, x_i, y_i, U_i and U_s. U_s is constant on the surface texture, according to the homogeneity assumption of section 2.2. Under a first order approximation, i.e. considering the

variations on the tangent plane at the point $\mathbf{x}_i$, the variables τ,σ,z_0 and f are also constant, as explained in section 6.1.1. We can therefore study the local variations of variables x_i, y_i and U_i, and solve analytically an otherwise ill-posed problem. This again explains why we write equation 6.15 as a function of z_0 instead of z_w: because the latter would change on the considered tangent plane.

It is of particular interest to note that the equations of sections 6.1.2.1 reduce to the solutions found by Massot and Hérault in [121] when we write the equations as functions of a single LSF.

Considering only the LSF $\mathbf{u}'_i = (u'_i, v'_i)$, we redefine r in equation 6.23 by setting $\mathbf{u}''_i = (u''_i, v''_i)$ to the same value in modulus ($|\mathbf{u}''_i| = |\mathbf{u}'_i|$) and to the phase in quadrature ($\angle(\mathbf{u}''_i) = \angle(\mathbf{u}'_i) + \frac{\pi}{2} + k\pi, k \in \mathbb{Z}$) with respect to $\mathbf{u}'_i$ at any point. In other words, we set $\mathbf{u}''_i = (-v'_i, u'_i)$ and re-write equation 6.23 as

$$r = |u'_i v''_i - u''_i v'_i| = |(u'_i)^2 + (v'_i)^2| = \nu_i^2 \ , \tag{6.49}$$

where we have used ν_i to indicate the modulus of $\mathbf{u}'_i$ ($\nu_i = |\mathbf{u}'_i|$).

By substituting for r in equations 6.33 and 6.34 according to equation 6.49, the solutions for τ and σ become

$$\tau = \angle(\nabla \nu_i) \tag{6.50}$$

$$\sigma = \arctan \frac{-2|\nabla(\log \nu_i)|f}{3 + 2|\nabla(\log \nu_i)|(x_i \cos\tau + y_i \sin\tau)} \ . \tag{6.51}$$

Equations 6.50 and 6.51 coincide with the solutions found by Massot and Hérault in [121] (equation 27 in the paper), where ν_i indicates the modulus of the LSF (v_i in the paper) and the few differences in the denominator are due to a rotated viewing geometry.

More interestingly, we can use equation 6.49 to also adapt equations 6.47 and 6.48 (and the equations A.19, A.20, A.36 and A.37 in appendix A) to the case of a single LSF.

Finally, we remark that the gradient of r, used in the given equations to determine the tilt and the slant locally, is proportional to the density gradient $\nabla\rho$, as defined by [52].

In fact, we can express the density gradient on the textured image by using two LSFs $\mathbf{u}'_i = (u'_i, v'_i)$ and $\mathbf{u}''_i = (u''_i, v''_i)$:

$$\nabla\rho \propto \nabla|u'_i v''_i - u''_i v'_i| = \nabla r \ . \tag{6.52}$$

The density gradient was used in early SFT studies by Gibson [59], Aloimonos [1] and Marinos and Blake [119] in the spatial domain. The theory presented here reformulates the density gradient in the spectral domain, providing more robust and applicable solutions for SFT.

6.1.3 Algorithm structure

The algorithm that we propose here does not require any further assumption about the surface texture under analysis, e.g. frontal points or planar textures.

In fact the presented geometric equations (the solutions for $[\tau,\sigma]$ in equations [6.33,6.34], [6.47,6.48], and in equations [A.19,A.20] and [A.36,A.37] in appendix A) only use local frequency variations of pairs of LSFs and stochastic homogeneity to recover the orientation parameters locally. So the method can be used on surfaces of any size and shape, planar and curved, developable and non-developable, small and extended.

Following the orientation estimation at each image pixel, the 3D shape, i.e. z_w, is recovered from the σ's and τ's by using the integration of [50], as discussed in section 2.5.

Our proposed algorithm reconstructs a 3D shape for each pair of LSFs. Then the utilised LSFs are back-projected to the surface texture using equation 6.1 and the determined (σ,τ,z_w). The homogeneity assumption states that the surface texture frequencies $\mathbf{u}_s$ should be nearly the same and that their variance should therefore be approximately zero. Hence we choose the shape associated with the lowest variance, assuming that lower values, closer to the ideal zero value, correspond to better estimates.

We summarize the general proposed structure in algorithm 6.

As already discussed in chapter 2 and in section 5.3.2, we note that using two LSFs does not reduce the applicability of our method, because most real textures have multiple LSFs.

However the over-specification of the systems of equations and the mathematical structure imposed by the eigendecomposition provide robustness to noise, as proved by the results in section 6.2.

Algorithm 6 Structure of the proposed algorithm to recover the 3D shape from multiple estimated LSFs by using the local variations of pairs of LSFs

for each pair of estimated LSFs **do**

- compute the orientation parameters (σ,τ) at each pixel by using the local variations of the LSFs (i.e. r in this chapter, or p in appendix A) and the relevant equations from one of the indicated methods (equations (6.33,6.34) and (6.47,6.48) in this chapter, or equations (A.19,A.20) and (A.36,A.37) in appendix A);
- integrate the orientations at the image pixels with the algorithm of [50] to recover the 3D shape z_w;
- back-project the utilised LSFs to the surface texture by using the computed orientations (σ,τ), the shape (z_w) and equation 6.1
- compute the variance of back-projected frequencies on the surface texture

end for

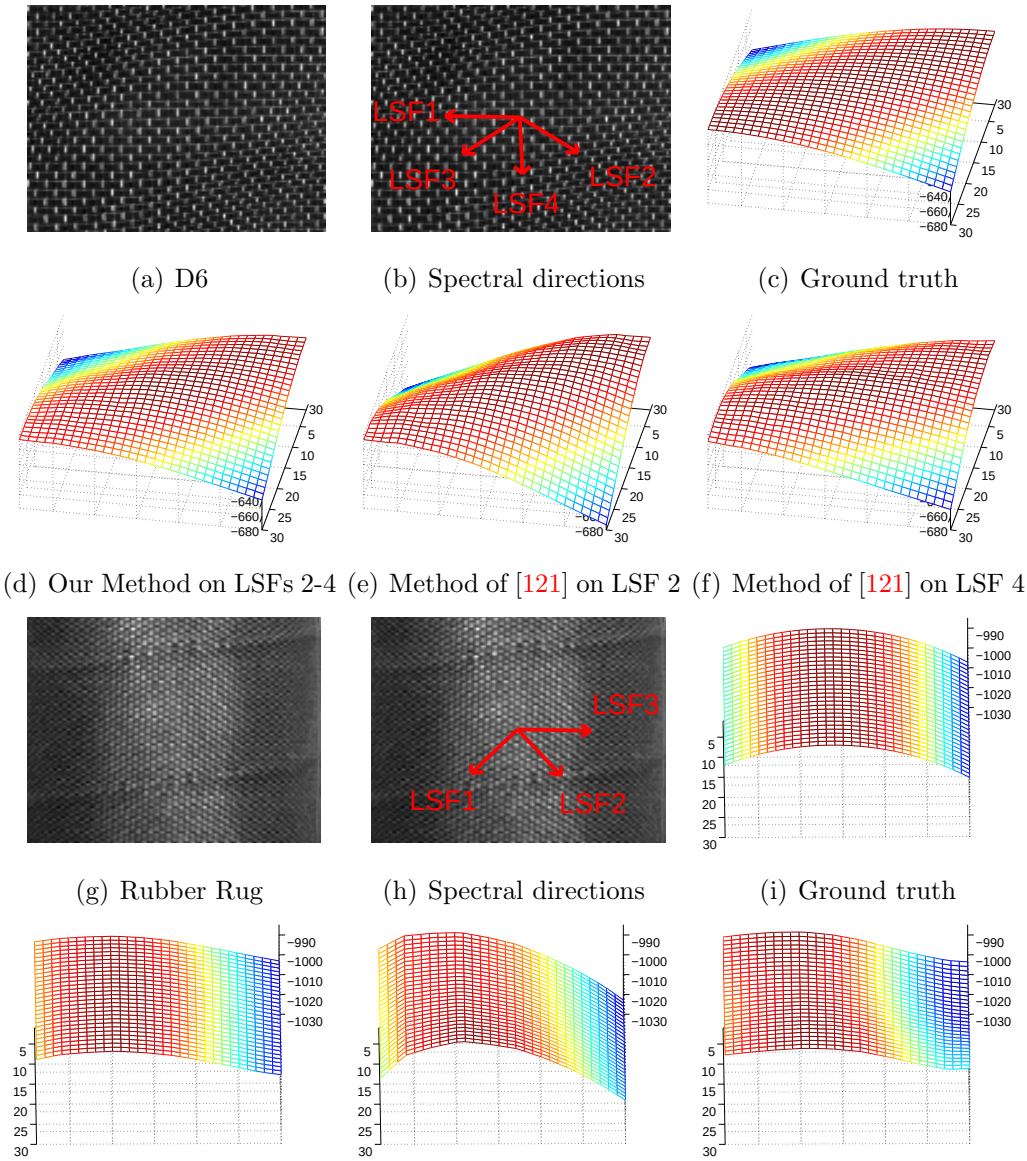
- choose the best shape reconstruction estimation by selecting the pair of LSFs which determines the smallest variance

Furthermore, as for algorithm 6, the local estimates are here achieved at each pixel independently, which prevents any misestimate from being propagated. In addition, the integration of [50] enforces integrability and generally discards single misestimates, which ensures a better robustness.

Accuracy and robustness are shown by the results in section 6.2, where we compare our algorithm with the 3D reconstruction method of [121] and show that the use of two LSFs is essential in dealing with the *tilt direction - spectral direction* problem.

6.2 Results and Comparisons

The proposed algorithm 6 is here applied to the semi-synthetic and real images already considered in sections 5.2.3 and 5.3.3, to an additional set of real images, and to the real video sequence presented in section 5.3.3. Our technique is compared in all cases with the reconstruction algorithm of [121], applied to the same estimated LSFs.



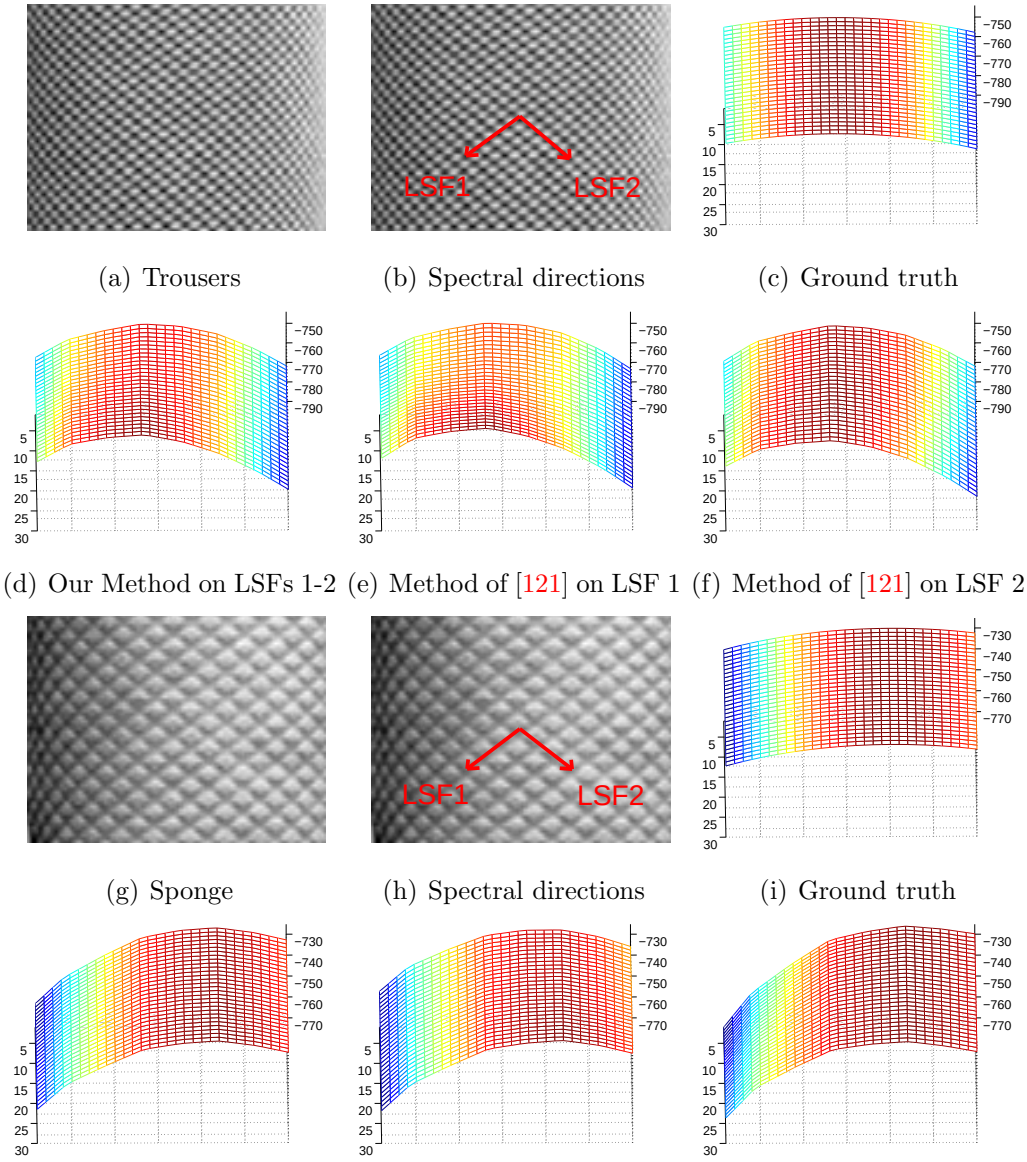
(a) D6 (b) Spectral directions (c) Ground truth

(d) Our Method on LSFs 2-4 (e) Method of [121] on LSF 2 (f) Method of [121] on LSF 4

(g) Rubber Rug (h) Spectral directions (i) Ground truth

(j) Our Method on LSFs 1-3 (k) Method of [121] on LSF 1 (l) Method of [121] on LSF 3

Figure 6.1: (*First row*) 128x128 semi-synthetic image of Brodatz texture D6 on a cosine surface, spectral directions of the selected LSFs and ground truth shape, central 64x64 part, already described in figs. 5.1 and 5.2(a). (*Second row*) Perspective reconstructions (same central part) of shape from image (a) using our method and the method of [121] on the indicated LSFs. (*Third row*) 180x180 real image of a rubber rug laid on a cylinder, spectral directions of the selected LSFs and ground truth shape, central 90x90 part, already described in figs. 5.4 and 5.5(a). (*Fourth row*) Perspective reconstructions (same central part) of shape from image (g) using our method and the method of [121] on the indicated LSFs



(a) Trousers (b) Spectral directions (c) Ground truth

(d) Our Method on LSFs 1-2 (e) Method of [121] on LSF 1 (f) Method of [121] on LSF 2

(g) Sponge (h) Spectral directions (i) Ground truth

(j) Our Method on LSFs 1-2 (k) Method of [121] on LSF 1 (l) Method of [121] on LSF 2

Figure 6.2: (*First row*) 128x128 real image of a pair of trousers laid on a cylinder, spectral directions of the selected LSFs and ground truth shape, central 64x64 part ((a) and (c) presented in figs. 5.7 and 5.8(a)). (*Second row*) Perspective reconstructions (same central part) of shape from image (a) using our method and the method of [121] on the indicated LSFs. (*Third row*) 128x128 real image of a sponge laid on a cylinder, spectral directions of the selected LSFs and ground truth shape, central 64x64 part ((g) and (i) presented in figs. 5.9 and 5.10(a)). (*Fourth row*) Perspective reconstructions (same central part) of shape from image (g) using our method and the method of [121] on the indicated LSFs

Figures 6.1 and 6.2 illustrate the semi-synthetic and real images of figures 5.1, 5.4, 5.7, 5.9 and 5.11. In particular we re-present here the images, the spectral direction of the selected LSFs at the central pixel and the ground truth shape of the 3D surface underlying the texture.

Figures 6.1 and 6.2 then present the best reconstructions achieved using two of the available LSFs and our algorithm 6, as well as the reconstructions obtained with the same two LSFs considered separately and the reconstruction algorithm of [121].

In fact, the work from Massot and Hérault [121] is the most recent significant contribution to SFT, under the same assumption of stochastic homogeneity. We were inspired by the work in developing the 3D reconstruction algorithm 6 and we feel that the work in [121] is the most meaningful comparison for our proposed technique.

Furthermore, as for our algorithm, the 3D reconstruction method of [121] does not require any assumption about the underlying shape of the considered texture, and is therefore applicable to any surface of any size, planar and curved, developable and non-developable, small and extended.

However, differently from our algorithm, [121] only considers one LSF, which is why we apply their algorithm to both the LSFs that we consider.

In addition, we introduce two other real images, extracted, as previously, from pictures acquired with a Pulnix TM-6EX camera, at a resolution of 640x480 with 256 grey levels.

The first image, in figure 6.3, represents a pillowcase photographed on a cylinder of known diameter (so as to retain the ground truth), arranged in front of the camera.

Figure 6.3 presents: (a) the entire picture from which the 240x240 central image was extracted, (b) the image, (c) its spectrum with labels indicating the two selected LSFs, (d) the spectral direction of the two LSFs at the central pixel, (e) the ground truth shape of the cylinder beneath the pillowcase, (f) the reconstructed surface achieved with our algorithm on the two LSFs, (g), (h) the reconstructions using the algorithm of [121] on the two LSFs separately.

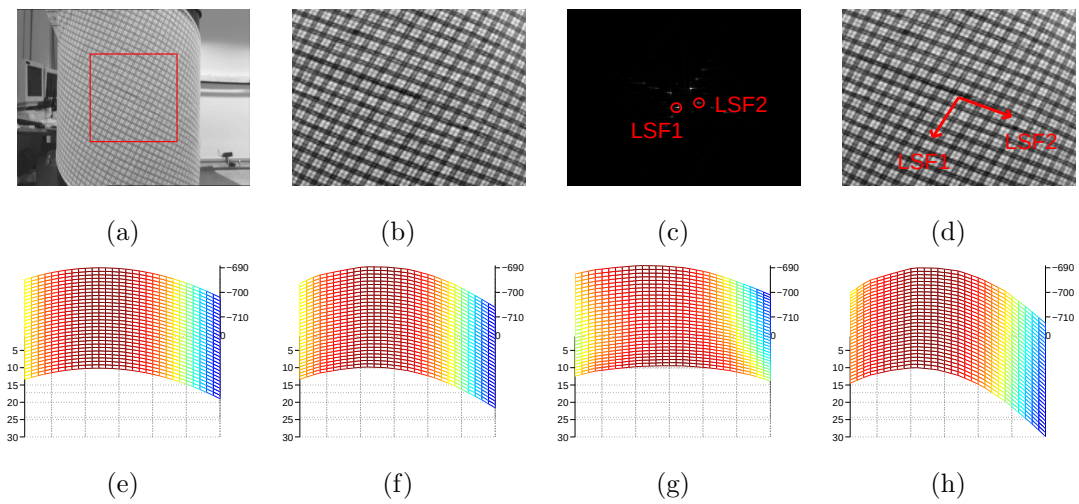


Figure 6.3: (a) 640x480 real picture of a pillowcase laid on a cylinder; (b) 240x240 real image extracted from (a); (c) spectrum of (b) with labels indicating the selected LSFs; (d) arrows indicating the respective spectral directions of the selected LSFs at the central pixel superposed on (b); (e) ground truth shape of the cylinder beneath the pillowcase in (b), central 120x120 part; (f,g,h) perspective reconstructions (same central part) of shape from image (b) using: our method on LSFs 1 and 2, the method of [121] on LSF 1, the method of [121] on LSF 2

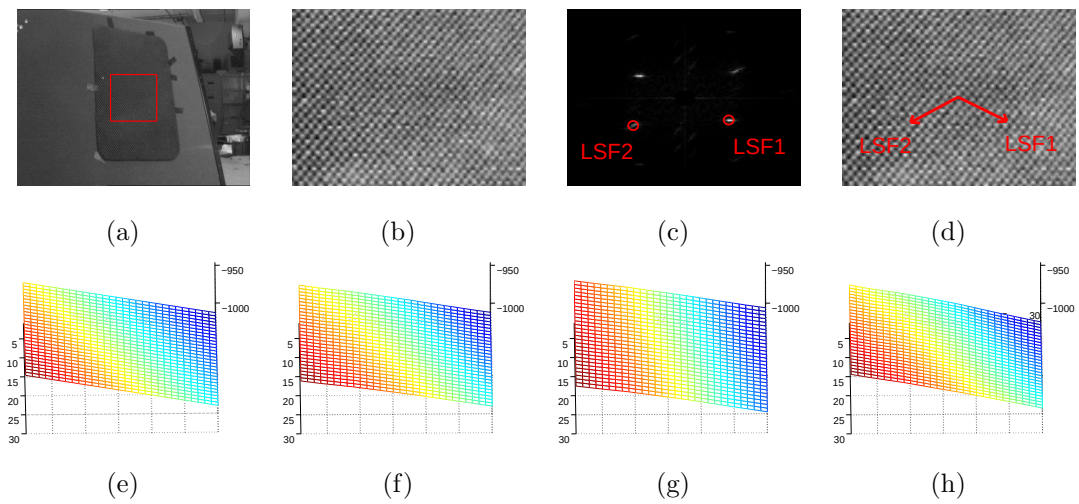


Figure 6.4: (a) 640x480 real picture of a rubber rug laid on a planar surface; (b) 128x128 real image extracted from (a); (c) spectrum of (b) with labels indicating the selected LSFs; (d) arrows indicating the respective spectral directions of the selected LSFs at the central pixel superposed on (b); (e) ground truth shape of the plane beneath the rubber rug in (b), central 64x64 part; (f,g,h) perspective reconstructions (same central part) of shape from image (b) using: our method on LSFs 1 and 2, the method of [121] on LSF 1, the method of [121] on LSF 2

The second image, already analysed by our planar algorithm 2 in figure 4.3(a), represents the same rubber rug as in figure 6.1(g), laid on a planar surface of known pose.

Again figure 6.4 illustrates the image, its spectrum with two LSFs labeled, their spectral directions, the ground truth planar shape, and then the reconstruction achieved with our algorithm on the two LSFs and the reconstructions using the algorithm of [121] on the two LSFs separately.

Across these examples, our reconstructions correspond well to the ground truth, notwithstanding the noise from reflections, imperfections, variations in illumination and self-shadowing, as already noted in sections 5.2.3 and 5.3.3.

Furthermore our reconstructions look more accurate than the reconstructions obtained by using the 3D reconstruction algorithm of [121], as also confirmed by table 6.1, representing the average of the errors in the tilt direction τ and in the slant σ estimation at the image pixels.

The results, both visually and numerically, may also be compared with the results in sections 5.2.3 and 5.3.3. In particular algorithm 6 achieves comparable results on both the tilt and slant estimation. However, some errors in the slant estimation, e.g. the trousers 6.2(a) and sponge 6.2(g) images, appear to be significantly higher.

We explain the bigger errors in those cases with the approximation used to solve for the local orientation parameters. In fact, in order to make the ill-posed problem a tractable one, and to find a local closed-form solution, both the Jacobian (equation 5.43) and the further differentiation (equation 6.15) are done on the tangent plane at the considered point, i.e. using a first order approximation.

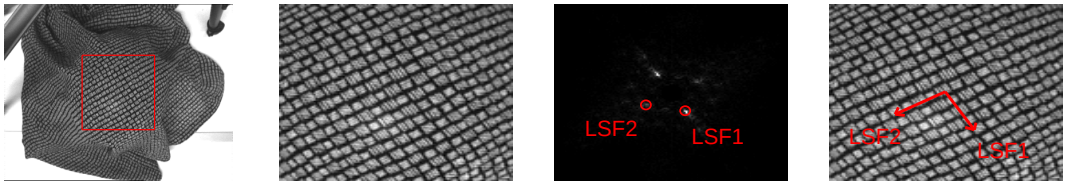
The approximation, used often in the relevant literature (e.g. [158], [42], [121]), has the drawback of not considering the two curvatures and the geodesic torsion (see [52] for more details).

As a consequence, we observed that the slant is estimated with a systematic error, which is a function of its local value and of its local variation.

For this reason the images of the pair of trousers and of a sponge present a bigger error, because in those cases the image includes nearly the steepest sides of the cylinders and because the cylinders are smaller and the variations of slant are bigger at each pixel.

Image	**Method**	**Average error**	
		$\lvert\epsilon_\tau\rvert$	$\lvert\epsilon_\sigma\rvert$
D6	Our method on LSFs 2 and 4	1.86	8.42
	Method of [121] on LSF 2	1.41	18.17
	Method of [121] on LSF 4	10.51	12.25
Rubber Rug	Our method on LSFs 1 and 3	4.24	6.12
	Method of [121] on LSF 1	7.89	12.87
	Method of [121] on LSF 3	15.83	8.10
Trousers	Our method on LSFs 1 and 2	2.66	19.02
	Method of [121] on LSF 1	3.85	23.55
	Method of [121] on LSF 2	5.56	22.05
Sponge	Our method on LSFs 1 and 2	4.20	13.00
	Method of [121] on LSF 1	4.80	16.95
	Method of [121] on LSF 2	7.55	17.70
Pillowcase	Our method on LSFs 1 and 2	2.18	6.60
	Method of [121] on LSF 1	12.52	1.82
	Method of [121] on LSF 2	4.03	13.03
Rubber Rug on Plane	Our method on LSFs 1 and 2	2.26	2.55
	Method of [121] on LSF 1	16.52	4.73
	Method of [121] on LSF 2	3.84	7.13

Table 6.1: Average estimation errors (in degrees) for σ and τ of the reconstructions illustrated in figs. 6.1, 6.2, 6.3, 6.4



(a) (b) (c) (d)

Figure 6.5: (a) Frame 1 extracted from a 400x300 real video sequence of a textured cloth laid on a deflating balloon; (b) 128x128 real image extracted from (a); (c) spectrum of (b) with labels indicating the two selected LSFs;(d) arrows indicating the respective spectral directions of the two LSFs at the central pixel superposed on (b)

However, the shape estimate is in any case qualitatively good, as the systematic error only rescales the σ estimates, and the cylinder remains a seemingly cylindrical shape, with steeper sides.

Furthermore, shapes with smaller variations of local slant are much less affected by the error and present accurate results, both qualitatively and numerically, as observed for the pillowcase (figure 6.3, where the texture is laid on a bigger cylinder) and for the rubber rug on the plane (figure 6.4, in fact planar cases are never affected).

Another meaningful comparison of our 3D reconstruction algorithm with that from [121] is done on the real video sequence of a textured cloth laid on a deflating balloon, as introduced in figure 5.12.

Figure 6.5 re-illustrates the image extracted from the first frame of the sequence, its spectrum with labels indicating the two selected LSFs, and the spectral direction of the LSFs.

Figure 6.6 presents four frames from the sequence (1,100,200,300), together with the respective 3D reconstructions achieved with our method using LSFs 1 and 2, and with the method of [121] on the two LSFs separately.

As already argued in section 5.3.3, the images from the video sequence are particularly challenging for a number of reasons: (i) the image quality is poor; (ii) the textured cloth is stretched to fit the spherical balloon and it creases due to the deflation; (iii) the tilt and slant take in the image a wide range of values.

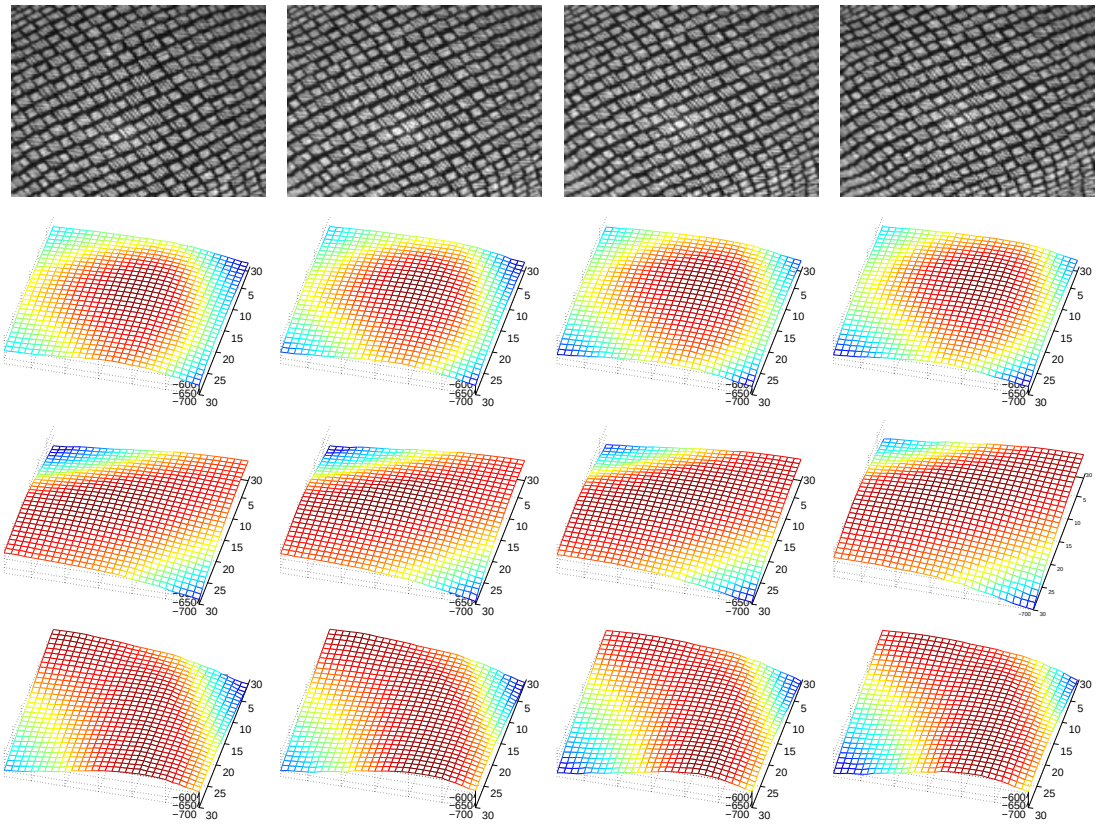
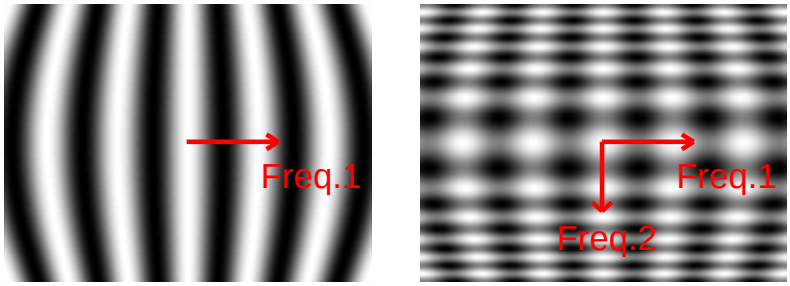


Figure 6.6: 128x128 real images extracted from the frames of a video sequence of a textured cloth laid on a deflating balloon. (*first row*) frames 1,100,200,300; (*second row*) perspective reconstructions with our method using LSFs 1 and 2; (*third row*) perspective reconstructions with the method of [121] using LSF 1; (*fourth row*) perspective reconstructions with the method of [121] using LSF 2



(a) (b)

Figure 6.7: (a) Cosine textured with a LSF (its spectral direction represented by the red arrow) orthogonal to the tilt direction; (b) cosine textured with a LSF orthogonal to the tilt direction (same as (a)) and a second LSF directed parallel to the tilt direction (the spectral directions of the LSFs are represented with arrows)

Despite the expected difficulties, the reconstructions with our method correspond well to the observed images. Also, although the frames were processed independently, the changes between close frames are consistent and the reconstructed 3D video is smooth and describes well the slow deflation of the balloon in the video sequence.

Compared with the reconstructions obtained with algorithm 5 shown in figure 5.13, algorithm 6 achieves equally good results and the reconstructed video is even smoother, while the frontal point extraction is here not necessary and the reconstruction is achieved locally.

On the other hand, figure 6.6 shows how the reconstructions obtained with the algorithm of [121] do not describe the corresponding images well.

In particular, those reconstructions seem to be heavily biased by the spectral direction of the LSF used, in the sense that the shape seems to be well reconstructed only for depth variations along the particular spectral direction.

Figure 6.7 illustrates our explanation for the failure of the reconstruction method of [121].

When a tilt direction is orthogonal to the spectral direction of a LSF, as in figure 6.7(a), the foreshortening effect is null and one would struggle to recognize

the underlying shape by only considering the left perspective effects of scaling and position.

These perspective effects correspond to the change in size due to distance and to the convergence of parallel lines in the world to vanishing points (or equivalently of parallel planes to vanishing lines), as already described in section 2.4. As also maintained by [110], perspective effects are generally negligible for most of the objects that we might want to consider for SFT.

However, adding a second LSF, parallel or at least not orthogonal to the tilt direction of the shape, creates a foreshortening effect and makes the structure pop up from the image, as in figure 6.7(b).

The reconstruction algorithm of [121], in agreement with this explanation, describes well the pixels where the spectral direction of the considered LSF is parallel to the tilt direction, as there is some foreshortening effect. However the algorithm fails on the pixels where the two directions are orthogonal, as the left perspective effects are insufficient to reconstruct the 3D shape in a real noisy image. Finally their algorithm shows increasing errors inbetween the two cases.

The same kind of bias towards the direction of the considered LSF can be found in previous figures 6.1, 6.2, 6.3, 6.4 (particularly evident in figures 6.1(f), 6.1(k), 6.2(f), 6.2(l), 6.3(g), 6.3(h), 6.4(g)). Also, the same explanation accounts partially for the failure of algorithm 3, only considering a single LSF at the time, on the same video sequence, as illustrated in figure 5.14.

By contrast, algorithm 6 does not fail, because it uses two LSFs, weighting at each pixel their contribution to the orientation estimate according to the alinement of each LSF with the tilt direction. If the two LSFs have distinct spectral directions, as is more likely to happen, our algorithm can deal with all the possible tilt orientations.

Moreover, when the tilt direction of the underlying shape is inbetween the two considered LSFs, our algorithm can count on an effective redundancy and cope with higher levels of noise. In fact our algorithm robustly reconstructs every frame of the real video sequence, despite noise, creases and stretches.

Finally, it is of interest to note that a similar conclusion is maintained by Li and Zaidi. Explaining their psychophysical experiments ([100, 101, 102, 103,

104, 105]), the authors argue that a single frequency is insufficient for conveying complex three-dimensional shapes.

On the other hand, it is not essential to have a spectral energy component in the tilt direction, as also maintained by Massot and Hérault ([120, 121]). Future investigations will address the cases of natural textures with diffuse spectra, as we explain in section 8.3.

6.3 Final Remarks

This chapter has presented a novel algorithm which uses a multi-scale distortion based on LSFs to recover the 3D coordinates of textured images.

Our algorithm utilises the local frequency variations of each pair of LSFs to locally recover the orientation estimates at each pixel via eigendecomposition, and then selects the pair of LSFs which gives the best reconstruction.

Our algorithm does not require any other assumptions apart from 'stochastic' homogeneity, it uses general perspective equations and recovers the orientation estimates locally. In other words, the presented algorithm does not require a frontal point, nor a point of known orientation, and is applicable to any shape of any size (planar and curved, developable and non-developable, small and extended).

Our proposed technique has been tested on semi-synthetic and real images. We have been shown it to be accurate, up to the state of the art of SFT, and robust against variations in illumination, self-shadowing, reflections and imperfections in the texture pattern.

In particular, the accuracy of this algorithm has been discussed in comparison with a previously presented algorithm based on SVD in chapter 5 (algorithm 5), and performance differences have been discussed and explained. Moreover, ways to improve the algorithm in this respect are under consideration, as we explain in chapter 8.

We have discussed the accuracy and robustness of our technique in comparison with the 3D reconstruction algorithm of [121], from which we drew inspiration. Our method achieves better results and does not suffer from the described *tilt direction - spectral direction* problem. This results in higher robustness and, above all, in a considerable extension of its applicability.

On a concluding note, the sensitivity of our algorithm to small changes, as in the video sequence example, opens the way to use laid-on textures as a non-invasive way of studying shape changes of objects, while the local orientation estimation enables us to consider complex shapes.

Chapter 7

Application to Texture Mapping and Replacement

This chapter presents the application of the estimated multi-scale distortion and of the geometric relations between surface texture and image plane to the *Computer Graphics* (CG) areas of *texture mapping* and *texture replacement*.

Firstly, introductions to texture mapping and replacement give the necessary background to our original techniques, explaining the key contributions to the field and defining the main problems being faced.

We then propose an original algorithm for texture mapping, which uses the relationship between surface texture and image plane coordinate systems to synthesize new textures and to re-render them onto the estimated shape, i.e. replace the texture.

Finally, we propose another original algorithm to replace the texture in an image by using the estimated LSFs, without reconstructing the 3D shape.

Since the implementation of complete algorithms for *texturing* and *re-texturing* (as texture mapping and replacement are also called) is beyond the scope of this thesis, a comparison with existing techniques is difficult. However, the strengths of our new methods are presented in context, results are shown, and prospects for future investigations are proposed.

7.1 Introduction to Texture Mapping and Replacement

In 1973 the book by Newman and Sproull ([125]) gave a comprehensive treatment of CG. Less than ten years later, in 1982, one of the best books on the subject, [47] by Foley and Dam, could offer a broad overview of CG, but the treatment was not as comprehensive as in [125], because CG had expanded so much.

Today, it is certainly difficult to describe the state of the art of *texture mapping*, one of the many research areas of CG. The technique, which was born in 1974, is used to give realism to rendered objects by applying textures. The wonderful variety of current results and the great number of algorithms proposed would alone require a dissertation. In the following we focus on the main contributions and define the main problems still under consideration.

On another note, the research area of *texture replacement* is more recent, with relevant work starting to appear only in the new millennium. Texture replacement has opened the way to editing simple textures in real pictures and movies for retexturing the original object with novel and attractive motifs. Being a younger research area, it is possible to give a better overview of the relevant work.

7.1.1 Introduction to texture mapping

Texture mapping was first introduced into computer graphics in the mid-seventies by Catmull ([25, 26]). The technique increases the realism of rendered objects significantly by applying high-quality visual details to the synthesised objects using textures. These details are often too complex to be directly applied via geometric modeling, whereas mapping realistic textures greatly improves the quality of graphic renderings at limited additional computational cost.

Before texturing, raster images were smooth shapes, because processors could not afford the computation necessary to render realistic 3D details, e.g. scratches or more detailed effects like fur. Nowadays, computers are immensely more powerful than they were in 1974, but texturing still remains an important issue for the animation studios. In fact, rather than complicating the 3D geometric modeling,

which is limited by computer power, recent studies seem to be focusing on higher quality texturing, e.g. 3D texturing, which we introduce later in this section.

Before proceeding with the description of texture mapping, we define the meaning of the term 'texture' in this field. Texture is not only used to describe a pattern with some properties, as seen in chapter 1. Here it is also used to simply indicate a general image that we want to map onto an object.

We also note that, in the literature, the texture that we want to apply is labeled with coordinates (u,v), the object on which the texture has to be mapped is labeled (x_0, y_0, z_0), and the screen image coordinates are given by (x,y). However, in the following sections (but not in this introduction) we will try to be consistent with the notation that we have used so far, which we described in section 2.3.

Following the approach of Heckbert in his 'Survey of Texture Mapping' ([68]), texture mapping can be split into two elements: *geometric mapping* and *filtering*. The former involves computing the warping functions which relate the texture image, the object surface and the screen image. The latter involves re-sampling the texture image on the screen to avoid aliasing.

Geometric mapping usually involves two phases: first we have the surface parametrization which maps the 2D texture coordinates (u,v) to the 3D object coordinates (x_0, y_0, z_0), then we have the viewing transformation, which maps the 3D object coordinates to the screen coordinates (x,y), usually with a perspective projection. However it is often the case that the texture coordinates are mapped directly onto the screen coordinates.

Surface parametrization can be done by scanning the screen coordinates and determining the functions that relate them to the texture coordinates (normally indicated with $u(x, y)$ and $v(x, y)$). Or it can be done by scanning the texture coordinates (therefore computing $x(u, v)$ and $y(u, v)$). The first method is usually preferred to the second because it is less likely to leave holes in the synthesized image.

A parameterization is clearly necessary to map 2D textures to 3D objects. This can be done with texture coordinates (u, v) as in standard texture mapping ([8]), with light rays ([12]) or normal vectors ([123]) as in illumination mapping, or with spatial coordinates (x_0, y_0, z_0), as is required when we want to be able to carve the object (see later for a description of *solid texturing*).

The geometric mapping of textures onto 3D objects and then into screen coordinates is often based on mapping polygons, either in orthographic or in perspective projections ([4, 67, 144]). [24, 129, 152] introduce a 2-pass method which allows one to compute the relations between screen and texture coordinates $x(u, v)$ and $y(u, v)$ in scanline order along x and y. [17, 133, 171] study the linear and non-linear coordinate transformations which are are used to formulate and speed up the previous relationships.

More recently *3D texturing* has been proposed to deal with limitations inherent in the conventional 2D geometric mapping. Dischler and Ghazanfarpour define two major problems ([40]):

- mapping a 2D texture to a 3D unfoldable surface implies distortions, e.g. at the poles of a sphere. In addition to this, a 2D texture mapping cannot guarantee texture continuity, e.g. between two faces sharing an edge. The study of this limitation has lead to the introduction of *solid texturing* (by Perlin and Peachey independently in [130, 132]), which we also find in [55, 56]. Solid texturing defines the texture in 3D so that it is actually possible to carve it on the object. The cited works empirically construct a 3D texture to apply, e.g. wood, granite, marble, clouds. [41, 58, 69] try to analytically characterize a texture using one or multiple views of the desired texture and some spectral analysis.

- the lack of a third dimension on the 2D texture means bumps on the object surface cannot be simulated. *Bump mapping* was introduced by Blinn in [11] to circumvent this problem by 'perturbating' the surface normals before applying the texture, and was followed by an actual displacement mapping in [32, 122], where points are moved along the surface normals. To deal with more complex textures like cotton and fur, some techniques known as *geometric texturing* were introduced, among which are *hypertextures* (e.g. [131]), consisting on perturbating the texture density volume, and *texel maps* (e.g. [79]), given by applying 3D texture elements, e.g. parallelepipeds, on the object surface.

The second main element of texture mapping is *filtering*. This addresses resampling the texture on the screen grid once the texture geometric mapping is

known. The simplest approach consists of choosing the nearest neighbor pixel, while more complex methods which use bilinear and bicubic interpolations have also been proposed. One of the main problems to be solved is aliasing, most noticeable with high frequency textures ([35, 173]). The texture can either be sampled at higher resolution (e.g. [33, 97]), with the drawback that the resolution can be unpredictably high, or it can be low pass filtered (e.g. [153]), so as to remove the high frequencies causing the aliasing before the re-sampling process.

In section 7.2 we propose a texture mapping algorithm, which is an application of the geometric framework presented in section 5.3.1.

We first estimate the first partial derivatives of the geometric mapping function, and then we recover the function itself by integrating the derivatives with the method of [50]. The geometric mapping function, indicated in the literature as $[u(x, y), v(x, y)]$, is here written as $\mathbf{x}_s(\mathbf{x}_i) = [x_s(x_i, y_i), y_s(x_i, y_i)]$.

The method that we propose applies to general shapes (planar and curved, developable and non-developable). However it is not directly comparable with other existing texturing algorithms, since here we only implement ideas for the geometric mapping and not the filtering.

7.1.2 Introduction to texture replacement

Texture replacement is a 'young' investigation area. The pioneering work of Tsin et al. ([163]) dates back to 2001, while important contributions have also been made in recent years ([45, 46, 63, 83, 108, 177]).

Re-texturing is therefore an open field for research. It is in fact an active area, with both researchers and companies (e.g. in the filming industry) starting to investigate how to apply complex textures in place of simple motifs (e.g. substituting textures in post-production).

As already mentioned, the first important contribution is due to Tsin et al. ([163]). They build on the methods of [44] and [166] to estimate the texture pattern by using neighbouring pixels with Markov Random Fields and look-up tables and trees. In particular, because such processes are time consuming, they assume a near regular texture on a planar surface and consider *texture tiles*, e.g. bricks, carpet textures etc., which are mapped with the registration technique

of [165]. The method of [163] requires user interaction to select a uniformly illuminated region, where a standard technique or another manual input is used to extract a tile.

The user interaction required in [127] is extensive, as one has to manually segment the layers in the image, assign a depth to each of them, and draw the hidden parts. They introduce a non-distorted 'clone brushing' for cloning parts of the image according to the depth, and an algorithm for decoupling texture and illumination from the image. Impressive new pictures from different viewpoints and with changed illumination are therefore produced at the price of extensive manual input.

Visually pleasing effects of texture replacements are obtained by using the 'Textureshop' of [45]. A simplified Shape-From-Shading (SFS) technique ([70]), under the assumptions of Lambertian reflectance and single light source, is used to recover an approximate height field. After a possible manual adjustment of normals, the new texture is synthesized on the height field while the graphcut texturing algorithm of [95] determines the optimal seam through the overlapping area. The algorithm has the drawbacks that the procedure has to be repeated when applying a different texture, graphcut textures are slow to synthesise, and the object has to be selected manually.

'Textureshop' is improved on by [177] and [46]. The former uses 'lazy snapping' ([106]) to reduce user interaction in the object selection. It also extends the SFS to textured objects by using a Gaussian mixture model ([146]): colour clusters are determined and the normals from the most reliable colour cluster are propagated to other clusters. The latter introduce 'RotoTexture', to extend 'Textureshop' to deal with both texture synthesis and mapping in the case of videos.

[108] uses regular tiling ([61]) of textures as an anchor point to define multi-modal deformation fields (geometry, lighting, colour) with the aid of user interaction. Once the deformation fields are computed, regular and near-regular textures are synthesized and manipulated through them so as to create the same visual effect (same illumination and viewpoint) using a different texture.

On another note, [83] shows that the human visual system makes some assumptions and requires that certain aspects of the interaction between light and

matter are accurately reproduced, e.g. the environment used to light the object has to have the correct range of colours and intensities. However even enormous inconsistencies can go un-noticed by observers, e.g. the position of reflections and refractions from the object. Aiming to substitute the material in a single high dynamic range photograph without a physically accurate 3D reconstruction, they estimate the approximate depth field and relight the object with complex techniques giving the compelling illusion of a new material.

Most recently, [63] contributes to retexturing. As already noted by Fang and Hart, Guo et al. [63] again underline that the two major problems in retexturing, geometry and illumination recovery, are impractical to achieve from a single photograph. However they base their work on the fact that full recovery of 3D geometry and lighting conditions can be relaxed in practice. With the purpose of imitating the texture distortion of the original image without computing the 3D geometry, they formulate texture distortion as a stretch-based parameterization, and transfer the lighting effect afterwards. Once the texture mapping is complete, any picture can be applied by using the parameterization, and the visual effects are pleasing even for videos.

The retexturing algorithm that we introduce in section 7.3 does not require user interaction and, once the geometric mapping is estimated, allows the application of any other texture. Furthermore it is based on LSFs, and is therefore not affected by small occlusions or changes in illumination, shading and contrast. However, issues concerning the object selection and re-lighting are not taken into account but indicated for future work.

Being based on multi-scale frequency estimations, our method differs completely from the illustrated works, mostly relying on SFS.

7.2 Texture Mapping

Here we propose an original algorithm for texture mapping, based on the geometric equations described in section 5.3.1. The algorithm can be used to generate images of textured objects and to apply a different texture onto a reconstructed shape, possibly after modifying and changing the viewpoint of the shape.

In particular, this section is centred on the problem of determining the geometric mapping, i.e. the vector function which relates the image plane ($\mathbf{x}_i$) and the surface texture ($\mathbf{x}_s$) coordinates.

Let us recall the revisited viewing geometry and projection model defined in section 5.3.1. They comprise:

- the viewing geometry of [158]: world coordinates $\mathbf{x}_w = (x_w, y_w, z_w)$, image coordinates $\mathbf{x}_i = (x_i, y_i)$, surface texture coordinates $\mathbf{x}_s = (x_s, y_s)$ and orientation parameters σ,τ;
- with an integrated counter-rotation as indicated by [42], so as to have the $\mathbf{x}_s$ coordinate system consistent on a virtually unfolded surface texture.

According to these, the equations which relate the surface texture and the image plane coordinates are

$$\mathbf{x}_s = \frac{z_w}{f} R(\tau) P^{-1}(\sigma) R(-\tau) \mathbf{x}_i \ , \tag{7.1}$$

$$\text{where} \quad z_w = \frac{z_0 f \cos\sigma}{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma} \tag{7.2}$$

in the perspective case, and

$$\mathbf{x}_s = R(\tau) P^{-1}(\sigma) R(-\tau) \mathbf{x}_i \tag{7.3}$$

in the orthographic case, having defined

$$R(\tau) = \begin{bmatrix} \cos\tau & -\sin\tau \\ \sin\tau & \cos\tau \end{bmatrix} \quad \text{and} \quad P(\sigma) = \begin{bmatrix} \cos\sigma & 0 \\ 0 & 1 \end{bmatrix} . \tag{7.4}$$

As explained in section 5.3.1 $R(\tau)$ and $P(\sigma)$ are respectively responsible for rotating the system of coordinates of τ and foreshortening the distances parallel to the tilt direction by a coefficient $\cos\sigma$, while z_0 is the z_w coordinate of the point at the intersection of the tangent plane at the considered point $\mathbf{x}_i$ and the optical axis.

Equations 7.1 and 7.3 express the *geometric mapping* functions $\mathbf{x}_s(\mathbf{x}_i)$ between the surface texture and image coordinates *when the surface texture is planar*.

When the shape is not planar but smooth (enough to allow a reasonable first order approximation), equations 7.1 and 7.3 still apply locally, as seen in 5.3.1, on the tangent plane passing through the given surface texture point.

In this case the geometric mapping can be expressed by using the first derivatives. For the perspective case we have

$$\partial_{x_i}\mathbf{x}_s(\mathbf{x}_i) = \partial_{x_i}\left[\frac{z_w}{f}R(\tau)P^{-1}(\sigma)R(-\tau)\mathbf{x}_i\right] \tag{7.5}$$

$$\partial_{y_i}\mathbf{x}_s(\mathbf{x}_i) = \partial_{y_i}\left[\frac{z_w}{f}R(\tau)P^{-1}(\sigma)R(-\tau)\mathbf{x}_i\right] , \tag{7.6}$$

and for the orthographic case we have

$$\partial_{x_i}\mathbf{x}_s(\mathbf{x}_i) = \partial_{x_i}\left[R(\tau)P^{-1}(\sigma)R(-\tau)\mathbf{x}_i\right] \tag{7.7}$$

$$\partial_{y_i}\mathbf{x}_s(\mathbf{x}_i) = \partial_{y_i}\left[R(\tau)P^{-1}(\sigma)R(-\tau)\mathbf{x}_i\right] . \tag{7.8}$$

Both sets of equations stand for all image points $\mathbf{x}_i$, and relate the value of the derivatives ($\partial_{x_i}\mathbf{x}_s(\mathbf{x}_i)$ and $\partial_{y_i}\mathbf{x}_s(\mathbf{x}_i)$) to the local shape parameters τ, σ, z_w (in particular we emphasise that z_w represents the height field and therefore the shape itself).

The derivatives of equations (7.5, 7.6) and (7.7,7.8) have already been derived in equations 5.45 and 5.47. In fact the equations coincide with the transpose Jacobian of the vector function $\mathbf{x}_s(\mathbf{x}_i)$:

$$J^t(\mathbf{x}_s, \mathbf{x}_i) = \begin{bmatrix} \partial_{x_i}x_s & \partial_{x_i}y_s \\ \partial_{y_i}x_s & \partial_{y_i}y_s \end{bmatrix} . \tag{7.9}$$

It is interesting to note that the local relations between the surface texture $\mathbf{x}_s$ and the image plane system coordinates $\mathbf{x}_i$ are, to a first order approximation, the same as the relations between the frequencies on the surface texture ($\mathbf{u}_s$) and the image plane ($\mathbf{u}_i$). This property is used in the following section 7.3.

We re-write below the analytical form for the Jacobian from equations 5.45 and 5.47:

$$J^t(\mathbf{x}_s, \mathbf{x}_i) = \frac{z_w}{f}\left\{-\frac{\sin\sigma\begin{bmatrix} x_i\cos\tau & y_i\cos\tau \\ x_i\sin\tau & y_i\sin\tau \end{bmatrix}}{\sin\sigma(x_i\cos\tau + y_i\sin\tau) + f\cos\sigma} + I\right\}R(\tau)P^{-1}(\sigma)R(-\tau) \tag{7.10}$$

in the perspective case, and

$$J^t(\mathbf{x}_s, \mathbf{x}_i) = R(\tau)P^{-1}(\sigma)R(-\tau) \tag{7.11}$$

in the orthographic case.

Equations 7.9, 7.10 and 7.11 relate the first derivatives of the geometric mapping function $\mathbf{x}_s$, i.e. $\partial_{x_i}\mathbf{x}_s$ and $\partial_{y_i}\mathbf{x}_s$, with the shape parameters z_w, τ, σ.

In fact we compute the local orientation parameters $(\tau(x_w, y_w),\ \sigma(x_w, y_w))$ from the shape given as input in the form of its height field $z_w(x_w, y_w)$:

$$\tau(x_w, y_w) = \tan^{-1}\left(\frac{\partial_{y_w} z_w}{\partial_{x_w} z_w}\right) \tag{7.12}$$

$$\sigma(x_w, y_w) = \tan^{-1}\left[\sqrt{(\partial_{x_w} z_w)^2 + (\partial_{y_w} z_w)^2}\right] \tag{7.13}$$

Once the shape parameters $z_w(x_w, y_w)$, $\tau(x_w, y_w)$, $\sigma(x_w, y_w)$ are known as functions of world coordinates $\mathbf{x}_w$, they may be projected onto the image plane to obtain $z_w(\mathbf{x}_i)$, $\sigma(\mathbf{x}_i)$ and $\tau(\mathbf{x}_i)$ by using the relevant equations for the perspective and orthographic cases (equations 4.3 and 5.12).

In particular the equations for perspective projection are

$$x_i = \frac{f}{z_w} x_w \tag{7.14}$$

$$y_i = \frac{f}{z_w} y_w\ , \tag{7.15}$$

and for orthographic projection

$$x_i = x_w \tag{7.16}$$

$$y_i = y_w\ . \tag{7.17}$$

We therefore have

$$z_w(x_i, y_i) \equiv z_w\left(\frac{z_w}{f}x_i, \frac{z_w}{f}y_i\right) \tag{7.18}$$

$$\tau(x_i, y_i) \equiv \tau\left(\frac{z_w}{f}x_i, \frac{z_w}{f}y_i\right) \tag{7.19}$$

$$\sigma(x_i, y_i) \equiv \sigma\left(\frac{z_w}{f}x_i, \frac{z_w}{f}y_i\right) \tag{7.20}$$

for perspective projection, and

$$z_w(x_i, y_i) \equiv z_w(x_w, y_w) \tag{7.21}$$

$$\tau(x_i, y_i) \equiv \tau(x_w, y_w) \tag{7.22}$$

$$\sigma(x_i, y_i) \equiv \sigma(x_w, y_w) \tag{7.23}$$

for orthographic projection.

By substituting for the respective z_w, τ and σ from equations 7.18, 7.19, 7.20 (perspective) or 7.21, 7.22, 7.23 (orthographic) in equations 7.10, 7.9 (perspective) or 7.11, 7.9 (orthographic), we compute the first order derivatives $\partial_{x_i}\mathbf{x}_s(\mathbf{x}_i)$ and $\partial_{y_i}\mathbf{x}_s(\mathbf{x}_i)$ as a function of the input shape and of the image coordinates $\mathbf{x}_i$. Finally, by integrating the derivatives, we obtain the *geometric mapping function* $\mathbf{x}_s(\mathbf{x}_i)$ *for a general shape*.

In particular, the integration of $\mathbf{x}_s(\mathbf{x}_i)$ is achieved with the method of [50]. As described in section 2.5, the method is based on forcing integrability by least-squares fitting the gradients of the mapping function to integrable Fourier bases. We observe that the use of local derivatives and the method of [50] also produces pleasing results on non-developable surfaces, where the stretch of the applied texture is smooth and minimally distorted, as we show in section 7.2.1.

The geometric mapping $\mathbf{x}_s(\mathbf{x}_i)$ unfolds the surface texture to a virtual plane, on which we 'lay' and re-render a new texture.

Given the geometric mapping, a new image A is obtained by mapping onto the shape a new texture I, in both perspective and orthographic cases, using the following:

$$A(\mathbf{x}_i) = I(\mathbf{x}_s(\mathbf{x}_i)) \ . \tag{7.24}$$

We again note that the term 'texture' may refer here to *any* image, not just a homogeneous texture image.

The re-rendering consists of a linear interpolation which re-samples the values of I according to the function $\mathbf{x}_s(\mathbf{x}_i)$. As described in the introduction 7.1.1, this may imply an aliasing when the texture I contains high frequencies, which can be solved by over-sampling or low-pass filtering the texture I. However, this aliasing issue is not discussed further here, as we wish to focus on the geometric mapping function.

We also note that it is possible to get a perspectively projected texture by using the geometric mapping computed from the orthographic equation 7.11. Equation 7.11 is simpler and less computationally expensive than equation 7.10 (note that the first derivatives have to be computed at each pixel of the image).

To use an orthographically-computed geometric mapping and render a texture perspectively, we firstly derive the relation between orthographic and perspective projections. From equations 7.14, 7.15, 7.16, 7.17 we have

$$\mathbf{x}_i^{\text{persp}} = \frac{f}{z_w}\mathbf{x}_i^{\text{orth}} \ , \tag{7.25}$$

or alternatively

$$\mathbf{x}_i^{\text{orth}} = \frac{z_w}{f}\mathbf{x}_i^{\text{persp}} \ . \tag{7.26}$$

Equation 7.26 allows us to produce a perspective image A using an orthographic geometric mapping function $\mathbf{x}_s(\mathbf{x}_i)$:

$$A(\mathbf{x}_i) = I\left(\mathbf{x}_s\left(\frac{z_w}{f}\mathbf{x}_i\right)\right) \ . \tag{7.27}$$

7.2.1 Examples of texture mapping

Figure 7.1 presents two examples of texture mapping, where the method discussed in section 7.2 is used. In particular the examples represent the mapping of two textures, D53 and D76 taken from the Brodatz database ([18]), onto the shapes of a segment of sphere and a corner of a cube.

In both examples the results are visually pleasing, and they illustrate how the local equations and the integration method of [50] minimise the distortion of mapping the 2D textures onto the non-developable shapes.

Figures 7.2 and 7.3 show another possible application of the texture mapping algorithm: replacing the texture by mapping a new texture onto the reconstructed shape. The figures illustrate the cases of the pillowcase on the cylinder and of the rubber rug on a plane, which were presented in figures 6.3 and 6.4.

Figures 7.2 and 7.3 illustrate the processed image, the reconstructed shapes (with algorithm 6), the new 2D textures that we want to apply (respectively in figures 7.2(c) and 7.3(c)), and the result of mapping the new textures onto the shapes (respectively in figures 7.2(d) and 7.3(d)).

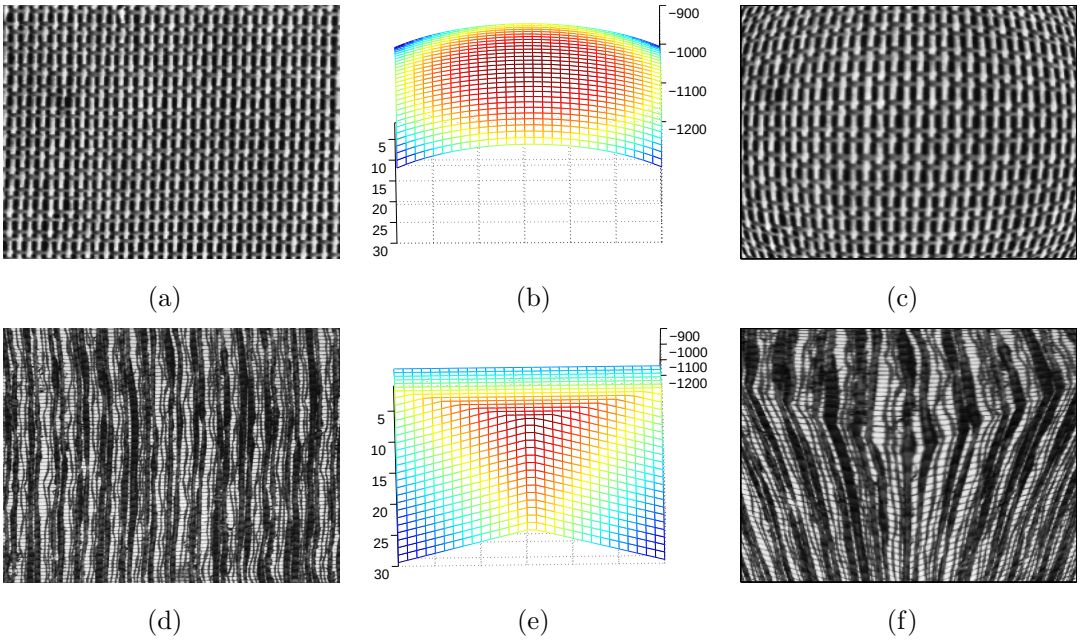
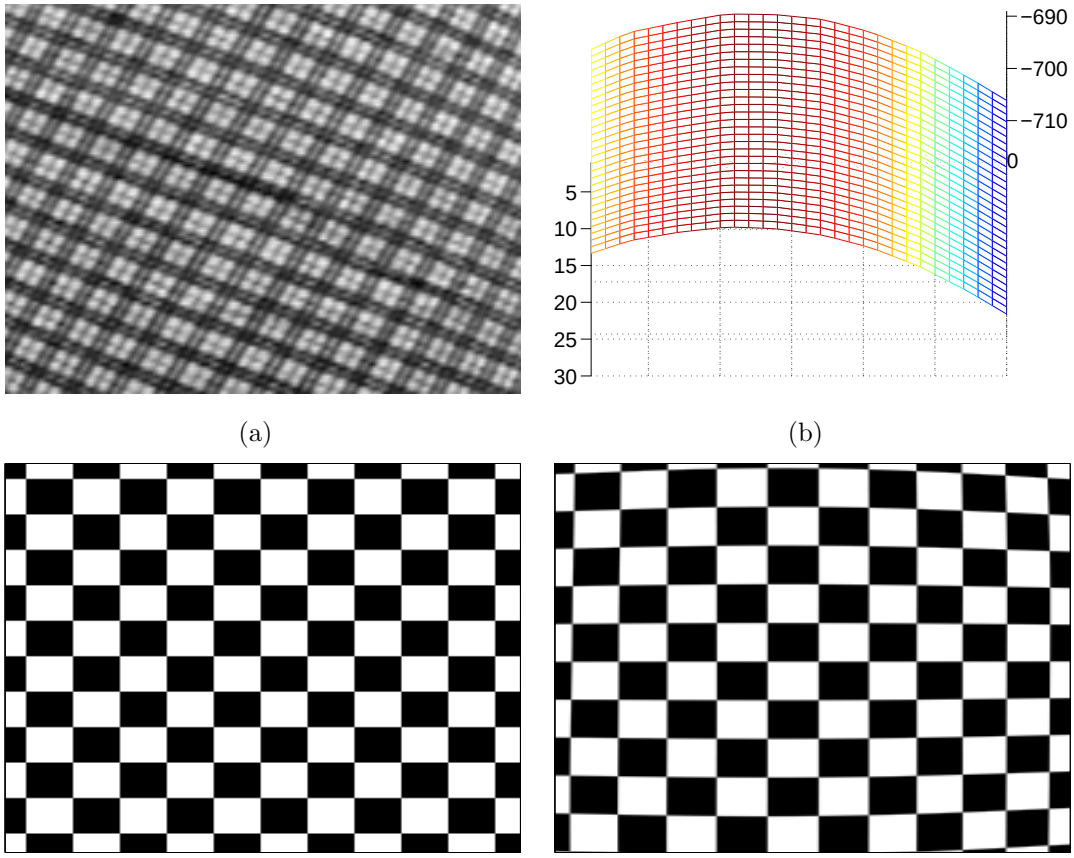


Figure 7.1: (a) D53 Brodatz texture; (b) segment of a sphere; (c) D53 Brodatz texture rendered on the segment of a sphere given in (b); (d) D76 Brodatz texture; (e) corner of a cube; (f) D76 Brodatz texture rendered on the corner of a cube given in (e)



(a) (b)

(c) (d)

Figure 7.2: (a) Real picture of a pillowcase laid on a cylinder, as illustrated in fig. 6.3(b); (b) perspective reconstruction of the cylindrical shape from image (a), as shown in fig. 6.3(f); (c) chessboard motif; (d) chessboard motif rendered on the shape of (b)

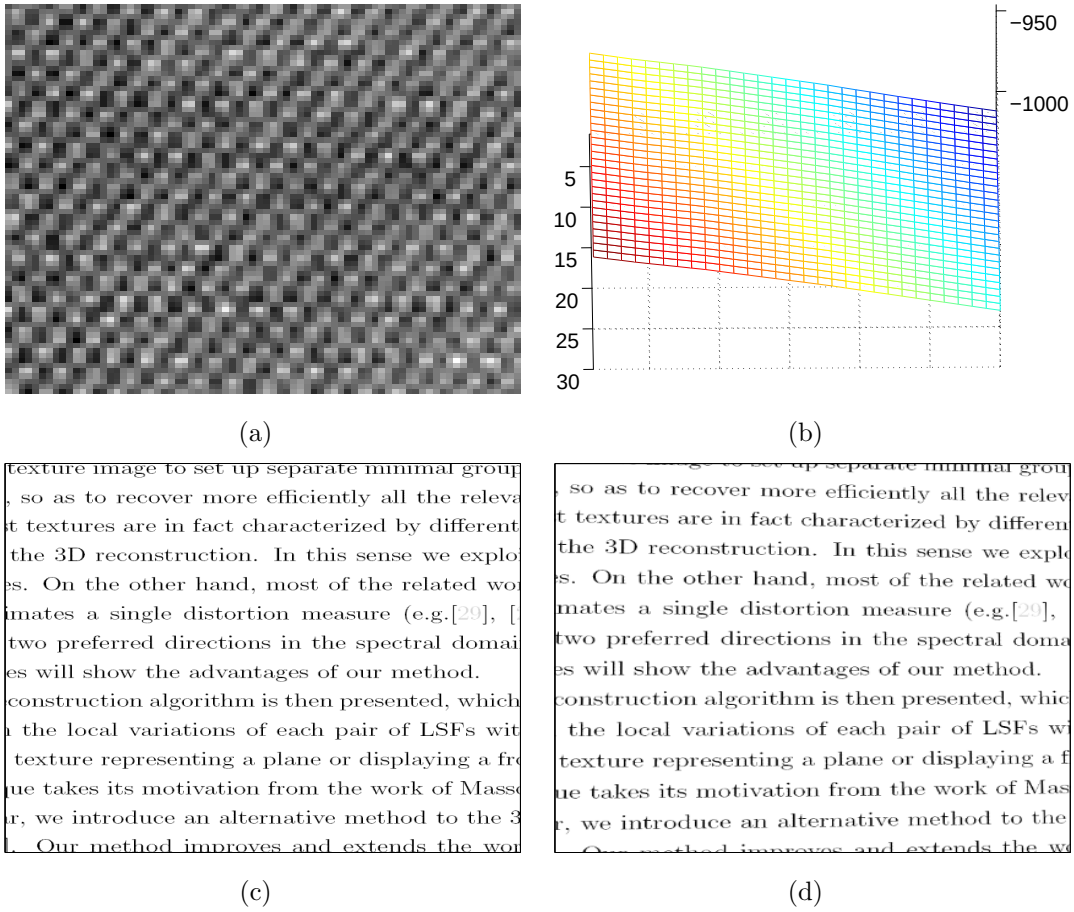


Figure 7.3: (a) Real picture of a rubber rug laid on a planar surface, as illustrated in fig. 6.4(b); (b) perspective reconstruction of the planar shape from image (a), as shown in fig. 6.4(f); (c) written page motif; (d) written page motif rendered on the shape of (b)

We observe that the 3D shape can also be modified before the texture mapping and that the new rendered texture can be observed from a different viewpoint with a different camera.

The algorithm is fast, as it only requires elementary operations and an integration with the method of [50], also very computationally efficient. In fact the slowest part of the algorithm at the moment is the linear interpolation that realises the filtering (MATLAB function 'interp2'), and we note that a detailed investigation into filtering does not form part of the proposed algorithm.

Finally, we note that the algorithm presented here shows a different application of the geometric equations of section 5.3.1. For the moment, only cases where a one to one mapping between the surface texture and the image plane have been considered. More complex issues, e.g. 'holes', would require further investigation.

7.3 Texture Replacement

Here we propose an original algorithm for texture replacement which re-renders a new image onto a given texture without estimating the underlying 3D shape. The algorithm uses the LSF-based multi-scale distortion measure of chapter 3 to distort the new image according to the input texture: it renders the new image as if it was laid onto the same shape and photographed from the same point of view with the same camera.

Let us recall, as in the previous section 7.2, the revisited viewing geometry and projection model of section 5.3.1, with the relevant world coordinates $\mathbf{x}_w = (x_w, y_w, z_w)$, image coordinates $\mathbf{x}_i = (x_i, y_i)$, orientation parameters σ,τ and the aligned surface texture coordinates $\mathbf{x}_s = (x_s, y_s)$.

Let us also recall equations 5.44, 5.45 and 5.47, which related the image plane and surface texture frequencies to the orientation parameters σ,τ and the image plane coordinates $\mathbf{x}_i$. In particular we consider the following:

$$\mathbf{u}_i = J^t(\mathbf{x}_s, \mathbf{x}_i)\mathbf{u}_s = \begin{bmatrix} \partial_{x_i} x_s & \partial_{x_i} y_s \\ \partial_{y_i} x_s & \partial_{y_i} y_s \end{bmatrix} \mathbf{u}_s \ , \tag{7.28}$$

which we can also write as

$$u_i = u_s \partial_{x_i} x_s(\mathbf{x}_i) + v_s \partial_{x_i} y_s(\mathbf{x}_i) \tag{7.29}$$

$$v_i = u_s \partial_{y_i} x_s(\mathbf{x}_i) + v_s \partial_{y_i} y_s(\mathbf{x}_i) \ . \tag{7.30}$$

In equations 7.28, 7.29 and 7.30, $\mathbf{u}_i$ is the image plane frequency value for a particular LSF, $\mathbf{u}_s$ is the corresponding surface texture frequency (constant over the surface texture according to the homogeneity assumption of section 2.2), and the vector function $\mathbf{x}_s(\mathbf{x}_i)$ is the geometric mapping, as in section 7.2, that we want to compute.

As seen in section 5.3.2, the surface texture frequency $\mathbf{u}_s$ may be recovered by using a displayed frontal point in the image (as indicated in section 5.1) and equations 5.53 and 5.59, by using a point of known orientation, or by measuring the texture frequency on a separately provided frontal patch of the same texture.

Let us assume here that the $\mathbf{u}_s$ is known for the LSFs that we consider, leaving to section 7.3.2 the discussion of the case when this information is not available.

If at least two LSFs are available from the multi-scale distortion estimation algorithm of chapter 3, we can write equation 7.28 for two different LSFs, $\mathbf{u}'_i = (u'_i, v'_i)$ and $\mathbf{u}''_i = (u''_i, v''_i)$, and then combine the relations into one matrix equation, as in equations 5.51 and 6.1:

$$U_i = \begin{bmatrix} \partial_{x_i} x_s & \partial_{x_i} y_s \\ \partial_{y_i} x_s & \partial_{y_i} y_s \end{bmatrix} U_s \tag{7.31}$$

$$\text{where} \quad U_i = \begin{bmatrix} u'_i & u''_i \\ v'_i & v''_i \end{bmatrix} \quad \text{and} \quad U_s = \begin{bmatrix} u'_s & u''_s \\ v'_s & v''_s \end{bmatrix} \ . \tag{7.32}$$

In this equation, the 2x2 matrix U_s contains the surface texture frequencies corresponding to the two LSFs, which are constant over the texture surface, and the 2x2 matrix U_i contains the image frequency values for the two LSFs, which are functions of the particular point $\mathbf{x}_i$.

By assuming that the two considered LSFs have different spectral directions, i.e. that they are distinct and not multiples of each other, the frequency vector values $\mathbf{u}'_s$ and $\mathbf{u}''_s$ are linearly independent and the matrix U_s can be inverted to determine the local derivatives of the geometric mapping function $\mathbf{x}_s(\mathbf{x}_i)$ at all

the image pixels:

$$\begin{bmatrix} \partial_{x_i} x_s & \partial_{x_i} y_s \\ \partial_{y_i} x_s & \partial_{y_i} y_s \end{bmatrix} = U_i U_s^{-1} \ . \tag{7.33}$$

Alternatively, if more than two LSFs are available, equation 7.31 still holds and matrices U_i and U_s can be used to group the different surface texture and image plane LSFs:

$$U_i = \begin{bmatrix} u_i' & u_i'' & u_i''' & \dots \\ v_i' & v_i'' & v_i''' & \dots \end{bmatrix} \quad \text{and} \quad U_s = \begin{bmatrix} u_s' & u_s'' & u_s''' & \dots \\ v_s' & v_s'' & v_s''' & \dots \end{bmatrix} \ . \tag{7.34}$$

In this case, by assuming that the two rows in matrices U_i and U_s are independent, i.e. that at least two of the n considered LSFs are linearly independent, the matrix of first derivatives of the geometric mapping in equation 7.33 can still be found by 'inverting' the matrix U_s, here using the pseudo-inverse:

$$\begin{bmatrix} \partial_{x_i} x_s & \partial_{x_i} y_s \\ \partial_{y_i} x_s & \partial_{y_i} y_s \end{bmatrix} = U_i U_s^{\dagger} = U_i U_s^t (U_s U_s^t)^{-1} \ . \tag{7.35}$$

The pseudo-inverse finds a least squares solution to the over-constrained system.

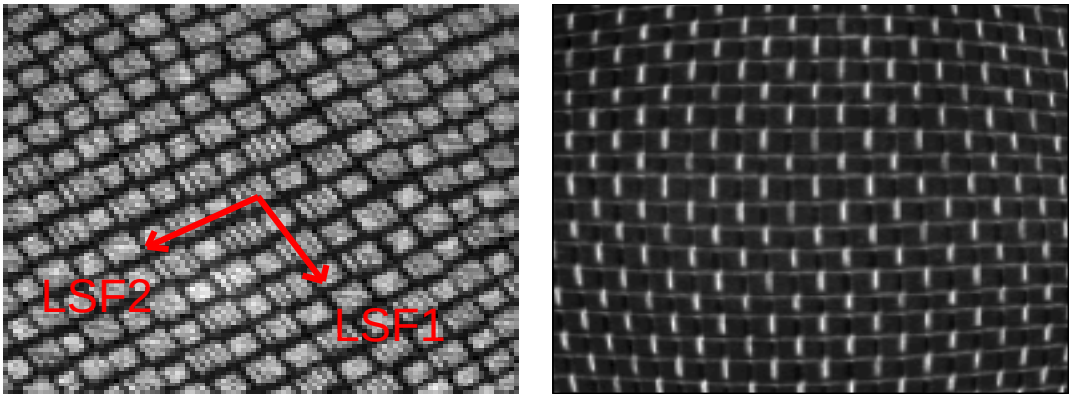
As seen in section 7.2, the first derivatives are integrated to determine the geometric mapping $\mathbf{x}_s(\mathbf{x}_i)$ with the method of [50].

Then, the new image A is obtained by rendering the new texture I according to the computed geometric mapping:

$$A(\mathbf{x}_i) = I(\mathbf{x}_s(\mathbf{x}_i)) \ . \tag{7.36}$$

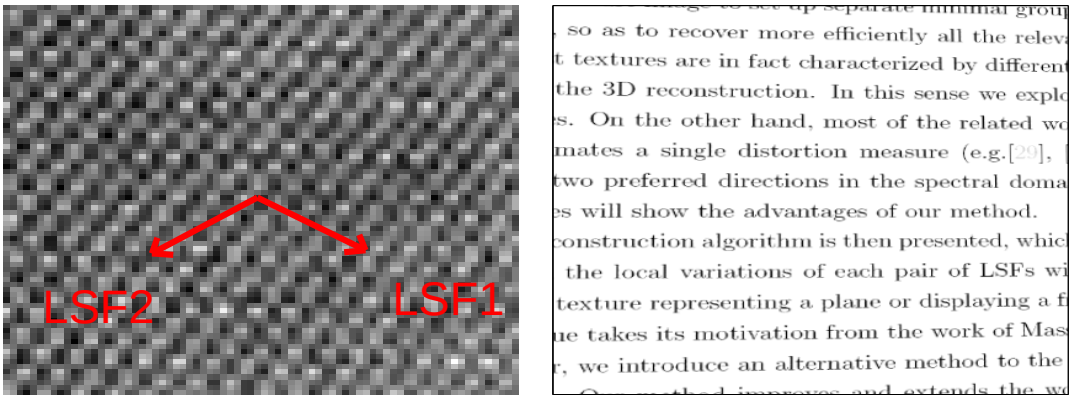
Again we note here that the new texture I can be any image, i.e. it does not need to be homogeneous. Also we reiterate that the rendering consists here of a simple linear interpolation which re-samples the values of I according to the function $\mathbf{x}_s(\mathbf{x}_i)$. A filtering algorithm to avoid aliasing (when texture I contains high frequencies) is left for future investigation.

Note that the algorithm described produces a new image A by deforming the texture I in exactly the same way as the original texture. The new image A will therefore show the texture I laid on the same shape, observed from the same viewpoint, and photographed with the same camera.



(a) (b)

Figure 7.4: (a) Real picture of a textured cloth laid on an inflated balloon, with arrows indicating the spectral directions of the two selected LSFs, as illustrated in fig. 6.5(d); (b) D6 Brodatz texture, shown in fig. 3.7(a), replaced onto the image of (a) by using the two indicated LSFs



(a) (b)

Figure 7.5: (a) Real picture of a rubber rug laid on a planar surface, as in fig. 7.3(a), with arrows indicating the spectral directions of the two selected LSFs, as illustrated in fig. 6.4(d); (b) written page motif of fig. 7.3(c) replaced onto the original image of (a) by using the two indicated LSFs

7.3.1 Examples of texture replacement

Figures 7.4 and 7.5 present some examples of re-texturing where the proposed algorithm is used to replace the original texture by using the computed LSFs.

Figure 7.4 re-illustrates the case of a textured cloth laid on the inflated balloon of figures 5.12 and 6.5. The D6 Brodatz texture ([18]), shown in figure 3.7(a), is replaced onto the cloth texture by using the two estimated LSFs (represented in figure 3.16).

Figure 7.5 shows how we replaced the texture of the rubber rug on the planar surface, as in figure 6.4, with the texture of the written page (illustrated in figure 7.3(c)), by using the estimated LSFs.

As noted for the texture mapping algorithm of section 7.2.1, the proposed re-texturing algorithm is based on elementary operations (the inverse or pseudo-inverse of a matrix) and on an integration with the method of [50]. The algorithm is therefore fast, and the slowest part is the linear interpolation (MATLAB function 'interp2'), which provisionally implements the filtering part. More calculations, especially convolutions, are required for the multi-scale LSF estimation, as explained in chapter 3.

As we only address the geometric mapping part of the proposed re-texturing algorithm, a comparison with other texture replacement algorithms is not possible. Moreover, the algorithm is here proposed as an application of the LSF-based multi-scale estimation method and of the geometric equations of section 5.3.1, more complex matters, e.g. 'holes' or re-applying the shading of the original picture, have therefore not been addressed, but are suggested for future work.

The re-texturing algorithm presented here is original, and we note that no solution has previously been proposed using frequencies. Furthermore, for substituting a different texture onto the same picture, the computation described does not have to be repeated and the geometric mapping can just be used in the same way with the new different texture.

Moreover, being based on the estimated LSFs, the proposed re-texturing algorithm is robust against variations of illumination, shading, surface texture contrast and imperfections, as we observed for the multi-scale distortion estimation of chapter 3.

Finally we note that figure 7.5 can be compared with figure 7.3(d): the latter replaces the original texture by using the recovered shape (figure 7.3(b)), while the former only uses the estimated LSFs. The nearly identical results demonstrate that the presented texturing and re-texturing algorithms are consistent in solving for the texture replacement problem.

It is important to notice that the image of figure 7.5(a) did not show a frontal point, nor was the orientation of any point of the planar surface assumed known. In fact the algorithm used the central image point and its LSF values as if it were the frontal point. The approximation implies a systematic distortion of the newly applied texture, as we describe in section 7.3.2, which is here unnoticeable.

7.3.2 Texture replacement when a frontal point is not available

Here we analyse the distortion that we get when a frontal point is not displayed in the considered image (nor a point of known orientation nor a separate available frontal patch of the same texture) and we approximate the frontal point with any point in the image.

First, let us recapitulate the main findings of sections 7.2 and 7.3. Equations 7.9, 7.28 and 7.31 showed that

$$U_i = J^t(\mathbf{x}_s, \mathbf{x}_i) U_s = \begin{bmatrix} \partial_{x_i} x_s & \partial_{x_i} y_s \\ \partial_{y_i} x_s & \partial_{y_i} y_s \end{bmatrix} U_s \ , \tag{7.37}$$

where $J^t(\mathbf{x}_s, \mathbf{x}_i)$ is the transpose of the Jacobian of the system coordinate transformation. Its value has been analytically computed in section 7.2 in equations 7.10 (perspective) and 7.11 (orthographic). There we also observed that it can be written as a function of the image screen coordinates $\mathbf{x}_i$, and therefore we adopt here the notation already used in equations 5.19 and 5.48:

$$J^t(\mathbf{x}_s, \mathbf{x}_i) = J^t_{\mathbf{x}_s \mathbf{x}_i}(\mathbf{x}_i) \ . \tag{7.38}$$

The notation indicates that the transpose Jacobian was derived by differentiating the geometric mapping function $\mathbf{x}_s(\mathbf{x}_i)$ with respect to $\mathbf{x}_i$, and that it is a function of the image plane coordinates $\mathbf{x}_i$.

According to the homogeneity assumption, the surface texture frequencies (contained in U_s), are constant over the surface texture. For an image plane point $\mathbf{x}_{if}$ corresponding to a frontal point we have in particular

$$U_{if} = J^t_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_{if})U_s \ , \tag{7.39}$$

and the surface texture frequencies can be expressed as

$$U_s = (J^t_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_{if}))^{-1}U_{if} \ , \tag{7.40}$$

having assumed that at least two of the LSFs in U_s have different spectral direction, as described in section 7.3.

In fact we saw (equations 5.53 and 5.59) that for a frontal point the transpose Jacobian assumes convenient values

$$J^t_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_{if}) = \begin{cases} I & \text{for orthograhic cases} \\ \frac{z_0}{f}I & \text{for perspective cases} \end{cases} \ . \tag{7.41}$$

Combining equation 7.37 with equation 7.40, we can write

$$U_i = J^t_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_i)(J^t_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_{if}))^{-1}U_{if} \ , \tag{7.42}$$

from which

$$J^t_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_i) = \begin{bmatrix} \partial_{x_i}x_s & \partial_{x_i}y_s \\ \partial_{y_i}x_s & \partial_{y_i}y_s \end{bmatrix} = U_iU_{if}^{-1}J^t_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_{if}) \ . \tag{7.43}$$

According to equation 7.43, the first derivatives of the geometric mapping function $\mathbf{x}_s(\mathbf{x}_i)$ can be computed at each pixel from the measured image plane frequencies at the considered point $\mathbf{x}_i$ (in U_i), the image plane frequencies at $\mathbf{x}_{if}$ (in U_{if}) and the coordinates of $\mathbf{x}_{if}$ (additionally f and z_0 are required for the perspective case).

When a frontal point is not contained in the image, nor is there any point of known orientation, nor is the frontal appearance of the texture available, it is still possible to use our method to replace the texture.

In what follows we will assume that the type of projection is not known. In fact a smooth surface is required for approximating the relation between the image plane and surface texture frequencies to first order with the transpose Jacobian, as in equation 7.37, however we do not need to know the analytical form of the transpose Jacobian matrix.

Let us consider a general point in the image $\mathbf{x}_{i0}$. Equation 7.42 becomes:

$$U_i = J^t_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_i)(J^t_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_{i0}))^{-1}U_{i0} \ , \tag{7.44}$$

which we write as

$$J^t_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_i)(J^t_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_{i0}))^{-1} = U_i(U_{i0})^{-1} \ . \tag{7.45}$$

In equation 7.45, the transpose Jacobian $J^t_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_i)$ represents the matrix of first derivatives of the geometric mapping function $\mathbf{x}_s(\mathbf{x}_i)$ that we want to compute, the frequencies in matrices U_i and U_{i0} are measured on the image screen, while the term $J^t_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_{i0})$ is unknown, but constant, and depends on the projection that we use (orthographic or perspective, possibly including non-linearities from the lenses) and on the local orientation at point $\mathbf{x}_{i0}$.

Knowing that $J^t_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_{i0})$ is constant, we can set a constant 2x2 matrix to represent it:

$$C = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} = (J^t_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_{i0}))^{-1} \ . \tag{7.46}$$

Equation 7.45 becomes

$$\begin{aligned} J^t_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_i)C &= \begin{bmatrix} \partial_{x_i}x_s & \partial_{x_i}y_s \\ \partial_{y_i}x_s & \partial_{y_i}y_s \end{bmatrix} C = \begin{bmatrix} C_{11}\partial_{x_i}x_s + C_{21}\partial_{x_i}y_s & C_{12}\partial_{x_i}x_s + C_{22}\partial_{x_i}y_s \\ C_{11}\partial_{y_i}x_s + C_{21}\partial_{y_i}y_s & C_{12}\partial_{y_i}x_s + C_{22}\partial_{y_i}y_s \end{bmatrix} \\ &= \begin{bmatrix} \partial_{x_i}[C_{11}x_s + C_{21}y_s] & \partial_{x_i}[C_{12}x_s + C_{22}y_s] \\ \partial_{y_i}[C_{11}x_s + C_{21}y_s] & \partial_{y_i}[C_{12}x_s + C_{22}y_s] \end{bmatrix} = U_i(U_{i0})^{-1} \ . \end{aligned} \tag{7.47}$$

Therefore, by using equation 7.47 at any pixel, we can compute the values of the first derivatives of the functions $[C_{11}x_s + C_{21}y_s]$ and $[C_{12}x_s + C_{22}y_s]$ from the values of the measured frequencies in matrices U_i and U_{i0}. And by numerically integrating the first derivatives with the method of [50], we obtain the value at each point of the functions $[C_{11}x_s + C_{21}y_s]$ and $[C_{12}x_s + C_{22}y_s]$.

We indicate the estimated functions as follows:

$$\hat{\mathbf{x}}_s(\mathbf{x}_i) = \begin{pmatrix} \hat{x}_s(\mathbf{x}_i) \\ \hat{y}_s(\mathbf{x}_i) \end{pmatrix} = \begin{pmatrix} C_{11}x_s + C_{21}y_s \\ C_{12}x_s + C_{22}y_s \end{pmatrix} \ . \tag{7.48}$$

The newly defined vector function $\hat{\mathbf{x}}_s(\mathbf{x}_i)$ is related to the true geometric mapping $\mathbf{x}_s(\mathbf{x}_i)$ as follows:

$$\mathbf{x}_s(\mathbf{x}_i) = \begin{pmatrix} x_s(\mathbf{x}_i) \\ y_s(\mathbf{x}_i) \end{pmatrix} = C^{-t}C^t\mathbf{x}_s(\mathbf{x}_i) = C^{-t}\begin{pmatrix} C_{11}x_s + C_{21}y_s \\ C_{12}x_s + C_{22}y_s \end{pmatrix} \tag{7.49}$$

$$= C^{-t}\hat{\mathbf{x}}_s(\mathbf{x}_i) = J_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_{i0})\hat{\mathbf{x}}_s(\mathbf{x}_i) \ . \tag{7.50}$$

Therefore, instead of computing $\mathbf{x}_s(\mathbf{x}_i)$, by integrating equation 7.47 we compute

$$(J_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_{i0}))^{-1}\mathbf{x}_s(\mathbf{x}_i) \tag{7.51}$$

By using the alternative geometric mapping $\hat{\mathbf{x}}_s(\mathbf{x}_i)$, we can still replace the texture in the original image according to equation 7.36, which becomes here:

$$\hat{A}(\mathbf{x}_i) = I[\hat{\mathbf{x}}_s(\mathbf{x}_i)] = I[(J_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_{i0}))^{-1}\mathbf{x}_s(\mathbf{x}_i)] \ . \tag{7.52}$$

The quality of the texture replacement depends now on the distortion of the chosen point $\mathbf{x}_{i0}$, and on the nature of the new texture that we want to apply.

In fact, using $\hat{\mathbf{x}}_s(\mathbf{x}_i)$ instead of $\mathbf{x}_s(\mathbf{x}_i)$ as the geometric mapping is equivalent to deforming the new texture I according to $(J_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_{i0}))^{-1}$, we call the distorted texture I', before re-rendering it onto the image according to the true geometric mapping function $\mathbf{x}_s(\mathbf{x}_i)$.

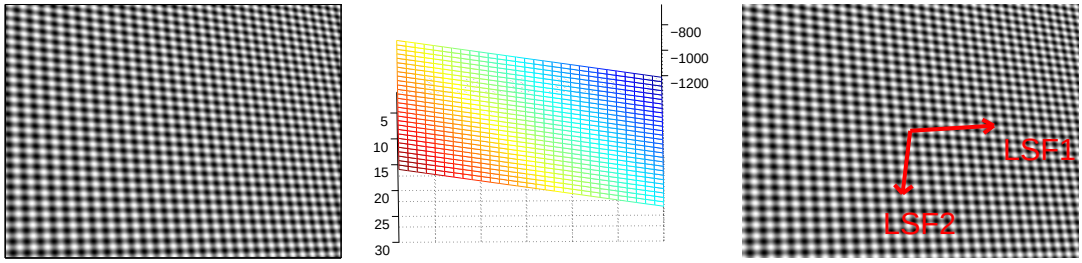
In other words, the new rendered image $\hat{A}$ is the same as we would get by applying the true mapping function $\mathbf{x}_s(\mathbf{x}_i)$ to the texture I', obtained by uniformly distorting I according to the distortion opposite to that at point $\mathbf{x}_{i0}$. We note that the texture at $\mathbf{x}_{i0}$ is related to the original texture I by the transformation matrix $J_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_{i0})$, and that the distortion opposite to that at $\mathbf{x}_{i0}$ is therefore given by the transformation matrix $(J_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_{i0}))^{-1}$.

As a result, the newly synthesized image $\hat{A}$ presents a null distortion of the texture I at the chosen point $\mathbf{x}_{i0}$ (where the applied distortion $J_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_{i0})$ compensates the distortion opposite to that of $\mathbf{x}_{i0}$ which we have at each pixel, $(J_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_{i0}))^{-1}$) and a distortion opposite to that at $\mathbf{x}_{i0}$ at a frontal point (where the distortion to apply should be null, leaving the un-compensated base distortion $(J_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_{i0}))^{-1}$).

Thus, the new texture is applied consistently over the original surface texture. However the effective applied texture is I', instead of I.

For this reason, the approximation goes unnoticed when the texture we wish to apply is not a picture but a geometric motif: deforming a pattern of superposed cosines gives an equivalent pattern of superposed cosines with different frequencies.

Confirming the theory, section 7.3.3 shows that pre-slanting the new texture I according to $J_{\mathbf{x}_s\mathbf{x}_i}(\mathbf{x}_{i0})$ and using $\hat{\mathbf{x}}_s(\mathbf{x}_i)$ produces the same result as using I and the true $\mathbf{x}_s(\mathbf{x}_i)$.



(a) (b) (c)

Figure 7.6: (a) Texture of two superposed cosines rendered on a plane; (b) planar surface underlying (a); (c) texture of (a), with arrows indicating the spectral directions of the two selected LSFs

Finally we again underline that in equation 7.52 we do not assume any particular camera model. This allows us to generalize our method to any camera as long as the approximate relationship between image and surface texture frequencies can be expressed as in equation 7.37.

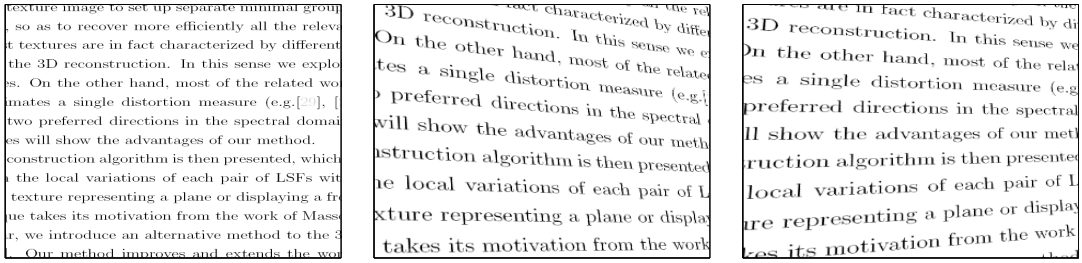
7.3.3 Examples of texture replacement when a frontal point is not available

Figure 7.5(b) in section 7.3.1 has illustrated the case when a frontal point is not available (nor a point of known orientation nor a separate frontal patch of the same texture).

Comparing figure 7.5(b) with 7.3(d) (i.e. comparing the approximately-replaced-texture image with that where the new texture has been appropriately replaced by mapping it onto the recovered shape) shows how the distortion due to the approximation is negligible, and using the central point instead of a frontal point cannot be noticed.

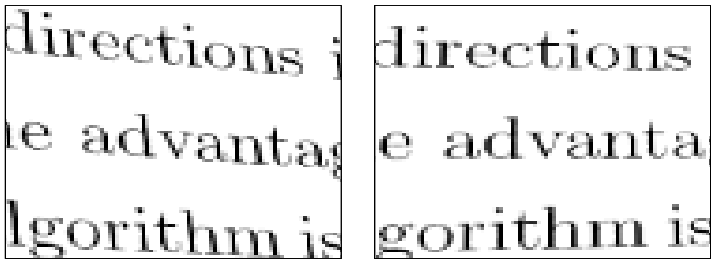
This happens in the case of figures 7.5(b) and 7.3(d) because in the image the texture changes only slightly, and equally slight is the distortion due to the approximation.

To illustrate a case which better shows the distortion described we consider the image in figure 7.6(a). The image has been synthesized by mapping a motif



(a) (b) (c)

Figure 7.7: (a) Written page motif, reproduced from fig. 7.3(c); (b) written page motif of (a) rendered onto the planar shape of fig. 7.6(b); (c) written page motif of (a) replaced onto the image of fig. 7.6(a) by using the two available LSFs and the central point of fig. 7.6(a)



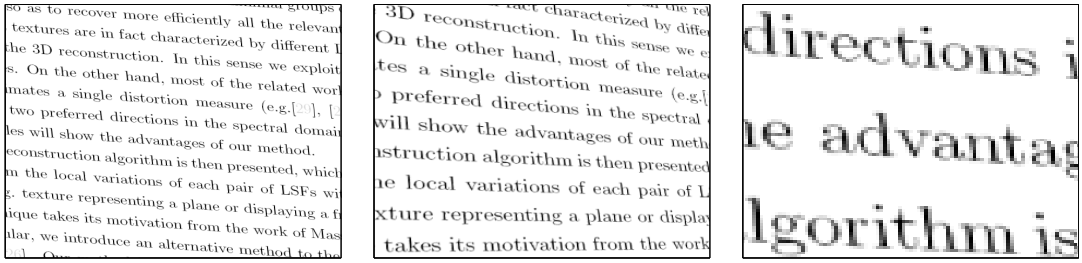
(a) (b)

Figure 7.8: Details of the central parts of (a) fig. 7.7(b) and (b) fig. 7.7(c)

of two superposed cosines onto the planar surface of figure 7.6(b). The LSFs associated with the two cosines can be recovered: in fact figure 7.6(c) shows the spectral direction of the two available LSFs at the central point.

As the utilised plane has the same tilt direction but greater slant, than in the previous cases, in figures 7.5(b) and 7.3(d), we expect to see that approximating a frontal point as the central point in the re-texturing algorithm results in a greater distortion.

In fact figure 7.7 shows the result of texturing the new texture of figure 7.7(a) (the written page motif of figure 7.3(c) used in figures 7.5(b) and 7.3(d)) onto the recovered 3D shape together with the result of re-texturing the original image of figure 7.6(a) by using the two recovered LSFs, the central point and the same



(a) (b) (c)

Figure 7.9: (a) Written page motif of fig. 7.7(a) pre-slanted according to the orientation of the central point of fig. 7.6(b); (b) pre-slanted written page motif of (a) replaced onto the image of fig. 7.6(a) by using the two available LSFs and the central point of fig. 7.6(a); (c) detail of the central part of (b)

new texture of figure 7.7(a).

Figure 7.8 shows the respective details of the central parts of figures 7.7(b) and 7.7(c): as section 7.3.2 described, in figure 7.8(a) the central part appears properly slanted, while in figure 7.8(b) the same central part appears unslanted. This accounts for the fact that the new texture that is effectively applied in figure 7.7 is virtually pre-deformed according to the opposite slant of the utilised central point. This results in a null distortion at the central point.

As also described in section 7.3.2, if we consider a new texture I'', pre-distorted according to the distortion at the central point, and re-texture the original image using I'' and the central point, then the results coincide.

In fact figure 7.9(a) shows the pre-distorted I'' and figures 7.9(b) and 7.9(c) show the result of re-texturing the original image of figure 7.6(a) by using I'': both the whole image and its detail perfectly agree with the result of the texturing (respectively in figures 7.7(b) and 7.8(a)), which confirms the correctness of our assertions.

7.4 Final Remarks

This chapter has presented two possible applications of the multi-scale distortion estimation, presented in chapter 3, and the geometric relations between the

surface texture and the image plane, which were studied in chapters 4, 5 and 6.

In particular a texture mapping algorithm has been presented, which uses the equations relating the surface texture and the image plane coordinates and their first derivatives to recover the geometric mapping.

The algorithm has been used to render textures onto synthetic 3D shapes and to replace textures by re-rendering new ones onto the recovered (with the use of a SFT algorithm) 3D shape (which can be modified, observed from a different viewpoint and with a different camera before the mapping).

A texture replacement algorithm has also been presented which replaces the existing texture of an image by using the estimated LSFs and a frontal point.

This second algorithm represents a new approach to re-texturing where a recovered 3D shape is not necessary, unlike existing texture replacement methods in the literature.

With this algorithm, new textures are replaced onto the original image exactly reproducing the original distortion: the new texture looks as if it was rendered onto the same shape and observed from the same viewpoint with the same camera.

Both algorithms have been tested on semi-synthetic and real images, and relevant examples have been presented to illustrate the distortion due to approximating a frontal with the central point in the re-texturing algorithm.

In particular the texture mapping technique has been shown to appropriately optimise the rendering of 2D texture onto 3D non-developable textures, while the texture replacement method advantageously inherits relevant robustness against variations of illumination, shading, surface texture contrast and imperfections from the LSF-estimation technique.

Since the purpose of the chapter is to introduce new original approaches as applications of the implemented SFT techniques, the geometric mapping alone has been analysed in simple cases. However, future investigations will consider more complex shapes and improve on the currently used filtering method.

Chapter 8

Conclusions and Future Work

This thesis has presented work on Shape-From-Texture under the assumption of 'stochastic' homogeneity.

In other words, the problem of recovering the 3D coordinates of a shape from its 2D monocular textured image has been here investigated, under the general assumption that the texture in the scene is the realization of a stochastic process which is stationary under translation.

8.1 Contributions

The work presented has generally assumed that the texture is the 'same' at different points of the surface in the scene, i.e. that the texture is a realization of a stochastic process stationary under translation. We have therefore assumed that the observed deformation is only due to the projection from the 3D world to the 2D image.

We have contributed to the relevant state of the art of SFT with the following original work.

- The definition of a multi-scale distortion estimation based on LSFs. The combined use of the Fourier transform and Gabor filters has been shown to be both computationally efficient and suitable to deal with natural textures, often containing useful information at multiple scales.

- The study of geometric algorithms which interpret the multi-scale distortion and reconstruct the 3D shape of the textured object. A range of 3D reconstruction algorithms have been used to solve the SFT problem at different levels of complexity: in the planar case, in the case of a surface displaying a frontal point and in the case of general shapes. Both the easier orthographic and the more general perspective cases have been considered, and the advantage of using redundant LSFs has been evaluated.

- The application of the multi-scale distortion and the geometric relations to texture mapping and replacement. An original approach has been implemented to recover the relevant geometric mapping in texturing and re-retexturing by using the multiple LSF estimates and the equations relating the surface texture and image plane coordinates.

The implemented algorithms have been tested on semi-synthetic and real images, the results have been compared and discussed, and the accuracy and robustness has been proved with respect to relevant work in the SFT literature.

8.2 Summary of Work

Shape-From-Texture is generally about estimating the texture distortion in an image and then reconstructing the surface 3D coordinates in the scene.

An initial assumption is always needed to identify the characteristic of the texture conveying the deformation, and a restrictive one can preclude applicability to natural images. We have discarded isotropy and the use of texels because only a few natural images are isotropic or show identifiable texels. Furthermore we have noted that the classic homogeneity and the use of texture gradients is sometimes difficult to apply, especially in the case of stochastic textures. We have adopted here the 'stochastic' homogeneity introduced by Malik and Rosenholtz. The chosen assumption requires the 'same' texture at different points of the surface in the scene. For deterministic texture this can be seen as periodicity, for stochastic textures this implies that the stochastic process which generates the texture is stationary under translation.

When possible we have considered both the orthographic and perspective camera models. The former usually suffices to describe the majority of objects that one may consider for SFT and implies simpler mathematical equations. The latter is more general, has given better results when perspective effects are evident in the observed image and is required to study planar surfaces, as homogeneity provides no information (i.e. is preserved) under an orthographic projection.

We have characterised the multi-scale distortion estimation by using *local spatial frequencies*. A novel method has been introduced which combines Fourier analysis and the use of Gabor functions to recover all the main instantaneous frequencies of the texture at each pixel. In particular the Fourier transform of the image has been shown to provide valuable information about the number and range of LSFs contained in the texture: the analysis of interest areas in the modulus of the Fourier transform provides fast setup and optimised computational expense for the subsequent use of selected Gabor filters. Local frequencies at each significant scale are in fact obtained by using the method of Super and Bovik [158] with the selected Gabor filters and their first derivatives. Our novel method has been shown to have a number of advantages over the present distortion estimation algorithms: it uses frequencies and therefore automatically normalizes for slow variations of illumination, shading and surface texture contrast; it uses the multi-scale nature of textures, while most related works only estimate a single distortion measure and only a few consider two preferred directions in the spectral domain.

We have used the multi-scale distortion estimation to recover the 3D shape of the textured scene at different levels of shape complexity.

Planar surfaces. Each LSF estimate has been processed using an extensive search over the pose parameters (σ,τ) (i.e. using the method proposed by Super and Bovik in [158]), and a novel method has been used to choose the best orientation estimation according to the residual variance. We have tested the algorithm on a wide range of semi-synthetic and real images, and compared the results with those of Hwang et al. [74]. Our algorithm was more accurate, and proved robust against simulated Gaussian noise, variations in illumination, self shadowing, imperfections in the texture pattern, creases and partial occlusions.

General surfaces with available frontal views. We have presented two algorithms for considering single LSFs and pairs of LSFs, combining the results according to the residual variance to select the best shape reconstruction. The former is a novel algorithm based on the closed-form solution for the value of the local orientation (σ,τ) at a pixel from a neighbouring pixel. The latter is based on *singular value decomposition* and draws on the algorithm of Dugué and Elghadi [42], which we extended to perspective cases. Both algorithms gave an accuracy in line with the state of the art of SFT on semi-synthetic and real images. In particular, to test the algorithms on real non-planar images of known ground truth, we have acquired textured objects on cylinders with known diameters, which fills a gap in the current SFT literature. Both algorithms have been shown to be robust against variations in illumination, self shadowing, reflections and imperfections in the texture pattern. In addition the second algorithm has been shown to be applicable to non-developable shapes, thanks to the use of redundant LSFs and SVD.

General surfaces locally. *Local frequency variations* and *eigendecomposition* have been used to process each pair of LSFs from the multi-scale distortion, selecting the best reconstruction according to the residual variance. The novel algorithm does not require any further assumptions and recovers the local orientation of general shapes, developable and non-developable, planar and curved, small or extended. It proved robust against variations in illumination, self shadowing, reflections and imperfections in the texture pattern. The algorithm also gave a better accuracy and applicability than the 3D reconstruction technique of Massot and Hérault [121], from which we drew inspiration. In addition the results have been compared with algorithms which make assumptions about the underlying shape of the texture image (i.e. available frontal view), and slight differences in accuracy in some cases have been discussed.

The determined multi-scale distortion estimation based on LSFs and the study of the geometric relations between the surface texture and the image plane have been applied to the *computer graphics* fields of *texture mapping* and *texture replacement*. In particular, the geometric mapping was solved for by using the defined viewing geometry and projection model. The geometric mapping, combined with a filtering based on a simple linear interpolation, gave visually pleasing

texturing results even in cases of non-developable shapes. The algorithm was used to replace homogeneous textures once the shape was recovered. Alternatively, a novel re-texturing algorithm has also been introduced which uses the estimated LSFs and does not require a reconstruction of the 3D shape. The re-texturing algorithm accurately replaced the original textures, and the results were consistent with those from our texturing algorithm. The re-texturing technique has been shown to be a new approach to the problem, while the consequences of the availability or non-availability of a frontal view have been discussed.

8.3 Directions for Future Work

We plan to concentrate future investigations on the two following aspects:

- SFT in a controlled environment;
- SFT for natural scenes.

The former direction refers to the use of SFT for non-invasively recovering the shape of objects by laying a textured film on them. The task easily extends to studying the shape changes over time of textured objects, by analysing video sequences. In fact one such case was examined in the results of chapters 5 and 6, where we considered a textured cloth laid over a deflating balloon.

We imagine that the challenge would be the level of accuracy that one can achieve. The textured film can be chosen to be as simple as that given by two superposed cosines 90° apart in the spectral domain. Such a texture would ensure a very easy setup due to maximum separation of the two very compact signals. The challenge would then translate into the following investigations.

- evaluation of the accuracy of the LSF estimations in the case of simple textures, and comparison with distortion estimation methods which use different filters (e.g. Log-normal filters, as in the work of Massot and Hérault);
- evaluation of the accuracy of the available 3D reconstruction algorithms in the case of low noise in the LSF estimation. This would imply solving the observed tilt-ambiguity to generalise the algorithms of chapter 5 to more

complex surfaces, and a better understanding of the first order approximations used in the differentiation in chapter 6

On the other hand, SFT for natural scenes would be more concerned with the challenge of applicability. Further investigations would ideally aim to recover, up to a certain level of accuracy, the 3D structure of, for example, a dry dam from a photograph showing the cracks in the earth, a sandy beach from its dunes, a wooded hill from an aerial view of the top of the trees. The 3D structure of natural scenes would then interest a wide range of scientists: from geographers, concerned with the earth shape and changes, to post-production engineers in the movie industry, who may want to modify and re-texture the landscape in a film.

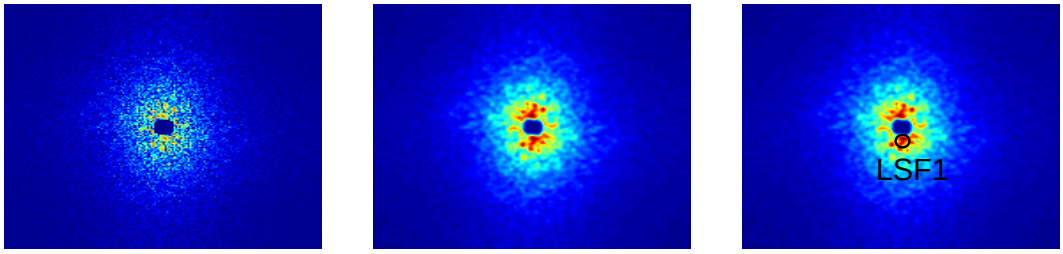
For this second direction, relevant investigations would focus on estimating the multi-scale distortion of a wide range of stochastic textures, evaluating the validity of the proposed method when the considered areas of the spectrum amplitude may not be well defined, nor the LSFs may be uniquely identifiable. At the same time the validity of the available 3D reconstruction algorithms in the presence of relevant levels of noise would need to be investigated for possible improvements.

Interesting issues would also be posed by the use of colour images or by considering the impact of image resolution on our algorithm. In particular, we believe that the combination of a convenient colour model and a probabilistic framework (for combining the multi-channel colour information) could in principle improve the LSF estimation.

Figure 8.1 offers an example of a natural scene and of some of the problems that may need further study. It is the example of the dry dam, which we reproduce from figure 1.1. Figure 8.2 shows the amplitude spectrum of the image. It is not an easy task to find distinct interest areas in the rich spectrum. However the spectrum becomes more informative after smoothing, as shown in figure 8.2, and an approximate area can be detected.

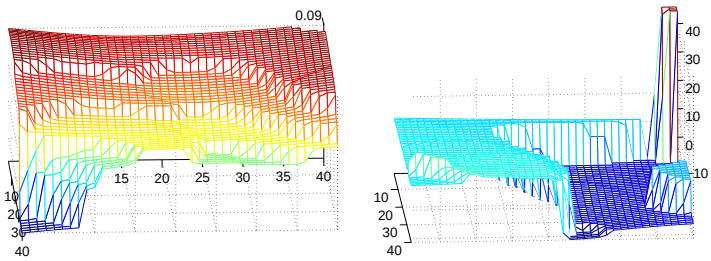
An estimate of LSF is obtained with Gabor filters set up on the approximate area. The modulus and phase of the LSF, represented in figure 8.3, look consistent with a LSF with spectral direction along the vertical axis, although noisy. Processing the LSF with the single-LSF-version of the algorithm in chapter 6

Figure 8.1: 240x240 real image of a dry dam



(a) (b) (c)

Figure 8.2: (a) Spectrum of the image in figure 8.1; (b) smoothed spectrum, spatial constant of smoothing Gaussian is $\gamma = 1$; (c) smoothed spectrum in (b) labeled to indicate the interest area for LSF1



(a) LSF 1 - Modulus (b) LSF 1 - Phase

Figure 8.3: Modulus and phase of the estimated LSF of the image in figure 8.1

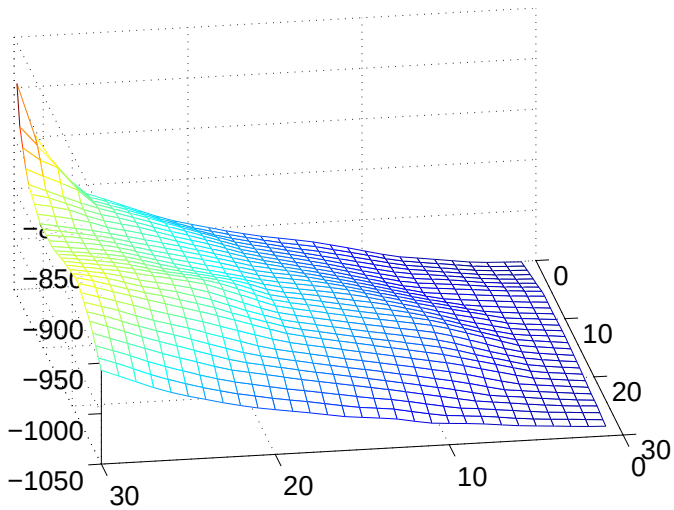


Figure 8.4: 3D reconstruction of the image in figure 8.1, seen from the side

(equations 6.50 and 6.51 or equivalently the 3D reconstruction method of [121]) we obtain the 3D shape illustrated in figure 8.4.

Future investigations would aim to recover at least two LSFs from the image, because, as discussed in chapters 5 and 6, two LSFs enable robust analysis of general planar and non-planar surfaces. Although the task may seem difficult, we note that a frontal view of the same dry dam does not have, in theory, preferred areas of interest in the spectrum. However the spectrum points out the detected area in the direction of the tilt for the slanted image. Accordingly, we could expect more interest areas along the main tilt directions for more complex shapes, but experiments, in this respect, still have to be done.

Interestingly, in the considered case the reconstruction has been achieved locally without any planarity assumption. However, it nonetheless shows a convincing planar shape, oriented on average along the 89.4° tilt direction with a slant angle of 55°.

On another note, it would certainly be of interest to work further on the texturing and re-texturing algorithms of chapter 7. In both cases, the filtering still needs to be implemented, to solve for the aliasing problem. Furthermore, the algorithms need to be tested on more challenging shapes and new ideas need to be introduced to solve, for example, for cases of missing one-to-one correspondences (e.g. shapes with holes). The use of LSFs, with their inherent robustness to issues of illumination, shading and contrast, looks, however, very promising.

Appendix A

Two other Methods to Solve for the Local Orientation from Local Frequency Variations

This appendix illustrates two methods to solve for the value of the slant σ and tilt τ at a given image point $\mathbf{x}_i$, in terms of the local values of two LSFs (expressed by U_i) and of their local variations (expressed by $\partial_{\mathbf{s}} U_i$).

The two methods solve equation 6.11 and give alternative solutions to those presented in sections 6.1.2.1 and 6.1.2.2 (which solve equation 6.15).

As already remarked in section 6.1.2.3, these alternative solutions coincide with those in sections 6.1.2.1 and 6.1.2.2.

A.1 Solving for Slant and Tilt Locally Using Ratios of Determinants

The variable r, used in the calculations in sections 6.1.2.1 and 6.1.2.2, is the absolute value of the determinant of the image plane frequency matrix U_i:

$$r = |\det U_i| \ . \tag{A.1}$$

We define

$$p = \det U_i \tag{A.2}$$

the signed determinant of the matrix U_i.

Let us consider equation 6.11, which is re-written below:

$$\begin{aligned} U_i = & \frac{z_0 \cos\sigma}{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma} \\ & \left\{ -\frac{\sin\sigma \begin{bmatrix} \cos\tau \\ \sin\tau \end{bmatrix} \begin{bmatrix} x_i & y_i \end{bmatrix}}{\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma} + I \right\} R(\tau) P^{-1}(\sigma) R(-\tau) U_s \ . \end{aligned} \tag{A.3}$$

Computing the determinant on both sides of equation 6.11 (A.3), we have

$$p = \frac{z_0^2 \cos^2(\sigma) f p_s}{[\sin\sigma(x_i \cos\tau + y_i \sin\tau) + f\cos\sigma]^3} \ , \tag{A.4}$$

where we set $p_s = \det U_s$, constant over the image according to the homogeneity assumption of section 2.2 and the alignment of section 5.3.1.

Let us consider the ratio

$$\frac{p(\mathbf{x}_i)}{p(\mathbf{x}_i + \mathbf{s})} \ , \tag{A.5}$$

where, as for equation 6.36, $\mathbf{s}$ is the increment vector between the two points $\mathbf{x}_i = (x_i, y_i)$ and $(\mathbf{x}_i + \mathbf{s}) = (x_i + s\cos\phi, y_i + s\sin\phi)$, with $\phi = \angle\mathbf{s}$ representing the increment direction and $s = |\mathbf{s}|$ representing the spatial shift, i.e. the distance between the considered image plane coordinate points. The expression for $\mathbf{s}$ was illustrated in equation 6.37, and is repeated below:

$$\mathbf{s} = |\mathbf{s}| \begin{pmatrix} \cos(\angle\mathbf{s}) \\ \sin(\angle\mathbf{s}) \end{pmatrix} = \begin{pmatrix} s\cos\phi \\ s\sin\phi \end{pmatrix} \ . \tag{A.6}$$

By substituting for p in expression A.5 we get to the same equation as in 6.39, but expressed in terms of p instead of r (i.e. in terms of the signed determinant $\det U_i$ instead of $|\det U_i|$):

$$\sqrt[3]{\frac{p(\mathbf{x}_i)}{p(\mathbf{x}_i+\mathbf{s})}}-1=\frac{\sin\sigma|\mathbf{s}|(\cos\phi\cos\tau+\sin\phi\sin\tau)}{\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma}$$
$$=\frac{\sin\sigma\ \ \mathbf{s}\cdot\hat{\tau}}{\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma}\ . \quad \text{(A.7)}$$

Equation A.7 can then be solved for the orientation parameters (σ,τ) at each point of the image plane as previously done with equations 6.27 and 6.39.

We consider the minimum of the left hand side of the equation. First of all we can assume that the signs of $p(\mathbf{x}_i)$ and $p(\mathbf{x}_i+\mathbf{s})$ are the same. In fact, as $\mathbf{s}$ is an arbitrary increment, its modulus can always be chosen small enough to satisfy this assumption. Then, using a first order approximation for $p(\mathbf{x}_i+\mathbf{s})$ we write

$$\sqrt[3]{\frac{p(\mathbf{x}_i)}{p(\mathbf{x}_i+\mathbf{s})}}-1=\sqrt[3]{\frac{p(\mathbf{x}_i)}{p(\mathbf{x}_i)+\nabla p\cdot\mathbf{s}}}-1\ . \quad \text{(A.8)}$$

The minimum of the left hand side of equation A.7 is then given by choosing the increment direction $\angle\mathbf{s}$ aligned with the direction of the gradient of p (i.e. $\angle\mathbf{s}=\angle\nabla p$), when the value of p computed at the coordinate $\mathbf{x}_i$ is positive ($p(\mathbf{x}_i)\geq 0$), or in the opposite direction (i.e. $\angle\mathbf{s}=\angle\nabla p+\pi$), otherwise ($p(\mathbf{x}_i)<0$). The value of the minimum is given by

$$\min_{\mathbf{s}}\left(\sqrt[3]{\frac{p(\mathbf{x}_i)}{p(\mathbf{x}_i)+\nabla p\cdot\mathbf{s}}}-1\right)=\sqrt[3]{\frac{|p(\mathbf{x}_i)|}{|p(\mathbf{x}_i)|+|\nabla p|s}}-1\ , \quad \text{(A.9)}$$

and is obtained when

$$\angle(\mathbf{s})=\begin{cases}\angle(\nabla p) & \text{when } p(\mathbf{x}_i)\geq 0\\ \angle(\nabla p)+\pi & \text{when } p(\mathbf{x}_i)<0\end{cases}\ . \quad \text{(A.10)}$$

We now write the minimum in A.9 as the variable a_i:

$$a_i=\sqrt[3]{\frac{|p(\mathbf{x}_i)|}{|p(\mathbf{x}_i)|+|\nabla p|s}}-1\ . \quad \text{(A.11)}$$

The minimum value of the right hand side of A.7 is determined according to the sign of the expression

$$\frac{\sin\sigma}{\sin\sigma(x_i\cos\tau + y_i\sin\tau) + f\cos\sigma} \quad : \tag{A.12}$$

- when A.12 is negative, the right hand side of A.7 is minimum for the tilt direction aligned with the direction of differentiation ($\tau = \angle\mathbf{s}$);
- when A.12 is positive, the right hand side of A.7 is minimum for the tilt direction opposite to the direction of differentiation ($\tau = \angle\mathbf{s} + \pi$).

In our coordinate system the first is usually the case, so we first assume expression A.12 has negative sign and we solve for (σ, τ). Using this assumption and equating the minima of the right and left hand sides of equation A.7, the increment direction is the same as the direction of ∇p modulo π ($\angle\mathbf{s} = \angle\nabla p$ when $p(\mathbf{x}_i) \geq 0$, $\angle\mathbf{s} = \angle\nabla p + \pi$ when $p(\mathbf{x}_i) < 0$), the tilt direction is aligned with $\mathbf{s}$ ($\tau = \angle\mathbf{s}$) and equation A.7 becomes

$$a_i = \frac{s\sin\sigma}{\sin\sigma(x_i\cos\tau + y_i\sin\tau) + f\cos\sigma}\ , \tag{A.13}$$

from which we get the solutions for (σ, τ):

$$\tau = \begin{cases} \angle(\nabla p) & \text{when } p(\mathbf{x}_i) \geq 0 \\ \angle(\nabla p) + \pi & \text{when } p(\mathbf{x}_i) < 0 \end{cases} \tag{A.14}$$

$$\sigma = \arctan\frac{a_i f}{s - a_i(x_i\cos\tau + y_i\sin\tau)}\ . \tag{A.15}$$

In the other case, when the expression A.12 is positive, by equating the minima of the right and left sides of equation A.7, we have again the increment direction $\angle\mathbf{s}$ aligned with the gradient of p modulo π ($\angle\mathbf{s} = \angle\nabla p$ when $p(\mathbf{x}_i) \geq 0$, $\angle\mathbf{s} = \angle\nabla p + \pi$ when $p(\mathbf{x}_i) < 0$). However, in this case, the tilt direction is opposite to the direction of $\mathbf{s}$ ($\tau = \angle\mathbf{s} + \pi$) and equation A.7 becomes

$$a_i = \frac{(-s)\sin\sigma}{\sin\sigma(x_i\cos\tau + y_i\sin\tau) + f\cos\sigma}\ , \tag{A.16}$$

from which we obtain the corresponding solution

$$\tau = \begin{cases} \angle(\nabla p) + \pi & \text{when } p(\mathbf{x}_i) \geq 0 \\ \angle(\nabla p) & \text{when } p(\mathbf{x}_i) < 0 \end{cases} \tag{A.17}$$

$$\sigma = \arctan\frac{-a_i f}{s + a_i(x_i\cos\tau + y_i\sin\tau)}\ . \tag{A.18}$$

In this second case the vector $\mathbf{s}$ is the same, while the tilt is in the opposite direction, i.e. τ is shifted by π radians. This implies that $\cos\tau$, $\sin\tau$ and $(x_i\cos\tau + y_i\sin\tau)$ all change sign. As a result, the sign of the σ computed in A.18 is opposite to the sign of the σ computed in A.15. However in our coordinate system, shifting τ by π radians and computing $-\sigma$ means getting the same orientation. The two hypotheses therefore produce the same solution:

$$\tau = \begin{cases} \angle(\nabla p) & \text{when } p(\mathbf{x}_i) \geq 0 \\ \angle(\nabla p) + \pi & \text{when } p(\mathbf{x}_i) < 0 \end{cases} \tag{A.19}$$

$$\sigma = \arctan \frac{a_i f}{s - a_i(x_i\cos\tau + y_i\sin\tau)} \,. \tag{A.20}$$

A.2 Solving for Slant and Tilt Locally Using Differences of Determinants

Let us again consider equation A.4, which we repeat below:

$$p = \frac{z_0^2\cos^2(\sigma) f p_s}{\left[\sin\sigma(x_i\cos\tau + y_i\sin\tau) + f\cos\sigma\right]^3} \,. \tag{A.21}$$

Instead of computing the ratio $\frac{p(\mathbf{x}_i)}{p(\mathbf{x}_i+\mathbf{s})}$ as in A.5, we consider here the difference $p(\mathbf{x}_i + \mathbf{s}) - p(\mathbf{x}_i)$:

$$\frac{p(\mathbf{x}_i+\mathbf{s}) - p(\mathbf{x}_i)}{z_0^2\cos^2(\sigma) f p_s} = \frac{1}{\left[\sin\sigma(x_i\cos\tau + y_i\sin\tau) + f\cos\sigma + \sin\sigma|\mathbf{s}|(\cos\phi\cos\tau + \sin\phi\sin\tau)\right]^3} - \frac{1}{\left[\sin\sigma(x_i\cos\tau + y_i\sin\tau) + f\cos\sigma\right]^3} \,, \tag{A.22}$$

where, as for equations 6.36 and A.5, $\mathbf{s}$ is the increment vector between the two points $\mathbf{x}_i = (x_i, y_i)$ and $(\mathbf{x}_i + \mathbf{s}) = (x_i + s\cos\phi, y_i + s\sin\phi)$, with $\phi = \angle\mathbf{s}$ representing the increment direction and $s = |\mathbf{s}|$ representing the spatial shift, i.e. the distance between the considered image plane coordinate points. The expression for $\mathbf{s}$ was illustrated in equations 6.37 and A.6.

By substituting for p_s from equation A.4 (A.21) into equation A.22 we get

$$\frac{p(\mathbf{x}_i+\mathbf{s})-p(\mathbf{x}_i)}{p(\mathbf{x}_i)}+1 = \frac{[\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma]^3}{[\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma+\sin\sigma|\mathbf{s}|(\cos\phi\cos\tau+\sin\phi\sin\tau)]^3}\,, \tag{A.23}$$

and by taking the cubic root

$$\sqrt[3]{\frac{p(\mathbf{x}_i+\mathbf{s})-p(\mathbf{x}_i)}{p(\mathbf{x}_i)}+1} = \frac{\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma}{\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma+\sin\sigma|\mathbf{s}|(\cos\phi\cos\tau+\sin\phi\sin\tau)}\,. \tag{A.24}$$

Considering a first order approximation, we can write the left hand side of equation A.24 as

$$\sqrt[3]{\frac{p(\mathbf{x}_i+\mathbf{s})-p(\mathbf{x}_i)}{p(\mathbf{x}_i)}+1} = \sqrt[3]{\frac{\nabla p\cdot\mathbf{s}}{p(\mathbf{x}_i)}+1}\,. \tag{A.25}$$

The maximum of the left hand side of equation A.24 is therefore given by choosing the direction of the increment vector $\mathbf{s}$ parallel to the gradient of p ($\angle\mathbf{s} = \angle\nabla p$), when the value of p computed at the coordinate $\mathbf{x}_i$ is positive ($p(\mathbf{x}_i)\geq 0$), or in the opposite direction ($\angle\mathbf{s} = \angle\nabla p+\pi$) otherwise ($p(\mathbf{x}_i)<0$). The value of the maximum is given by

$$\max_{\mathbf{s}}\left(\sqrt[3]{\frac{\nabla p\cdot\mathbf{s}}{p(\mathbf{x}_i)}+1}\right) = \sqrt[3]{\frac{|\nabla p|s}{|p(\mathbf{x}_i)|}+1}\,, \tag{A.26}$$

and is obtained when

$$\angle(\mathbf{s}) = \begin{cases} \angle(\nabla p) & \text{when } p(\mathbf{x}_i)\geq 0 \\ \angle(\nabla p)+\pi & \text{when } p(\mathbf{x}_i)<0 \end{cases}\,. \tag{A.27}$$

We again write the above maximum as a_i:

$$a_i = \sqrt[3]{\frac{|\nabla p|s}{|p(\mathbf{x}_i)|}+1}\,. \tag{A.28}$$

We then observe that the maximum of the right hand side of equation A.24 is

$$\max_{\tau}\left[\frac{\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma}{\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma+\sin\sigma|\mathbf{s}|(\cos\phi\cos\tau+\sin\phi\sin\tau)}\right]$$
$$=\left.\frac{|\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma|}{|\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma|-s\,|\sin\sigma|}\right|_{\tau=\phi+k\pi,\,k\in\mathbb{Z}}, \quad \text{(A.29)}$$

which is therefore obtained when τ is in the direction of $\mathbf{s}$ or in the opposite direction, as detailed below.

In our coordinate system $[\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma]$ is usually negative while $[\sin\sigma]$ is usually positive. Having assumed this, by equating the maxima of the left and right hand sides of equation A.24 we get

$$a_i=\frac{\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma}{\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma+s\,\sin\sigma}\,, \quad \text{(A.30)}$$

which is in general the form of equation A.24 when $[\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma]$ and $[\sin\sigma]$ have opposite signs. Comparing the above equation A.30 with equation A.24, we see that it is obtained when τ is aligned with $\mathbf{s}$ ($\tau=\angle\mathbf{s}$). Hence the solution in this case is given by

$$\tau = \begin{cases}\angle(\nabla p) & \text{when } p(\mathbf{x}_i)\geq 0\\ \angle(\nabla p)+\pi & \text{when } p(\mathbf{x}_i)<0\end{cases} \quad \text{(A.31)}$$

$$\sigma = \arctan\frac{(1-a_i)f}{s\,a_i+(a_i-1)(x_i\cos\tau+y_i\sin\tau)}\,. \quad \text{(A.32)}$$

Alternatively, when $[\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma]$ and $[\sin\sigma]$ have the same sign, the result of equating the maxima of the left and right hand sides of equation A.24, i.e. the corresponding form of equation A.30, is

$$a_i=\frac{\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma}{\sin\sigma(x_i\cos\tau+y_i\sin\tau)+f\cos\sigma-s\,\sin\sigma}\,. \quad \text{(A.33)}$$

Again, by comparing the above equation A.33 with equation A.24 we see that it is obtained when τ is aligned with the direction opposite to $\mathbf{s}$ ($\tau=\angle\mathbf{s}+\pi$). Hence the solution in this case is given by

$$\tau = \begin{cases}\angle(\nabla p)+\pi & \text{when } p(\mathbf{x}_i)\geq 0\\ \angle(\nabla p) & \text{when } p(\mathbf{x}_i)<0\end{cases} \quad \text{(A.34)}$$

$$\sigma = \arctan\frac{(1-a_i)f}{-s\,a_i+(a_i-1)(x_i\cos\tau+y_i\sin\tau)}\,. \quad \text{(A.35)}$$

Comparing this case (same signs) with the previous case (opposite signs), we note that the tilt is in the opposite direction, i.e. τ is shifted by π radians. This implies that $\cos\tau$, $\sin\tau$ and $(x_i \cos\tau + y_i \sin\tau)$ all change sign. As a result, the sign of the σ computed in A.35 is opposite to the sign of the σ computed in A.32. However in our coordinate system, shifting τ by π radians and computing $-\sigma$ means getting the same orientation. The two hypotheses therefore produce the same solution:

$$\tau = \begin{cases} \angle(\nabla p) & \text{when } p(\mathbf{x}_i) \geq 0 \\ \angle(\nabla p) + \pi & \text{when } p(\mathbf{x}_i) < 0 \end{cases} \tag{A.36}$$

$$\sigma = \arctan \frac{(1 - a_i) f}{s\, a_i + (a_i - 1)(x_i \cos\tau + y_i \sin\tau)} \,. \tag{A.37}$$

References

[1] Yiannis Aloimonos. Shape from texture. *Biological Cybernetics*, **58**(5):345–360, 1988. 16, 20, 52, 114

[2] Yiannis Aloimonos and Michael J Swain. Shape from patterns: Regularization. *International Journal of Computer Vision*, **2**(2):171–187, 1988. 15

[3] George J Andersen, Myron L Braunstein, and Asad Saidpour. The perception of depth and slant from texture in three-dimensional scenes. *Perception*, **27**(9):1087–1106, 1998. 16

[4] Masayoshi Aoki and Martin D. Levine. Computer generation of realistic pictures. *Computers & Graphics*, **3**(4):149–161, 1978. 132

[5] Joseph J. Atick and A. Norman Redlich. What does the retina know about natural scenes? *Neural Computation*, **4**(2):196–210, 1992. 28, 41

[6] R. Bajcsy and L. Lieberman. Texture gradient as a depth cue. *Computer Graphics and Image Processing*, **5**(1):52–67, 1976. 11, 21

[7] M. Bichsel and A. P. Pentland. A simple algorithm for shape from shading. In *Computer Vision and Pattern Recognition*, pages 459–465, 1992. 18

[8] Eric Bier and Kenneth Sloan. Two-part texture mappings. *IEEE Computer Graphics and Applications*, **6**(9):40–53, 1986. 131

[9] A Blake and C Marinos. Shape from texture: estimation, isotropy and moments. *Artificial Intelligence*, **45**:323–380, 1990. 9, 20, 65

[10] Andrew Blake, Heinrich H. Bülthoff, and David Sheinberg. Shape from texture: Ideal observers and human psychophysics. *Vision Research*, **33**(12):1723–1737, 1993. 10, 16, 70

[11] James F. Blinn. Simulation of wrinkled surfaces. *SIGGRAPH Computer Graphics*, **12**(3):286–292, 1978. 132

[12] James F. Blinn and Martin E. Newell. Texture and reflection in computer generated images. *Communications of the ACM*, **19**(10):542–547, 1976. 131

[13] D. Blostein and N. Ahuja. Shape from texture: Integrating texture-element extraction and surface estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **11**(12):1233–1251, 1989. 10, 20, 52

[14] Adrian G. Bors, Edwin R. Hancock, and Richard C. Wilson. Terrain analysis using radar shape-from-shading. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **25**(8), 2003. 18

[15] B. Boufama, R. Mohr, and F. Veillon. Euclidean constraints for uncalibrated reconstruction. In *International Conference on Computer Vision*, pages 466–470, May 1993. 1

[16] Alan C. Bovik, Joseph P. Havlicek, and Mita D. Desai. Theorems for discrete filtered modulated signals. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 153–156, Minneapolis, MN, apr 1993. 27

[17] C. Braccini and G. Marino. Fast geometrical manipulations of digital images. *Computer Graphics and Image Processing*, **13**(2):127–141, jun 1980. 132

[18] Phil Brodatz. *Textures: A Photographic Album for Artists and Designers*. Dover, New York, 1966. xiv, 36, 43, 57, 58, 76, 140, 148

[19] Michael J. Brooks and W. Chojnacki. Direct computation of shape from shading. *International Conference on Pattern Recognition*, **1**:114–119 vol.1, Oct 1994. 1

[20] Michael J. Brooks and Berthold K.P. Horn. Shape and source from shading. In *International Joint Conference on Artificial Intelligence*, pages 98–101, Los Angeles, U.S.A., aug 1985. 1

[21] L. G. Brown and H. Shvaytser. Surface orientation from projective foreshortening of isotropic texture autocorrelation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **12**(6):584–588, June 1990. 20

[22] D Buckley, J P Frisby, and E Spivey. Stereo and texture cue combination in ground planes: an investigation using the table stereometer. *Perception*, **20**:91, 1991. 3

[23] Mario Castelán and Edwin R. Hancock. Imposing integrability in geometric shape-from-shading. In *Iberoamerican Congress on Pattern Recognition*, pages 196–203, 2003. 18

[24] Ed Catmull and Alvy Ray Smith. 3-d transformations of images in scanline order. *SIGGRAPH Computer Graphics*, **14**(3):279–285, 1980. 132

[25] Edwin Catmull. Computer display of curved surfaces. In *IEEE Conference on Computer Graphics, Pattern Recognition, and Data Structures*, pages 11–17, Los Angeles, USA, May 1975. 130

[26] Edwin Earl Catmull. *A Subdivision Algorithm for Computer Display of Curved Surfaces*. PhD thesis, Dept of CS, University of Utah, December 1974. 130

[27] M J Chantler. Why illuminant direction is fundamental to texture analysis. *IEE Proceedings: Vision, Image and Signal Processing*, **142**(4):199–206, 1995. 19

[28] Y. Choe and R. L. Kashyap. Shape from texture and shaded surface. In *International Conference on Pattern Recognition*, **1**, pages 294–296, 1990. 15

[29] T. A. C. M. Claasen and W. F. G. Mecklenbrauker. The wigner distribution - a tool for time-frequency signal analysis. *Philips Journal of Research*, **35**(3):217–250, 1980. 25

[30] Maureen Clerc and Stéphane Mallat. The texture gradient equation for recovering shape from texture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24**(4):536–549, 2002. 3, 11, 16, 19, 22

[31] E. North Coleman, Jr. and Ramesh Jain. Obtaining 3-dimensional shape of textured and specular surfaces using four-source photometry. *Computer Graphics and Image Processing*, **18**(4):309–328, apr 1982. 18

[32] Robert L. Cook. Shade trees. *SIGGRAPH Computer Graphics*, **18**(3):223–231, 1984. 132

[33] Robert L. Cook. Stochastic sampling in computer graphics. *ACM Transactions on Graphics*, **5**(1):51–72, 1986. 133

[34] Antonio Criminisi and Andrew Zisserman. Shape from texture: Homogeneity revisited. In *British Machine Vision Conference*, pages 82–91, 2000. 16, 52, 70

[35] Franklin C. Crow. The aliasing problem in computer-generated shaded images. *Communications of the ACM*, **20**(11):799–805, 1977. 133

[36] James Edwin Cryer, Ping-Sing Tsai, and Mubarak Shah. Integration of shape from x modules: Combining stereo and shading. In *Computer Vision and Pattern Recognition*, pages 720–721, 1993. 1

[37] James E. Cutting and Robert T. Millard. Three gradients and the perception of flat and curved surfaces. *Journal of Experimental Psychology: General*, **113**(2):198 – 216, 1984. 3, 10

[38] I. Daubechies. Orthonormal bases of compactly supported wavelets. *Communications on Pure and Applied Mathematics*, **41**:909–996, 1988. 25

[39] M. Daum and G. Dudek. On 3-d surface reconstruction using shape from shadows. In *Computer Vision and Pattern Recognition*, pages 461–468, Washington, DC, USA, 1998. IEEE Computer Society. 1

[40] Jean-Michel Dischler and Djamchid Ghazanfarpour. A survey of 3d texturing. *Computers & Graphics*, **25**(1):135–151, 2001. 132

[41] Jean-Michel Dischler, Djamchid Ghazanfarpour, and R. Freydier. Anisotropic solid texture synthesis using orthogonal 2d views. *Computer Graphics Forum, Proceedings of Eurographics*, **17**(3):87–96, 1998. 132

[42] Anne Guérin Dugué and Mohammed Elghadi. Shape from texture by local frequencies estimation. In *Scandinavian Conference on Image Analysis*, pages 533–544, Kangerlussuaq, Greenland, June 1999. 3, 5, 11, 13, 21, 22, 65, 66, 84, 85, 86, 87, 88, 89, 92, 101, 121, 136, 160

[43] Paul Dupuis and John Oliensis. Direct method for reconstructing shape from shading. In *Computer Vision and Pattern Recognition*, pages 453–458, Champaign, IL, USA, 1992. 18

[44] Alexei A. Efros and Thomas K. Leung. Texture synthesis by non-parametric sampling. In *International Conference on Computer Vision*, pages 1033–1038, Washington, DC, USA, 1999. 133

[45] Hui Fang and John C. Hart. Textureshop: texture synthesis as a photograph editing tool. In *SIGGRAPH: ACM SIGGRAPH 2004 Papers*, pages 354–359, New York, NY, USA, 2004. ACM. 133, 134

[46] Hui Fang and John C. Hart. Rototexture: Automated tools for texturing raw video. *IEEE Transactions on Visualization and Computer Graphics*, **12**(6):1580–1589, 2006. 133, 134

[47] James D. Foley and Andries Van Dam. *Fundamentals of interactive computer graphics*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1982. 130

[48] David A. Forsyth. Shape from texture without boundaries. In *European Conference on Computer Vision*, pages 225–239, London, UK, 2002. Springer-Verlag. 10, 23, 65

[49] J. Francos and H. Permuter. Parametric estimation of the orientation of textured planar surfaces. *IEEE Transactions on Image Processing*, **10**(3):403–418, March 2001. 3

[50] Robert T. Frankot and Rama Chellappa. A method for enforcing integrability in shape from shading algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **10**(4):439–451, jul 1988. 18, 75, 76, 90, 91, 92, 114, 115, 133, 139, 140, 144, 146, 148, 151

[51] D. Gabor. Theory of communications. *J. Inst. Elec. Engrs*, **93**(26):429–457, November 1946. 25, 26

[52] Jonas Gårding. Shape from texture for smooth curved surfaces in perspective projection. *Journal of Mathematical Imaging and Vision*, **2**:327–350, 1992. xii, 3, 8, 15, 16, 20, 22, 68, 73, 113, 121

[53] Jonas Gårding. Shape from texture and contour by weak isotropy. *Artificial Intelligence*, **64**(2):243–297, 1993. 9, 20, 65

[54] Jonas Gårding and Tony Lindeberg. Direct computation of shape cues using scale-adapted spatial derivative operators. *International Journal of Computer Vision*, **17**(2):163–191, 1996. 11

[55] Geoffrey Y. Gardner. Simulation of natural scenes using textured quadric surfaces. *SIGGRAPH Computer Graphics*, **18**(3):11–20, 1984. 132

[56] Geoffrey Y. Gardner. Visual simulation of clouds. *SIGGRAPH Computer Graphics*, **19**(3):297–304, 1985. 132

[57] Shaheen Ghayourmanesh and Yun Zahng. Shape from shading of SAR imagery in fourier space. In *International Geoscience and Remote Sensing Symposium*, pages 835–837, jul 2007. 18

[58] D. Ghazanfarpour and J.M. Dischler. Spectral analysis for automatic 3-d texture generation. *Computers & Graphics*, **19**(3):413–422, May 1995. 132

[59] James J. Gibson. *The Perception of the Visual World*. Houghton Mifflin, Boston, Massachusetts, 1950. 7, 16, 20, 52, 114

[60] Thomas Greiner and Sukhendu Das. Recovering orientation of a textured planar surface using wavelet transform. In *Indian Conference on Computer Vision, Graphics and Image Processing*, Ahmedabad, India, 2002. 11, 52

[61] Branko Grnbaum and G C Shephard. *Tilings and Patterns*. W.H. Freeman and Company, New York, NY, USA, 1987. 134

[62] B.G. Gumming, E.B. Johnston, and A.J. Parker. Effects of different texture cues on curved surfaces viewed stereoscopically. *Vision Research*, **33**(5–6):827 – 838, 1993. 10

[63] Yanwen Guo, Hanqiu Sun, Qunsheng Peng, and Zhongding Jiang. Mesh-guided optimized retexturing for image and video. *IEEE Transactions on Visualization and Computer Graphics*, **14**(2):426–439, 2008. 133, 135

[64] M. Hatzitheodorou and J.R. Kender. An optimal algorithm for the derivation of shape from shadows. *Computer Vision and Pattern Recognition*, pages 486–491, Jun 1988. 1

[65] Joseph P. Havlicek, Alan C. Bovik, and Petros Maragos. Modulation models for image processing and wavelet-based image demodulation. In *Asilomar Conference on Signals, Systems and Computers*, pages 805–810, Pacific Grove, California, October 1992. 27

[66] Glenn Healey and Ramesh Jain. Depth recovery from surface normals. In *International Joint Conference on Pattern Recognition*, pages 894–896, Montreal, Canada, aug 1984. 18

[67] Paul S. Heckbert. Texture mapping polygons in perspective. Technical Memo No. 13, NYIT Computer Graphics Lab, apr 1983. 132

[68] Paul S. Heckbert. Survey of texture mapping. *IEEE Computer Graphics and Applications*, **6**:56–67, 1986. 131

[69] David J. Heeger and James R. Bergen. Pyramid-based texture analysis/synthesis. In *SIGGRAPH*, **29**, pages 229–238, New York, NY, USA, 1995. ACM. 132

[70] Berthold K. P. Horn. Height and gradient from shading. *International Journal Computer Vision*, **5**(1):37–75, 1990. 1, 18, 134

[71] Berthold K. P. Horn and Michael J. Brooks. The variational approach to shape from shading. *Computer Vision, Graphics and Image Processing*, **33**(2):174–208, feb 1986. 18

[72] Berthold K. P. Horn and Michael J. Brooks. *Shape from shading.* MIT Press, Cambridge, MA, USA, 1989. 1

[73] Jun-Wei Hsieh, Hong-Yuan Mark Liao, Ming-Tat Ko, and Kuo-Chin Fan. Wavelet-based shape from shading. *Graphical Models and Image Processing*, **57**(4):343 – 362, 1995. 18

[74] Wen-Liang Hwang, Chun-Shien Lu, and Pau-Choo Chung. Shape from texture: Estimation of planar surface orientation through the ridge surfaces of continuous wavelet transform. *IEEE Transactions on Image Processing*, **7**(5):773–780, May 1998. xx, 3, 11, 13, 21, 52, 57, 59, 60, 63, 159

[75] Ken ichi Kanatani. Structure and motion from optical flow under orthographic projection. *Computer Vision, Graphics and Image Processing*, **35**(2):181–199, 1986. 17

[76] Yu ichi Ohta, Kiyoshi Maenobu, and Toshiyuki Sakai. Obtaining surface orientation from texels under perspective projection. In *International Joint Conference on Artificial Intelligence*, pages 746–751, 1981. 20

[77] Katsushi Ikeuchi. Shape from regular patterns. *Artificial Intelligence*, **22**(1):49–75, 1984. 20

[78] J. Y. Jau and R. T. Chin. Shape from texture using the wigner distribution. *Graphical Model and Image Processing*, **52**(2):248–263, November 1990. 16

[79] J. T. Kajiya and T. L. Kay. Rendering fur with three dimensional textures. In *SIGGRAPH*, pages 271–280, New York, NY, USA, 1989. ACM. 132

[80] Kenichi Kanatani and Tsai-Chia Chou. Shape from texture: general principle. *Artificial Intelligence*, **38**(1):1–48, February 1989. 3, 20

[81] John R. Kender. Shape from texture: An aggregation transform that maps a class of textures into surface orientation. In *International Joint Conference on Artificial Intelligence*, pages 475–480, 1979. 20

[82] John R. Kender and Earl M. Smith. Shape from darkness. In *International Conference on Computer Vision*, pages 539–546, 1987. 1

[83] Erum Arif Khan, Erik Reinhard, Roland W. Fleming, and Heinrich H. Bülthoff. Image-based material editing. *ACM Transactions on Graphics*, **25**(3):654–663, 2006. 133, 134

[84] Reinhard Klette, Andreas Koschan, and Karsten Schluns. *Computer Vision: Three-Dimensional Data from Images*. Springer-Verlag Singapore Pte. Limited, 1998. 17

[85] Reinhard Klette and Karsten Schlüns. Height data from gradient fields. In *Machine Vision Applications, Architectures, and Systems Integration, SPIE*, pages 204–215, 1996. 17

[86] D. C. Knill. Discriminating planar surface slant from texture: Human and ideal observers compared. *Vision Research*, **38**(11):1683 – 1711, 1998. 10

[87] D. C. Knill. Ideal observer perturbation analysis reveals human strategies for inferring surface orientation from texture. *Vision Research*, **38**(17):2635 – 2656, 1998. 10

[88] D. C. Knill. Surface orientation from texture: Ideal observers, generic observers and the information content of texture cues. *Vision Research*, **38**(11):1655 – 1682, 1998. 10

[89] Jan J Koenderink and Sylvia C Pont. Irradiation direction from texture. *Journal of the Optical Society of America*, **20**(10):1875–1882, 2003. 19

[90] J.J. Koenderink. What does the occluding contour tell us about solid shape? *Perception*, **13**:321–330, 1984. 1

[91] Peter Kovesi. Shapelets correlated with surface normals produce surfaces. In *International Conference on Computer Vision*, pages 994–1001, Washington, DC, USA, 2005. IEEE Computer Society. 18

[92] Ryszard Kozera. An overview of the shape from shading problem. *Machine Graphics and Vision*, **(1/2) 7**:291–312, 1998. 1

[93] J. Krumm and S. A. Shafer. Texture segmentation and shape in the same image. In *International Conference on Computer Vision*, pages 121–127, Washington, DC, USA, 1995. IEEE Computer Society. 2

[94] John Krumm and Steven Shafer. A characterizable shape-from-texture algorithm using the spectrogram. In *Proceedings of the IEEE-SP International Symposium on Time-Frequency and Time-Scale Analysis*, pages 322 – 325, October 1994. 21

[95] Vivek Kwatra, Arno Schödl, Irfan Essa, Greg Turk, and Aaron Bobick. Graphcut textures: image and video synthesis using graph cuts. *ACM Transactions on Graphics*, **22**(3):277–286, 2003. 134

[96] K. M. Lee and C. C. J. Kuo. Direct shape from texture using a parametric surface model and an adaptive filtering technique. In *Computer Vision and Pattern Recognition*, pages 402–407, 1998. 23, 46

[97] Mark E. Lee, Richard A. Redner, and Samuel P. Uselton. Statistically optimized sampling for distributed ray tracing. *SIGGRAPH Computer Graphics*, **19**(3):61–68, 1985. 133

[98] Sylvie Lelandais, Laurent Boutté, and Justin Plantier. Shape from texture: Local scales and vanishing line computation to improve results for macrotextures. *International Journal of Image and Graphics*, **5**(2):329–350, 2005. 11, 22, 52

[99] Thomas Leung and Jitendra Malik. On perpendicular texture: Why do we see more flowers in the distance? In *Computer Vision and Pattern Recognition*, pages 807–813, Washington, DC, USA, 1997. 19

[100] Andrea Li and Qasim Zaidi. Perception of three-dimensional shape from texture is based on patterns of oriented energy. *Vision Research*, **40**(2):217 – 242, 2000. 23, 126

[101] Andrea Li and Qasim Zaidi. Erratum to information limitations in perception of shape from texture. *Vision Research*, **41**(22):2927 – 2942, 2001. 23, 126

[102] Andrea Li and Qasim Zaidi. Information limitations in perception of shape from texture. *Vision Research*, **41**(12):1519 – 1533, 2001. 23, 126

[103] Andrea Li and Qasim Zaidi. Veridicality of three-dimensional shape perception predicted from amplitude spectra of natural textures. *Journal of the Optical Society of America*, **18**(10):2430–2447, 2001. 23, 126

[104] Andrea Li and Qasim Zaidi. Observer strategies in perception of 3-d shape from isotropic textures: developable surfaces. *Vision Research*, **43**(26):2741–2758, 2003. 23, 127

[105] Andrea Li and Qasim Zaidi. Three-dimensional shape from non-homogeneous textures: carved and stretched surfaces. *Journal of Vision*, **4**(10):860–878, 2004. 23, 127

[106] Yin Li, Jian Sun, Chi-Keung Tang, and Heung-Yeung Shum. Lazy snapping. *ACM Transactions on Graphics*, **23**(3):303–308, 2004. 134

[107] P. L. Lions, E. Rouy, and A. Tourin. Shape-from-shading, viscosity solutions and edges. *Numerische Mathematik*, **64**(3):323–353, 1993. 19

[108] Yanxi Liu, Wen-Chieh Lin, and James Hays. Near-regular texture analysis and manipulation. In *SIGGRAPH: ACM SIGGRAPH 2004 Papers*, pages 368–376, New York, NY, USA, 2004. ACM. 133, 134

[109] Xavier Lladó, Maria Petrou, and Joan Martí. Surface texture recognition by surface rendering. *Journal of Optical Engineering*, **44**(3), 2005. 18

[110] Anthony Lobay and D. A. Forsyth. Shape from texture without boundaries. *International Journal of Computer Vision*, **67**(1):71–91, April 2006. 10, 17, 23, 65, 83, 98, 126

[111] Angeline M. Loh. *The Recovery of 3-D Structure Using Visual Texture Patterm*. PhD thesis, School of Computer Science and Software Engineering, The University of Western Australia, feb 2006. 1, 9

[112] Angeline M. Loh and Richard Hartley. Shape from non-homogeneous, non-stationary, anisotropic, perspective texture. In *British Machine Vision Conference*, pages 69–78, 2005. 3, 10, 19, 23, 65

[113] Angeline M. Loh and Peter Kovesi. Estimation of surface normal of a curved surface using texture. In *Digital Image Computing: Techniques and Applications*, pages 155–164, 2003. 65, 66

[114] Angeline M. Loh and Peter Kovesi. Shape from texture without estimating transformations. Technical report, UWA-CSSE-05-001, jul 2005. 11, 16, 22, 23, 25, 65, 66, 68, 69

[115] Angeline M. Loh and Andrew Zisserman. Estimating the affine transformation between textures. In *Digital Image Computing: Techniques and Applications*, page 67, Washington, DC, USA, 2005. IEEE Computer Society. 52

[116] David G. Lowe and T.O. Binford. The recovery of three-dimensional structure from image curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **7**(3):320–326, May 1985. 1

[117] Jitendra Malik and Ruth Rosenholtz. Recovering surface curvature and orientation from texture distortion: A least squares algorithm and sensitivity analysis. In *European Conference on Computer vision*, **1**, pages 351–364, Secaucus, NJ, USA, 1994. Springer-Verlag New York, Inc. 11, 12, 21, 22

[118] Jitendra Malik and Ruth Rosenholtz. Computing local surface orientation and shape from texture for curved surfaces. *International Journal of Computer Vision*, **23**(2):149–168, June 1997. 3, 10, 11, 12, 16, 20, 21, 22, 56, 72, 101

[119] C Marinos and A Blake. Shape from texture: The homogeneity hypothesis. In *Proceedings of the International Conference on Computer Vision*, pages 350–353, 1990. 9, 20, 114

[120] Corentin Massot. *Texture et Perception 3D dans les Scènes Naturelles: Modèles d'Inspiration Biologique et Expérimentations Psychophysiques*. PhD thesis, Laboratoire des Images e des Signaux, Universite Joseph Fourier de Grenoble, September 2006. 10, 12, 13, 16, 22, 127

[121] Corentin Massot and Jeanny Hérault. Model of frequency analysis in the visual cortex and the shape from texture problem. *International Journal of Computer Vision*, **76**(2):165–182, February 2008. xvi, xvii, 3, 5, 11, 13, 16, 19, 22, 23, 25, 78, 82, 87, 94, 100, 106, 113, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 160, 165

[122] Nelson L Max. Horizon mapping: Shadows for bump-mapped surfaces. *The Visual Computer*, **4**(2):109–117, mar 1988. 132

[123] Gene S. Miller and C. Robert Hoffman. Illumination and reflection maps: Simulated objects in simulated and real environments. *SIGGRAPH 84 Course Notes for Advanced Computer Graphics Animation*, **18**(3):11–20, jul 1984. 131

[124] R. Mohr, L. Quan, and F. Veillon. Relative 3d reconstruction using multiple uncalibrated images. *International Journal of Robotics Research*, **14**(6):619–632, 1995. 1

[125] William M. Newman and Robert F. Sproull. *Principles of Interactive Computer Graphics*. Mc-Graw Hill, New York, USA, 1973. 130

[126] Lyle Noakes and Ryszard Kozera. Nonlinearities and noise reduction in 3-source photometric stereo. *Journal of Mathematical Imaging and Vision*, **18**(2):119–127, 2003. 18

[127] Byong Mok Oh, Max Chen, Julie Dorsey, and Frédo Durand. Image-based modeling and photo editing. In *SIGGRAPH*, pages 433–442, New York, NY, USA, 2001. ACM. 134

[128] Ipek Oru, Laurence T. Maloney, and Michael S. Landy. Weighted linear cue combination with possibly correlated error. *Vision Research*, **43**(23):2451 – 2468, 2003. 16

[129] A W Paeth. A fast algorithm for general raster rotation. In *Proceedings on Graphics Interface '86/Vision Interface '86*, pages 77–81, Toronto, Ont., Canada, Canada, 1986. Canadian Information Processing Society. 132

[130] Darwyn R. Peachey. Solid texturing of complex surfaces. In *SIGGRAPH*, pages 279–286, New York, NY, USA, 1985. ACM. 132

[131] K. Perlin and E. M. Hoffert. Hypertexture. *SIGGRAPH Computer Graphics*, **23**(3):253–262, 1989. 132

[132] Ken Perlin. An image synthesizer. *SIGGRAPH Computer Graphics*, **19**(3):287–296, 1985. 132

[133] D. Perny, M. Gangnet, and Ph. Coueignoux. Perspective mapping of planar textures. *SIGGRAPH Computer Graphics*, **16**(1):70–100, 1982. 132

[134] Hossein Ragheb and Edwin R. Hancock. Surface normals and height from non-lambertian image data. In *3D Data Processing, Visualization, and Transmission*, pages 18–25, Washington, DC, USA, 2004. IEEE Computer Society. 18

[135] E. Ribeiro, P. L. Worthington, and E. R. Hancock. An eigendecomposition method for shape-from-texture. In *International Conference On Pattern Recognition*, **1**, pages 758–761, 2000. 22

[136] Eraldo Ribeiro and Edwin R. Hancock. Adapting spectral scale for shape from texture. In *European Conference on Computer Vision*, pages 421–435, London, UK, 2000. Springer-Verlag. 23, 46

[137] Eraldo Ribeiro and Edwin R. Hancock. Detecting multiple texture planes using local spectral distortion. In *British Machine Vision Conference*, pages 102–111, 2000. 52

[138] Eraldo Ribeiro and Edwin R. Hancock. Shape from periodic texture using the eigenvectors of local affine distortion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **23**(12):1459–1465, 2001. 11, 13, 22

[139] Eraldo Ribeiro and Edwin R. Hancock. Estimating the perspective pose of texture planes using spectral analysis on the unit sphere. *Pattern Recognition*, **35**:2141–2163, 2002. 3, 11, 52, 78

[140] Eraldo Ribeiro, Antonio Robles-Kelly, and Edwin R. Hancock. Detecting multiple texture planes using local spectral distortion. *Image and Vision Computing*, **20**:739 – 750, 2002. 52

[141] Antonio Robles-Kelly and Edwin R. Hancock. A graph-spectral method for surface height recovery from needle-maps. In *Computer Vision and Pattern Recognition*, pages 141–148, 2001. 18

[142] Antonio Robles-Kelly and Edwin R. Hancock. Surface height recovery from surface normals using manifold embedding. In *International Conference on Image Processing*, **3**, October 2004. 18

[143] V. Rodehorst. Vertiefende analyse eines gestalts-constraints von aloimonos und shulman. Technical report, Intitut fur Technische Informatik, Berlin, 1993. 18

[144] David F. Rogers. *Procedural elements for computer graphics*. McGraw-Hill, Inc., New York, NY, USA, 1985. 132

[145] Ruth Rosenholtz and Jitendra Malik. Surface orientation from texture: Isotropy or homogeneity (or both)? *Vision Research*, **37**(16):2283–2293, 1997. 9, 10

[146] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. "grabcut": interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, **23**(3):309–314, 2004. 134

[147] Ko Sakai and Leif H. Finkel. Characterization of the spatial-frequency spectrum in the perception of shape from texture. *Journal of the Optical Society of America*, **A 12**:1208–1224, 1995. 11, 21

[148] Ko Sakai and Leif H. Finkel. Spatial-frequency analysis in the perception of perspective depth. *Network: Computation in Neural Systems*, **8**(3):335–352, 1997. 21

[149] Ufuk Sakarya, Uğur Murat Leloğlu, and Erol Tunal. Three-dimensional surface reconstruction for cartridge cases using photometric stereo. *Forensic Science International*, **175**(2-3):209–217, 2008. 18

[150] W. Brent Seales and Olivier D. Faugeras. Building three-dimensional object models from image sequences. *Computer Vision and Image Understanding*, **61**(3):308–324, 1995. 1

[151] Jack Sklansky. Image segmentation and feature extraction. *IEEE Transactions on Systems, Man and Cybernetics*, **8**(4):237–247, apr 1978. 2

[152] Alvy Ray Smith. Incremental rendering of texture in perspective. SIGGRAPH '80 Animation Graphics seminar notes, jul 1980. 132

[153] Alvy Ray Smith. Digital filtering tutorial for computer graphics. Computer Graphics Department, Computer Division, Lucasfilm Ltd. Technical Memo 27 (Also in Tutorial Notes for Siggraph 1983 and Siggraph 1984), mar 1983. 133

[154] William A. Smith and Edwin R. Hancock. Facial shape-from-shading and recognition using principal geodesic analysis and robust statistics. *International Journal Computer Vision*, **76**(1), 2008. 18

[155] Kent A Stevens. The information content of texture gradients. *Biological Cybernetics*, **42**(2):95–105, 1981. 8, 20, 52

[156] J. V. Stone and S. D. Isard. Adaptive scale filtering: a general-method for obtaining shape from texture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **17**(7):713–718, July 1995. 23, 46, 52

[157] James V Stone. Shape from local and global analysis of texture. *Philosophical Transactions of the Royal Society of London, B*, pages 53–65, 1993. 20

[158] Boaz J. Super and Alan C. Bovik. Planar surface orientation from texture spatial frequencies. *Pattern Recognition*, **28**(5):729–743, 1995. xii, 5, 11, 13, 14, 16, 19, 21, 23, 25, 26, 27, 28, 29, 32, 33, 36, 39, 40, 41, 42, 43, 46, 50, 52, 53, 55, 57, 63, 70, 78, 84, 101, 121, 136, 159

[159] Boaz J. Super and Alan C. Bovik. Shape from texture using local spectral moments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **17**(4):333–343, April 1995. xii, 3, 11, 15, 21, 22, 23, 25, 27, 28, 29, 32, 33, 36, 39, 65, 66, 67, 68

[160] Demetri Terzopoulos. The computation of visible-surface representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **10**(4):417–438, 1988. 18

[161] J T Todd and R A Akerstrom. Perception of three-dimensional form patterns of optical texture. *Journal of Experimental Psychology: Human Perception and Performance*, **13**(2):242 – 255, 1987. 3, 10

[162] James T. Todd, Lore Thaler, and Tjeerd M.H. Dijkstra. The effects of field of view on the perception of 3d slant from texture. *Vision Research*, **45**(12):1501–1517, 2005. 16

[163] Yanghai Tsin, Yanxi Liu, and Visvanathan Ramesh. Texture replacement in real images. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, **2**:539, 2001. 133, 134

[164] J.K. Udupa. Determination of 3-d shape parameters from boundary information. *Computer Graphics and Image Processing*, **17**(1):52–59, September 1981. 1

[165] Paul A. Viola and William M. Wells III. Alignment by maximization of mutual information. In *International Conference on Computer Vision*, pages 16–23, jun 1995. 134

[166] Li-Yi Wei and Marc Levoy. Fast texture synthesis using tree-structured vector quantization. In *SIGGRAPH*, pages 479–488, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co. 133

[167] Tiangong Wei and Reinhard Klette. A new algorithm for gradient field integration. In *Image and Vision Computing New Zealand*, pages 109–114, Dunedin, New Zealand, 2001. 18

[168] Tiangong Wei and Reinhard Klette. Theoretical analysis of finite difference algorithms for linear shape from shading. In *International Conference on Computer Analysis of Images and Patterns*, pages 638–645, London, UK, 2001. Springer-Verlag. 18

[169] Tiangong Wei and Reinhard Klette. A wavelet-based algorithm for height from gradients. In *International Workshop on Robot Vision*, pages 84–90, London, UK, 2001. Springer-Verlag. 18

[170] Tiangong Wei and Reinhard Klette. Depth recovery from noisy gradient vector fields using regularization. In *International Conference on Computer Analysis of Images and Patterns*, pages 116–123, Groningen, The Netherlands, November 2003. 18

[171] Carl F. R. Weiman. Continuous anti-aliased rotation and zoom of raster images. In *SIGGRAPH*, pages 286–293, New York, NY, USA, 1980. ACM. 132

[172] Ryan White and David A. Forsyth. Combining cues: Shape from shading and texture. In *Computer Vision and Pattern Recognition*, pages 1809–1816, Washington, DC, USA, 2006. IEEE Computer Society. 1

[173] Turner Whitted. The causes of aliasing in computer generated images. SIGGRAPH '81 Advanced Image Synthesis seminar notes, aug 1981. 133

[174] Andrew P. Witkin. Recovering surface shape and orientation from texture. *Artificial Intelligence*, **17**:17–45, August 1981. 9, 20, 65

[175] Zhongquan Wu and Lingxiao Li. A line-integration based method for depth recovery from surface normals. *Computer Vision, Graphics and Image Processing*, **43**(1):53–66, 1988. 18

[176] Yizhou Yu and J. T. Chang. Shadow graphs and surface reconstruction. In *European Conference on Computer Vision*, pages 31–45, London, UK, 2002. Springer-Verlag. 1

[177] Steve Zelinka, Hui Fang, Michael Garland, and John C. Hart. Interactive material replacement in photographs. In *Proceedings of Graphics Interface*, pages 227–232, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 2005. Canadian Human-Computer Communications Society. 133, 134

[178] Ruo Zhang, Ping-Sing Tsai, James Edwin Cryer, and Mubarak Shah. Shape from shading: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **21**(8):690–706, 1999. 1