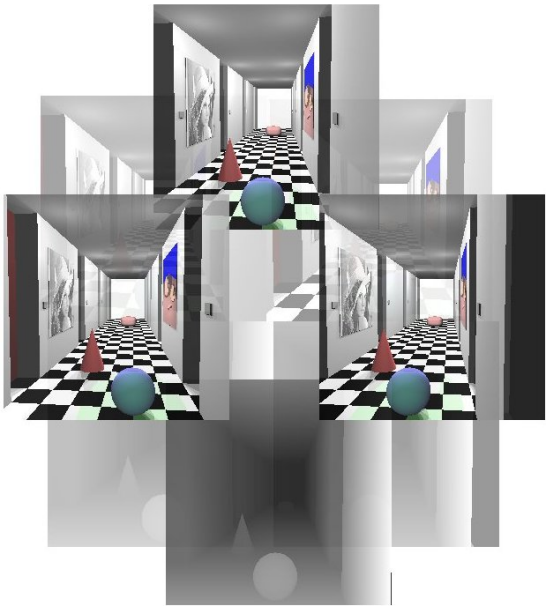
Evaluation of Stereo Matching Algorithm for Hardware Implementation



Philippe LECLERCQ

This thesis is presented for the degree of
Doctor of Philosophy
of The University of Western Australia
Faculty of Engineering, Computing and Mathematics
School of Electrical, Electronic and Computer Engineering
CIIPS - Centre for Intelligent Information Processing Systems

June 16, 2004

Abstract

Stereovision is an active research domain: many different solutions to the matching problem have been proposed. This study's scope was **binocular** stereo, **dense** matching and algorithms suitable for efficient **hardware implementation**. Post-processing was not included to keep the comparisons to the core of the algorithms.

Performance of Census, Pixel-to-Pixel and three correlations - normalised square of difference (C_1), normalised multiplicative (C_2) and sum of absolute differences (SAD) - was *measured* using ground truth maps. Although Census performed worst, it was a good hardware implementation candidate. Therefore two variants were evaluated: weighting the transform costs by their distance to the centre did not have significant effect, but modifying the bit patterns showed improvements up to 17%.

Noise robustness is vital for stereovision systems. Using a noiseless ray-traced image corrupted with Gaussian noise, Pixel-to-Pixel clearly outperformed the other algorithms, up to a SNR of 15dB and was stable up to 33dB.

Two transforms using colour information showed no improvements, emphasizing the strong correlation between colour and intensity values. Both this study and literature showed that colour does not *consistently* improve matching quality.

Increasing the baseline increases the depth accuracy but also occlusions and the difference in numbers of pixels in subtended surfaces; the Lambertian surfaces assumption also becomes harder to justify. As the baseline increased from $0.1mto0.9m$, good matches dropped from 70% to $\sim$ 25% and standard deviation increased from 0.5 to $\sim$ 18.

A shell was measured both under ambient light and using active colour illumination. The latter consistently improved matching with no increase in computation complexity.

An aerial set with both laser range scanner and manual entry disparity maps was provided by the French National Geographic Institute. Using the manual map as the ground truth, SAD showed 20% good matches ($\sigma = 16$) *vs.* 38% ($\sigma = 6$) for laser range scanning.

Contents

List of Figures

List of Tables

Acknowledgements

Doing a Ph.D. is like a quest, where the knight *seeks wisdom*[1]. Even though the quest is mostly the knight's journey, it involves different characters helping solve parts of the enigmas. After training in my family's castle for a while I went to an engineering school. After becoming a knight, I had to find a kingdom to support my new quest. Australia granted me support on the basis of an initial test I had to pass as well as my former jousting records. While preparing for this test, I met a fairy who gave me the gift of a foreign language to help me. Once started, my quest took me all the way from France to Australia first. There I met a wizard who offered to train me. Using *ingenio et labore*[2] along with my wizard's guidance and my Lady of the Lake's support I made my way assimilating new knowledge and writing scrolls. At last the quest took me to New-Zealand, with my wizard, to face my last challenges. This present scroll tells the whole story of my journey...

Credits

I would like to thank, in order of appearance:

My parents who have always been interested in my projects and very supportive of my choices. Mom, I know you would have been the happiest seeing this work come to an end but you unfortunately left us far too soon.

Dr. Yves Delignon and Dr. Laurent Clavier from my engineering school - Ecole Nouvelle d'Ingenieurs en Communication (ENIC, Telecom Lille 1) - and Dr. Marc Gazalet from the Université de Valenciennes et du Hainaut Cambrésis (UVHC) for helping me to obtain a second masters while finishing my principal masters of engineering at the ENIC. This is where this whole project started...

Karen Maddern - the Lady of the Lake - for being such a good friend.

Diane Hart - the fairy - for her patience bringing my writing skills up to the IELTS standards for the International Postgraduate Research Scolarship (IPRS).

The Australian Government - the kingdom - and the University of Western Australia for providing me with an IPRS scholarship to undergo my Ph.D., especially in

[1] University of Western Australia's motto

[2] University of Auckland's motto

the person of Nicholas Baker, Education Advisor at the Australian Embassy in Paris who has been of great help and encouragement at the early stages of the project.

A/Prof Thomas Braünl, director of Centre for Intelligent Information Processing Systems (CIIPS) at the University of Western Australia to welcome me in his lab.

A/Prof John Morris - the wizard - both skipper on Iolanthe and of my research. It was a great pleasure, as well as a bit of destiny, to follow you in the city of sails - Auckland - to finish my research after all the great sailing in Perth.

Viviane and Chris Jenson, Charles Ackland and Michael Knowles from Trinity college (former Kingswood) who made my life as an "older" student in college an unforgettable experience.

A/Prof Georgy Gimel'farb, the University of Auckland, for sharing his insights and broad experiences of the stereovision domain.

Dr. Patrice Delmas, the University of Auckland, for offering me a position keeping me researching on "after thesis" topics.

Thanks to Marc Pierrot-Deseilligny and Grégoire Maillet from the MATIS laboratory (Institut Géographique National, IGN) for providing the aerial rectified stereopair with both manual entry and laser range scanner dense groundtruths.

φ^2

Part I

Introduction and Stereo Geometry

Chapter 1

General Introduction

Stereovision is the use of two images - taken with two cameras or a single camera translated by a known distance - of the same scene to compute its depth information. The main task of stereovision algorithms is matching an object's position in one image with the same object's position in the second image to compute its depth by triangulation.

Lane and Thacker surveyed a dozen stereo vision algorithms from 1973 up to 1992 [40]. They split the algorithms into:

- *area-based*: producing dense depth maps *i.e.* every single pixel in the image has a computed depth, and
- *feature-based*: producing sparse depth maps *i.e.* only pre-processed pixels (edges for instance) are matched and other pixels are interpolated if needed.

Unfortunately Lane and Thacker's survey only described the different algorithms and did not produce any comparisons between them. This study was motivated by a desire to produce a real-time stereo vision system in hardware and thus an expectation that the literature would provide guidance as to the most efficient and best performing algorithms rather than just a menu from which a random choice must be made. Metrics to compare algorithm performance are clearly needed in this domain where a large number of approaches have been proposed but not carefully compared: the metrics used in this study are described in section 3.4.

In 1993, Koschan in [41] was mentioning the existence of more than 150 algorithms and underlining the lack of any form of comparison or framework for comparing these algorithms.

After this study had commenced, the large study by Scharstein and Szeliski [42] and [43], (improving the previous [44]) appeared: they compared several algorithms

accuracies on several sets with known ground truth. However, additional metrics were used here and several different experiments - for instance this study's robustness to noise experiment (see chapter 7 and appendix D) - have been carried out, complementing their study. The algorithms used in this study are discussed in section 3.2.

Stereo vision has a wide range of possible applications which can be classified into:

- *static scene description* where accuracy is the most important consideration and the processing time is not critical. Situations needing precise 3D measures with the possibility of setting a practical device up to take stereo pairs of a scene and reconstruct them afterwards, for instance for:
  - cartography,
  - dimensions of an object measurements in situations where access to the objects is constrained so that conventional tools (rules, calipers,...) are not useable,
  - crime scene reconstruction for the police, where the position of all items would be easily accessible through the 3D model,
  - car crashes scene reconstruction, would give the final position of all objects and could be used as a backward condition to a 3D model to analyse the crash parameters like speed and trajectory,
  - architects, where panoramic pictures enable the 3D reconstruction of the scene and work with 3D models of the constructions,
  - ...
- *dynamic scene description* where real-time processing is critical as well as a satisfying an accuracy requirement. Possible situations are:
  - obstacle avoidance, for instance cars detecting pedestrians and alerting the driver or even applying the brakes automatically
  - autonomous navigation, where the mobile object updates its internal map while moving,
  - automated security controls, *e.g.* an unmanned railway station - Garibotto *et al.* in [45] provide a detailed description of this application,
  - height estimation, for instance for an autonomous helicopter - Corke [46],
  - ...

This study compares several algorithms on defined sets of stereo pairs[1] using consistent metrics, with the ultimate aim of guiding a hardware implementation. Only area based algorithms have been studied because they involve fewer decisions and thus they are better suited to pipelined hardware processing. In addition a parallel study of feature based algorithms was performed by Catherine Davey [47]. The hardware and real-time orientation of this work had a considerable influence on the algorithms chosen for evaluation, for example, graph cut described in section 3.2.4 was excluded, despite its good performances, because of its processing time. Although the initial goal of this study was real-time applications, it discusses points belonging to both the static and dynamic areas; the geometry and especially the accuracy was formally analysed in order to provide a better understanding of the experimental configurations. A series of experiments comparing existing algorithms or modifications of them were conducted.

- Chapter 2 discusses the geometry for both parallel and verging (*i.e.* crossing) camera axes. Use of a biprism is also discussed. It introduces a depth accuracy measure and shows how to determine the camera configuration which obtains the best depth accuracy.

- Chapter 3 describes the experimental environment: the different stereo algorithms, an outline of the software written, metrics used and the stereo pairs used.

- Chapter 4 describes procedures procedures to set up an experiment to achieve the accuracies discussed in chapter 2.

- Chapter 5 describes the results of 'baseline' experiments in which all the original algorithms were run on the synthetic stereo pairs as well as those used by Scharstein and Szeliski [43].

- Chapter 6 assesses two variants of the Census algorithm (which has considerable potential for efficient hardware implementation but relatively poor performance) to determine whether it warranted further consideration.

- Chapter 7 evaluates the robustness of selected algorithms against increasing levels of noise in the images.

- Chapter 8 evaluates the possible benefit of using the colour information.

[1] All the images used by Scharstein and Szeliski [43] were used in this study as well as some new ones.

- Chapter 9 evaluates the consequences of increasing the baseline in order to achieve higher depth accuracy.
- Chapter 10 describes measurements of a fossil shell as an example of the application of the theory described in chapter 2 and configuration procedures in chapter 4.
- Chapter 11 describes experiments on an image pair taken with verging (*i.e.* non-parallel) camera axes: all other experiments used configurations in which the camera axes were parallel. It also compares the binocular stereo algorithms of this study against a laser generated disparity map.
- Chapter 12, the final chapter - Conclusions - does exactly what its title suggests...
- Appendix A gives a more detailed description of the software written for this study.
- Appendix B reproduces de Corridor (one of the used stereo pair) ray-traced scene description file.
- Appendix C and D contain the full versions of two papers based on the work in this thesis and presented at conferences.

Chapter 2

Stereo Geometry

2.1 Introduction

This chapter presents an overview of the geometry and configurations for stereovision. Firstly configurations in which the optical axes of the cameras are parallel are discussed. This part starts by describing the camera parameters and geometry, then introduces the *depth accuracy* measure for experiments setup and fully explains the behaviour of this accuracy. In chapter 4 an experimental way of determining the camera's sensor pixel size is illustrated with two examples, followed by a description of procedures for setting up a camera pair to achieve a target accuracy requirement. Then, the case of non-parallel camera optical axes is discussed: one of the image sets used as a test has this configuration (see **??**). Finally the use of a biprism, described by Lee *et al.*(see [1]), as a stereovision device is discussed.

Throughout this discussion, epipolar geometry is assumed: Zhu and Zhang provide an extensive discussion of this [48]. It is usual to consider the *epipolar constraint*: a point on one line in one image will be found on the same line in the other image. This can easily be achieved by careful camera setup (see the practical experiment for more details 4.4) or can be pre-processed by image *rectification* or *regularisation*: for instance, Dr. Thomas Fischer's rectification application (see [49]) is composed of the following steps:

- manual selection of a set of corresponding pixels in both images,
- computation of the fundamental matrix using an approximate linearised approach [50],
- refinement of the approximate matrix by using non-linear optimisation with the Levenberg-Marquadt gradient-based algorithm and

- forming a rectified pair where each initial epipolar line becomes a horizontal scan-line.

Definitions

The parallax is the apparent displacement, or difference of position, of an object, as seen from two different stations, or points of view; the *binocular* parallax is the apparent difference in position of an object as seen separately by one eye, and then by the other, the head remaining unmoved.

In image processing, disparity has the same meaning as 'binocular parallax' and is usually the difference of position in pixels of a point between the two views.

2.2 Parallel Camera Axes

2.2.1 Notations

Figure 2.1 illustrates the stereo configuration: two cameras have parallel optical axes ($O_lO'_l$ and $O_rO'_r$ for the left and right cameras respectively, the camera's optical centres are O_l and O_r and their projections on the object plane are O'_l and O'_r) set apart by a distance b (the baseline). FoV is the Field of View for a *single* camera, a is the object's extent and $CFoV$[1] is the *common* field of view of the two cameras, *i.e.* the binocularly visible points.

p is the *physical* width of one pixel on the camera's sensor and n the number of pixels on one scanline. If D_{chip} is the width of the active part of the camera's sensor, then:

$$\boxed{p = \frac{D_{chip}}{n}} \tag{2.1}$$

Distances are denoted by $D_{...}$ and disparities using $d_{...}$. Usually, the output of a stereo algorithm is disparities in *pixels*, representing the distance between the projection of one point on one scanline on one image to the projection of the same point on the same scanline in the second image. For equations *homogeneity*, disparities are homogenous to *distances*. Using p, the physical width of one pixel, we may convert easily between pixels and μm for instance.

Both cameras have the same parameters: Θ_{MAX}, the half view angle of the

[1] Note that the *common* field of view and the object's extent, a, can be related by a constant:

$$CFoV = \rho \cdot a \qquad \text{where} \quad \rho \geq 1$$

camera, and D_1, the focal distance. Since most modern cameras have zoom lenses, Θ_{MAX} and D_1 have to be determined for *any* configuration of the camera.

Note that precise geometric information is not usually available for the economic digital cameras that should be able to be used. A procedure for determining the actual geometry is outlined in section 4.4.

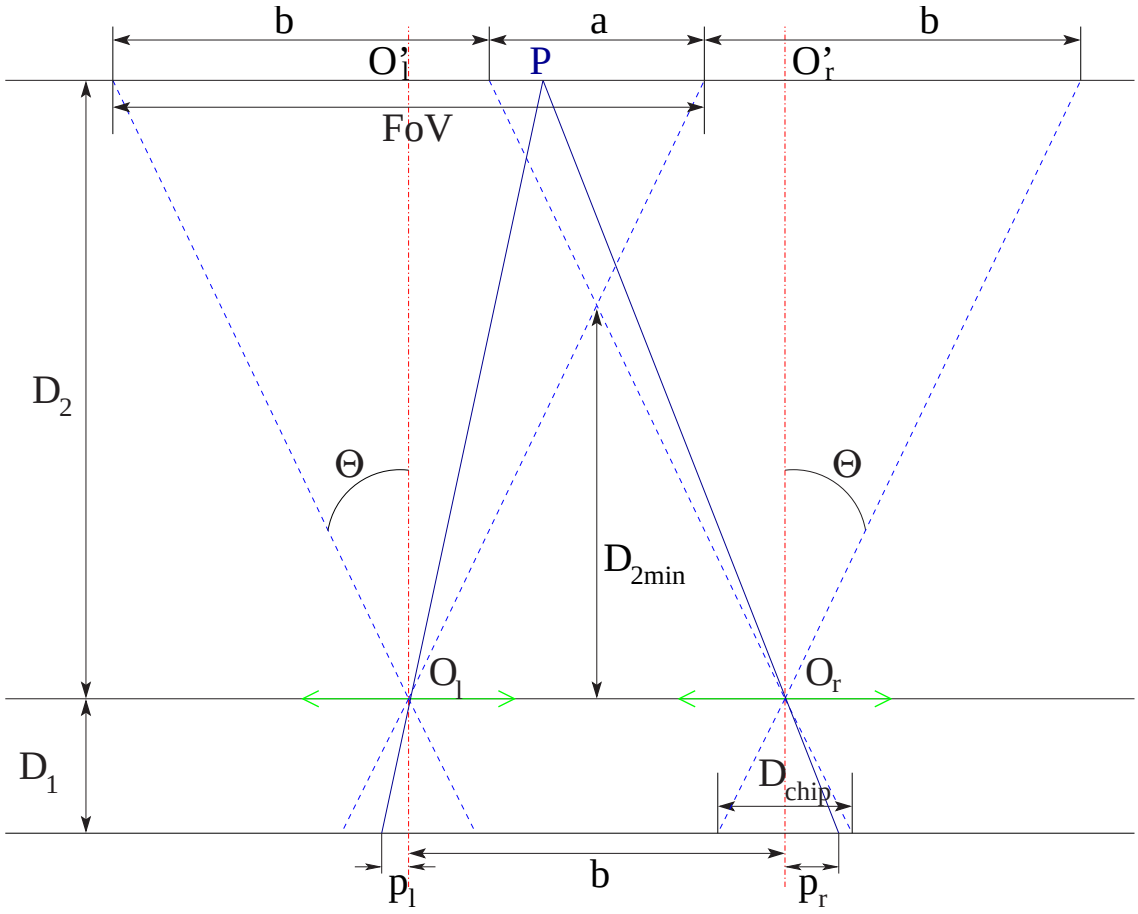


FIGURE 2.1: Stereo camera configuration: two cameras are positioned with their optical centres at O_l and O_r separated by the baseline, b. The remaining symbols are explained in the text.

2.2.2 Geometry Description

Camera characterisation

The size of the *image* plane D_{chip}, the focal distance D_1 and Θ_{MAX} are related (*cf.* figure 2.1) by:

$$\tan\Theta_{MAX} = \frac{D_{chip}}{2 \cdot D_1} \quad \text{or} \quad D_1 = \frac{D_{chip}}{2 \cdot \tan\Theta_{MAX}} \tag{2.2}$$

The size of the *object* plane FoV, the distance to the object, D_2, and Θ_{MAX} are related (*cf.* figure 2.1) by:

$$\boxed{\tan\Theta_{MAX} = \frac{FoV}{2 \cdot D_2} \quad \text{or} \quad D_2 = \frac{FoV}{2 \cdot \tan\Theta_{MAX}}} \tag{2.3}$$

Note that D_{chip} is a constant, so using the image plane as a reference, Θ_{MAX} will determine the position of the optical centre (focal distance) and the field of view at distance D_2.

Common Field of View

From figure 2.1 we easily find that:

$$FoV = CFoV + b$$

Replacing FoV in equation 2.3 leads to a relation between $CFoV$ (the *common* field of view of the two cameras, *i.e.* the size of the largest object that can be fully imaged by both cameras at distance D_2), the object's distance D_2 and the baseline b:

$$FoV = 2 \cdot D_2 \cdot \tan\Theta_{MAX} = CFoV + b$$

i.e.

$$\boxed{CFoV = 2 \cdot D_2 \cdot \tan\Theta_{MAX} - b} \tag{2.4}$$

Distance and Depth

In figure 2.1, P is seen by both cameras: its disparity is d_P. The *similar triangles* theorem (Thales) links the distances on the camera sensors (p_l or p_r) to the distances $\overline{O'_lP}$ or $\overline{O'_rP}$, D_1 and D_2, *i.e.*:

$$p_l = \frac{D_1 \cdot O'_lP}{D_2} \qquad p_r = \frac{D_1 \cdot O'_rP}{D_2}$$

Note that, as already mentioned, p_l and p_r are usually given in pixel position on the images but may be converted to a distance by multiplying by p, the *physical* width of a pixel on the sensor.

The disparity d_P of point P is defined as the difference in position on the left

and the right images:

$$\begin{aligned} d_P &= p_l - p_r \\ &= \frac{D_1}{D_2} \cdot (\overline{O_l'P} - \overline{O_r'P}) \\ &= \frac{D_1}{D_2} \cdot (\overline{O_l'P} + \overline{PO_r'}) \\ &= \frac{D_1}{D_2} \cdot b \end{aligned}$$

$$\boxed{d_P = \frac{D_1 \cdot b}{D_2}} \tag{2.5}$$

If the object is placed at an infinite distance:

$$D_2 \to \infty \quad \Rightarrow \quad d_p \to 0 \qquad \forall \quad b \neq 0, D_1 \neq 0.$$

This shows that an object at infinity appears at the same place on both image planes. Practically, the camera resolution (*i.e.* pixel width p) will fix the minimum measurable disparity, *i.e.* the maximum depth.

On the other hand:

$$D_2 \to 0 \quad \Rightarrow \quad d_p \to \infty \qquad \forall \quad b \neq 0, D_1 \neq 0.$$

Figure 2.1 shows that D_2 cannot reach 0 for a finite sensor width. Equation 2.4 is used to determine the smallest distance D_{2min} for which a single point is imaged by both cameras (*i.e.* $CFoV = 0$):

$$\begin{aligned} a = 0 &= 2 \cdot D_{2min} \cdot \tan\Theta_{MAX} - b \\ \Rightarrow \quad D_{2min} &= \frac{b}{2 \cdot \tan\Theta_{MAX}} \end{aligned}$$

replacing $\tan\Theta_{MAX}$:

$$\boxed{D_{2min} = \frac{b \cdot D_1}{n \cdot p}} \tag{2.6}$$

One can verify that the closest distance is obtained for the largest possible disparity.

Depth Accuracy

Let $\delta D_2'$ be the distance between a point at depth D_2 and the point just in front at depth D_2'. Similarly, let $\delta D_2''$ be the distance between a point at depth D_2 and

the point just behind at depth D_2'', Figure 2.2 illustrates that second situation. Assuming that the smallest measurable difference is p, the width of one pixel, leads to $d_2' = d_2 + p$ and $d_2'' = d_2 - p$. Then using equation 2.5 to replace d_2 by D_2 gives:

$$\begin{aligned} \delta D_2' &= D_2 - D_2' \\ &= D_2 - \frac{b \cdot D_1}{d_2'} \\ &= \frac{p \cdot D_2^2}{b \cdot D_1 + p \cdot D_2} \end{aligned} \qquad \begin{aligned} \delta D_2'' &= D_2'' - D_2 \\ &= \frac{b \cdot D_1}{d_2''} - D_2 \\ &= \frac{p \cdot D_2^2}{b \cdot D_1 - p \cdot D_2} \end{aligned}$$

All values are distances or dimensions and are positive, so $\delta D_2''$ is larger than $\delta D_2'$. In order to treat the worst case, write:

$$\boxed{\delta D_2 = \frac{p \cdot D_2^2}{b \cdot D_1 - p \cdot D_2}} \tag{2.7}$$

Equation 2.7 gives the maximum *absolute* accuracy for a given configuration as a function of the distance D_2.

Relative Depth Accuracy

Other applications might need a maximum *relative* accuracy, defined as:

$$\varepsilon_{D_2} = \frac{\delta D_2}{D_2}$$

From equation 2.7:

$$\boxed{\varepsilon_{D_2} = \frac{p \cdot D_2}{b \cdot D_1 - p \cdot D_2}} \tag{2.8}$$

This measure is useful in obstacle avoidance problems as it gives the accuracy as a function of the distance to the obstacle.

Rewriting equation 2.8 by dividing by $p \cdot D_2$ gives:

$$\varepsilon_{D_2} = \frac{1}{\frac{b \cdot D_1}{p \cdot D_2} - 1} \quad \Rightarrow \quad \boxed{\begin{cases} D_2 \nearrow & \Rightarrow \quad \varepsilon_{D_2} \nearrow \\ D_2 \searrow & \Rightarrow \quad \varepsilon_{D_2} \searrow \end{cases}}$$

$$\Rightarrow \quad \boxed{\begin{cases} b \nearrow & \Rightarrow \quad \varepsilon_{D_2} \searrow \\ b \searrow & \Rightarrow \quad \varepsilon_{D_2} \nearrow \end{cases}}$$

For obstacle avoidance problems, the relative accuracy ε_{D_2} decreases for nearer ob-

jects and increases for farther objects.

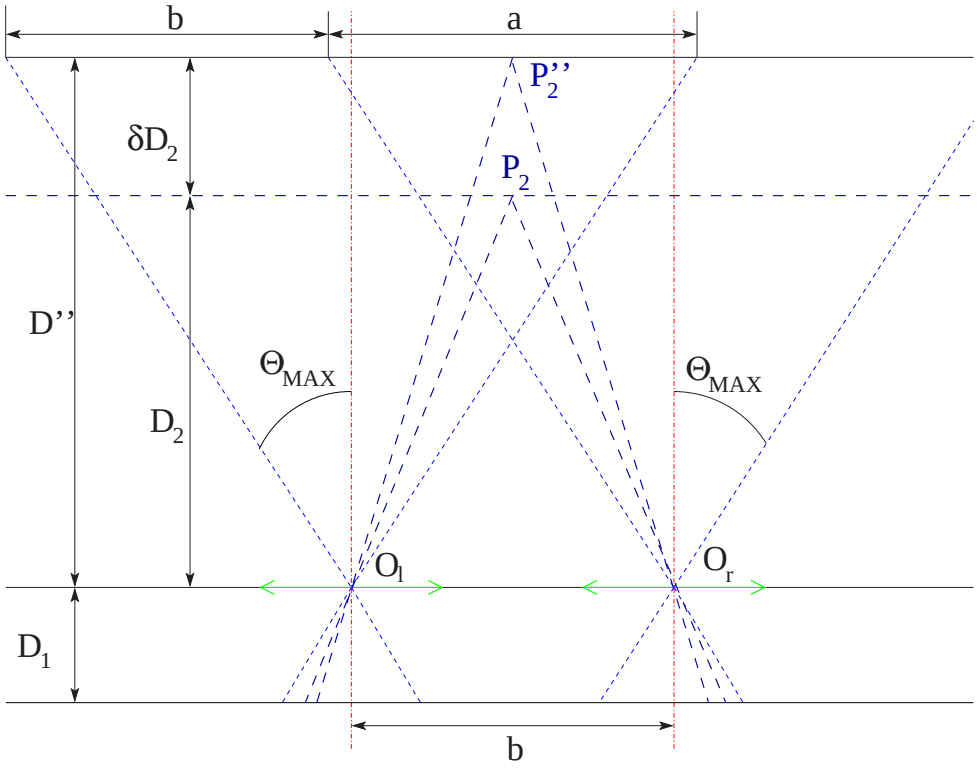


FIGURE 2.2: Stereo cameras setting.

2.2.3 Derived Relations

In a practical application, there will be an object (or collection of objects) with a maximum extent, a, in a plane parallel to the image plane. In practice one needs to have a small region around the object of interest, so that the *common* field of view, $CFoV = \rho \cdot a$, where $\rho \geq 1$. In what follows, for simplicity, it is assumed that a is the common field of view also. The application will determine the accuracy to which we need to be able to measure features of the object(s), *e.g.* one may have a collection of objects subtending a $100mm \times 100mm$ region in the object plane for which surface features need to be measured to an accuracy of 1mm. In this case, $\delta D_{2target} = 1mm$ and it is necessary to determine D_2 and b so that $\delta D_{2limit} \leq 1mm$. The extent of the object will clearly determine a minimum value for D_2, the closest distance at which the object can be placed and still be visible in both images.

Behaviour of the Accuracy Limit with D_2

It is necessary to determine the value of D_2 which leads to the best accuracy, *i.e.* the point at which $\frac{\partial \delta D_2}{\partial D_2} = 0$. δD_2 is given in equation 2.7 as a function of b, but a is related to the object(s) size and is fixed by the problem conditions and is thus a constant in any experiment. b is replaced by its expression as a function of D_2 using equation 2.4:

$$\delta D_2 = \frac{p \cdot D_2^2}{D_2 \cdot (2 \cdot \lambda \cdot D_1 - p) - a \cdot D_1} \tag{2.9}$$

D_1 and p are constants as they are properties of the camera itself [2]. See section 4.4 for an example solution.

Setting:

$$\mu = 2 \cdot D_1 \cdot \lambda - p = D_{chip} - p = (n-1) \cdot p \quad \text{(using 2.2)}$$

which is a *constant* for a *given* camera sensor as it only depends on the camera internal parameters.

Note that μ is always positive: D_{chip} is the size of the chip and p the size of one pixel.

Setting:

$$\begin{cases} u = p \cdot D_2^2 \\ v = D_2 \cdot \mu - a \cdot D_1 \end{cases}$$

gives:

$$\delta D_2 = \frac{u}{v}$$

and the differentiation rule gives:

$$\Rightarrow \quad \frac{\partial \delta D_2}{\partial D_2} = \frac{u' \cdot v - u \cdot v'}{v^2}$$

where u' and v' are the derivatives of u and v with respect to D_2:

$$\begin{cases} u' = 2 \cdot p \cdot D_2 \\ v' = \mu \end{cases}$$

[2] In cameras with zoom lenses, D_1 may be adjustable over a fixed range

Substituting back:

$$\begin{aligned}\frac{\partial \delta D_2}{\partial D_2} &= \frac{\mu \cdot p \cdot D_2^2 - 2a \cdot p \cdot D_1 \cdot D_2}{v^2} \\ &= \frac{p \cdot D_2}{v^2} \cdot \left(\mu \cdot D_2 - 2 \cdot a \cdot D_1\right)\end{aligned} \tag{2.10}$$

Now:

$$\frac{\partial \delta D_2}{\partial D_2} = 0 \quad \Rightarrow \quad p \cdot D_2 \cdot (\mu \cdot D_2 - 2 \cdot a \cdot D_1) = 0$$

$$\Rightarrow \begin{cases} D_2 = 0 \\ \text{or,} \\ D_2 = \dfrac{2 \cdot a \cdot D_1}{\mu} \end{cases}$$

$D_2 = 0$ is not a realisable solution because equation 2.6 showed that $D_{2min} > 0$. So the only solution is:

$$\boxed{D_{2best} = \frac{2 \cdot a \cdot D_1}{\mu}} \tag{2.11}$$

The derivative expression's sign describes the function's behaviour: p, μ, a, D_1 and v^2 are always positive. There is a unique value of D_2 that minimises δD_2, so given the camera parameters and common field of view, $\rho \cdot a$, one can find the optimum position for the object.

The position of the vertical asymptote of the accuracy function is found when the denominator in expression 2.9 equals zero:

$$\mu \cdot D_{2asymptote} - a \cdot D_1 = 0$$

The asymptote's position is:

$$D_{2asymptote} = \frac{a \cdot D_1}{\mu}$$

Back substituting μ leads to:

$$\boxed{D_{2asymptote} = \frac{a \cdot D_1}{(n-1) \cdot p}} \tag{2.12}$$

The sign on both sides of the vertical asymptote is:

left: $D_2 < \dfrac{a \cdot D_1}{\mu}$ $\qquad D_2 \cdot \mu < a \cdot D_1$ $\qquad \delta D_2 < 0$

right: $D_2 > \dfrac{a \cdot D_1}{\mu}$ $\qquad D_2 \cdot \mu > a \cdot D_1$ $\qquad \delta D_2 > 0$

The function derivative and behaviour for all D_2 values is:

D_2	$-\infty$		0		$\dfrac{2 \cdot a \cdot D_1}{\mu}$	$+\infty$
$\dfrac{\partial \delta D_2}{\partial D_2}$	$+$		0	$-$	0	$+$
$\delta D_2(D_2)$	$\nearrow$			$\searrow$		$\nearrow$

TABLE 2.1: Behaviour of the accuracy limit δD_2 *vs.* D_2

Rewriting equation 2.12 brings:

$$\frac{D_{2asymptote}}{a} = \frac{D_1}{(n-1) \cdot p}$$

which clearly shows that the asymptote corresponds to an object of extension a viewed over a *common* field of $n-1$ pixels (see figure 2.3). So that there is only one pixel "available" for a shift and the only possible disparities are 0 and 1.

Substituting $D_{2asymptote}$ in the expression for the common field of view 2.4 leads to:

$$a_{asymptote} = \frac{2 \cdot a \cdot D_1 \cdot \tan \Theta_{MAX}}{(n-1) \cdot p} - b$$

which leads to the expression of the baseline $b_{asymptote}$:

$$b_{asymptote} = a_{asymptote} \cdot (\frac{n \cdot p}{(n-1) \cdot p} - 1) = a \cdot \frac{1}{n-1}$$

Note that if an object of extent, a, were to have an image one pixel wider, so that:

$$D_2 = \frac{a \cdot D_1}{n \cdot p}$$

then $b = 0$ and the configuration is no longer stereo vision.

There are only two disparities possible, 0 and 1 and the boundary between the

domain where $d = 0$ and $d = 1$ is situated at:

$$D_{2d=1} = D_{2asymptote} \qquad = \frac{b \cdot D_1}{p} = \frac{a \cdot D_1}{(n-1) \cdot p}$$

Note that this is also $D_{2asymptote}$.

To determine the region visible to both cameras which starts at depth, D_{2inter}, note that:

$$\begin{aligned} \tan\Big(\frac{\pi}{2} - \Theta_{MAX}\Big) &= \frac{b}{2 \cdot D_{2inter}} \\ &= \frac{1}{\tan \Theta_{MAX}} \\ &= \frac{2 \cdot D_2}{a+b} \\ &= \frac{2 \cdot D_2 \cdot (n-1)}{a \cdot n} \\ \Rightarrow \quad D_{2inter} &= \frac{a \cdot n}{2 \cdot (n-1) \cdot \tan \Theta_{MAX}} \\ &= \frac{a \cdot D_1}{(n-1) \cdot p} \end{aligned}$$

and finally:

$$\boxed{D_{2inter} = D_{2asymptote} = D_{2d=1}}$$

This shows that the only "visible", or measurable, disparity is $d = 0$ as all the $d = 1$ disparity region lies before $D_2 < D_{2inter}$ *i.e.* outside the common field of view, see figure 2.3. This explains the asymptote's position $D_{2asymptote}$.

Note that the best position is at:

$$2 \cdot D_{2asymptote} = \frac{2 \cdot a \cdot D_1}{(n-1) \cdot p}$$

In this case, the common field of view has an extent of $2 \cdot a$ over a region of $(n-1)$ pixels, which corresponds to the result found in the following section: the best situation is when the baseline length is similar to the object extent, a.

The graph in figure 2.4 illustrates the function behaviour for positive values of D_2. Note that usable values for D_2 are greater than the asymptote's position, $D_{2asymptote}$.

Knowing the best value for D_2 (*cf.* 2.11), and substituting D_2 from equation 2.4

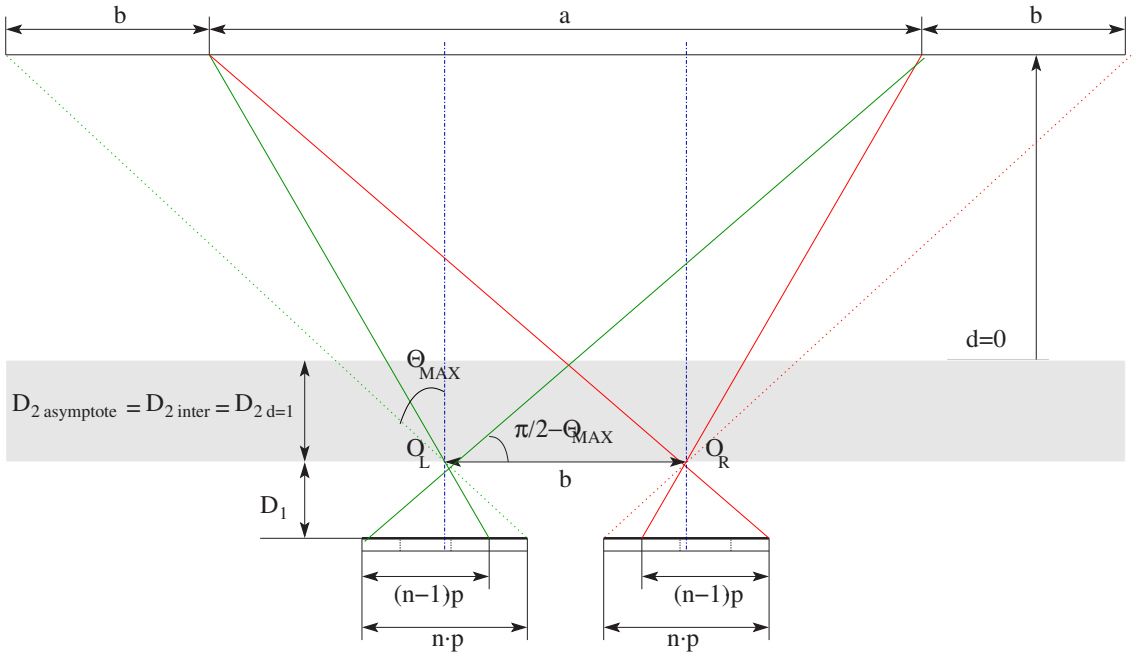


FIGURE 2.3: Situation where the common field of view is represented by a region of $(n-1)$ pixels.

into equation 2.11 gives the best value for the baseline:

$$b_{best} = a \cdot \left(\frac{4 \cdot D_1 \cdot \tan \Theta_{MAX}}{\mu} - 1 \right)$$

Replacing $D_1 \cdot \tan \Theta_{MAX}$ by $\dfrac{D_{chip}}{2}$ and $\tan \Theta_{MAX}$ by $\dfrac{D_{chip}}{2 \cdot D_1}$:

$$\boxed{b_{best} = a \cdot \left(\frac{D_{chip} + p}{D_{chip} - p} \right)} \tag{2.13}$$

This expression gives the baseline, b, providing the best depth accuracy as a function of the common field of view a.

In real cameras:

$$\frac{D_{chip} + p}{D_{chip} - p} \simeq 1 \quad \text{because} \quad D_{chip} = n \cdot p \quad \text{where typically } n \gg 100,$$

thus

$$\boxed{b_{best} \simeq a} \tag{2.14}$$

Thus the optimum baseline should be chosen to be equal to the common field of

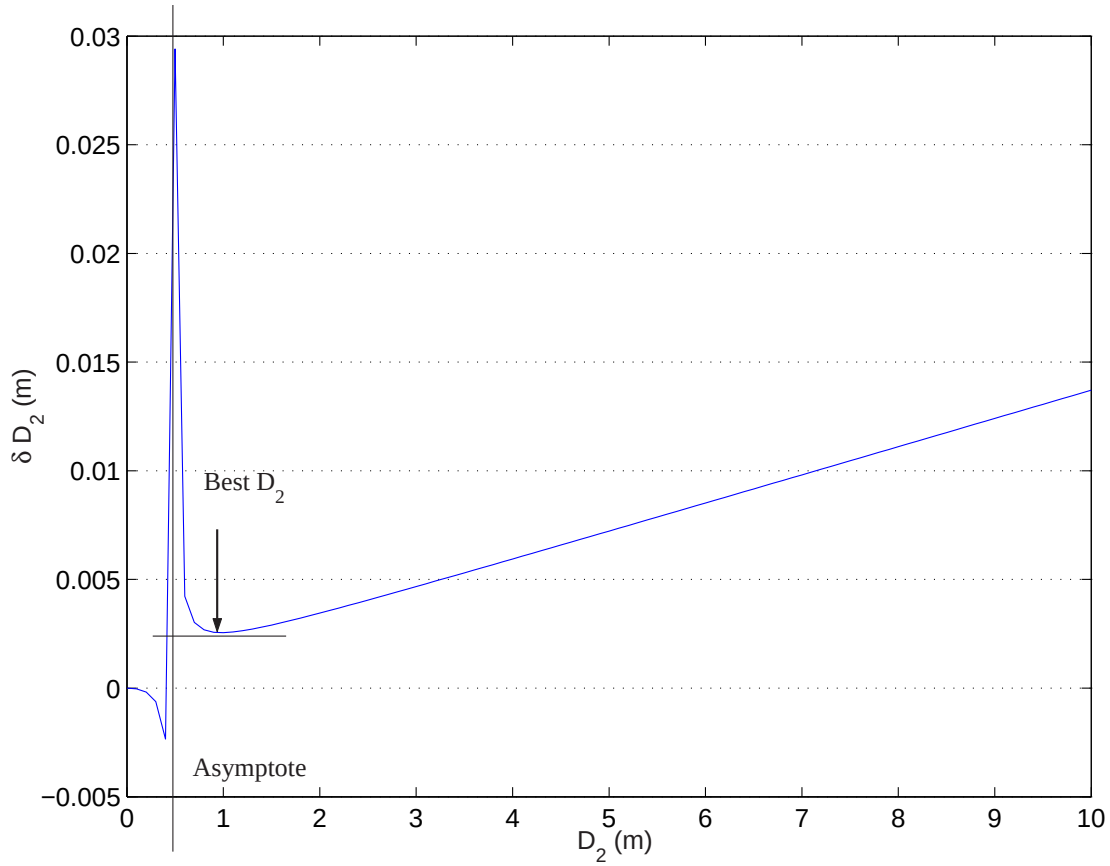


FIGURE 2.4: δD_2 *vs.* D_2
Graph parameters: $D_1 = 15mm$; $a = 0.1m$; $D_2 : 0 \sim 10m$;$p = 4\mu m$; $n = 768$

view, a.

The limiting value of the accuracy obtainable with the optimum configuration is obtained by replacing D_2 by its expansion (equation 2.7):

$$\delta D_{2_{limit}} = \frac{\dfrac{4 \cdot p \cdot a^2 \cdot D_1^2}{\mu^2}}{\left(-\dfrac{4 \cdot a \cdot \lambda \cdot D_1}{\mu}\right) \cdot D_1 + \dfrac{2 \cdot p \cdot a \cdot D_1}{\mu}}$$

$$\boxed{\delta D_{2limit} = \frac{4 \cdot p \cdot a \cdot D_1}{\mu^2}} \tag{2.15}$$

Behaviour of the Accuracy Limit with D_1

For the study of the accuracy limit as a function of D_1, D_2 and b are fixed; subsituting equation 2.2 in equation 2.4, brings D_2 as a function of D_1:

$$D_2 = \frac{(a+b)\cdot D_1}{D_{chip}}$$

Replacing this expression for D_2 as a function of D_1 in equation 2.7 brings:

$$\delta D_2 = \frac{p\cdot(a+b)^2\cdot D_1}{D_{chip}\cdot\big(b\cdot D_{chip} - p\cdot(a+b)\big)} \tag{2.16}$$

p, $(a+b)^2$ and D_1 are positive. Thus the behaviour of δD_2 versus D_1 will be determined by the sign of the denominator in 2.16:

$$D_{chip}\cdot\big(b\cdot D_{chip} - p\cdot(a+b)\big)$$

D_{chip} is positive, so the sign will depend on:

$$b\cdot D_{chip} - p\cdot(a+b) = b\cdot(D_{chip}-p) - a\cdot p = \mu\cdot b - a\cdot p$$

$$\begin{cases} a < \dfrac{\mu\cdot b}{p} \quad \Rightarrow \quad \begin{cases} D_1 \nearrow \quad \Rightarrow \quad \delta D_2 \nearrow \\ D_1 \searrow \quad \Rightarrow \quad \delta D_2 \searrow \end{cases} \\ a > \dfrac{\mu\cdot b}{p} \quad \Rightarrow \quad \begin{cases} D_1 \nearrow \quad \Rightarrow \quad \delta D_2 \searrow \\ D_1 \searrow \quad \Rightarrow \quad \delta D_2 \nearrow \end{cases} \end{cases}$$

Note that $\dfrac{\mu\cdot b}{p}$ is an upper limit for a:

$$\begin{aligned} \text{if} \quad & a > \frac{\mu\cdot b}{p} \\ \text{knowing that} \quad & \delta D_2(D_1 = 0) = 0 \\ \text{brings:} \quad & D_1 \nearrow \quad \Rightarrow \quad \delta D_2 < 0 \end{aligned}$$

This limit on a corresponds to the width of the common field of view placed at the maximum sensible distance D_{2MAX}: computing D_2 for the biggest distance, *i.e.*

the smallest disparity $(1 \cdot p)$, using equation 2.5 brings:

$$D_{2_{MAX}} = \frac{D_1 \cdot b}{1 \cdot p}$$

Replacing $D_{2_{MAX}}$ into the common field of view equation 2.4 gives:

$$\begin{aligned} a &= 2 \cdot D_2 \cdot \tan \Theta_{MAX} - b \\ &= 2 \cdot \frac{D_1 \cdot b}{p} \cdot \tan \Theta_{MAX} - b \\ &= \frac{b}{p} \cdot (2 \cdot D_1 \cdot \tan \Theta_{MAX} - p) \\ a &= \frac{\mu \cdot b}{p} \end{aligned}$$

So the realisable values are:

$$a < \frac{\mu \cdot b}{p}$$
$$D_{2MAX} = \frac{D_1 \cdot b}{p}$$

In real cameras configurations:

μ	defined by $D_{chip} - p$	$\simeq 5 \cdot 10^{-3} m$
b	baseline, several cm	$10^{-2} m \leq b \leq 10^{-1} m$
p	sensor's pixel size, several μm	$\simeq 5 \cdot 10^{-6} m$

If a is low enough (small objects, up to a few hundreds of millimetres), the accuracy (δD_2) has the same behaviour as the focal length (D_1).

Note that a smaller focal length, D_1, can increase the accuracy, but distortion free short focal length lenses are also harder to design and fabricate, see [51] and [52].

Behaviour of the Accuracy Limit with b

When D_1 is fixed, the behaviour of δD_2 against b is determined by substituting D_2 from 2.7 giving:

$$\delta D_2 = \nu \cdot \frac{a^2 + 2 \cdot a \cdot b + b^2}{\mu \cdot b - a \cdot p}$$

where:

$$\nu = \frac{p}{2 \cdot \lambda} \quad \text{is positive because } 0 < \Theta_{MAX} < 90^o$$

Writing $\delta D_2 = \frac{u}{v}$,

$$\text{where} \quad \begin{cases} u = \nu \cdot (a^2 + 2 \cdot a \cdot b + b^2) \\ v = \mu \cdot b - a \cdot p \end{cases}$$

and using the differentiation rule:

$$\frac{\partial \delta D_2}{\partial b} = \frac{u' \cdot v - u \cdot v'}{v^2}$$

where u' and v' are the derivatives of u and v with respect to b:

$$\begin{cases} u' = 2 \cdot \nu \cdot (a + b) \\ v' = \mu \end{cases}$$

The derivative of δD_2 with respect to b is:

$$\frac{\partial \delta D_2}{\partial b} = \nu \cdot \frac{\mu \cdot b^2 - a^2 \cdot (2 \cdot p + \mu) - 2 \cdot a \cdot p \cdot b}{v^2} \tag{2.17}$$

Because $\nu > 0$ and $v^2 > 0$, we consider the sign of:

$$(\mu \cdot b^2 - a^2 \cdot (2 \cdot p + \mu) - 2 \cdot p \cdot a \cdot b)$$

This is a second order equation in b with roots:

$$\begin{cases} b_1 = -a \\ \\ b_2 = a \cdot \left(\dfrac{4 \cdot D_1 \cdot \lambda}{\mu} - 1 \right) \end{cases}$$

Studying the sign of 2.17 between its zeros, replacing v^2 by its value in 2.17 leads to:

$$\frac{\partial \delta D_2}{\partial b} = \nu \cdot \frac{\mu \cdot b^2 - a^2 \cdot (2 \cdot p + \mu) - 2 \cdot a \cdot p \cdot b}{(\mu \cdot b - a \cdot p)^2}$$

For values of b:

- If $b \to \pm\infty$, only the highest terms of the polynomial in b are kept, leading to:

$$\frac{\partial \delta D_2}{\partial b} = \nu \cdot \frac{1}{\mu} > 0$$

- If $-a < b < a \cdot \left(\frac{4 \cdot D_1 \cdot \lambda}{\mu} - 1\right)$, i.e. $b = 0$ for instance:

$$\frac{\partial \delta D_2}{\partial b} = -\nu \cdot \frac{2 \cdot p + \mu}{p^2} < 0$$

- If $b = \frac{a \cdot p}{\mu}$, the denominator is zero and we have a vertical asymptote.

Sign on both sides of the vertical asymptote:

left of b: $\quad b < \frac{a \cdot p}{\mu} \qquad b \cdot \mu < a \cdot p \qquad \delta D_2 < 0$

right of b: $\quad b > \frac{a \cdot p}{\mu} \qquad b \cdot \mu > a \cdot p \qquad \delta D_2 > 0$

Leading to the following table for all values of b:

b	$-\infty$	$-a$		$a \cdot \left(\frac{4 \cdot D_1 \cdot \lambda}{\mu} - 1\right)$	$+\infty$
$\frac{\partial \delta D_2}{\partial b}$	$+$	0	$-$	0	$+$
$\delta D_2(b)$	$\nearrow$		$\searrow$		$\nearrow$

TABLE 2.2: Behaviour of the accuracy limit δD_2 *vs.* b

A curve constrained to real cases ($b > 0$) is illustrated in figure 2.2.3:

$$\begin{cases} b < a \cdot \left(\frac{p + 2 \cdot D_1 \cdot \lambda}{2 \cdot D_1 \cdot \lambda - p}\right) & \begin{cases} b \nearrow & \Rightarrow \delta D_2 \searrow \\ b \searrow & \Rightarrow \delta D_2 \nearrow \end{cases} \\ b > a \cdot \left(\frac{p + 2 \cdot D_1 \cdot \lambda}{2 \cdot D_1 \cdot \lambda - p}\right) & \begin{cases} b \nearrow & \Rightarrow \delta D_2 \nearrow \\ b \searrow & \Rightarrow \delta D_2 \searrow \end{cases} \end{cases} \tag{2.18}$$

Note that the same value for b as in equation 2.13 is obtained. This shows that the best baseline is just a bit smaller than the common field of view a. Accuracy can be increased by moving the baseline b towards this value, but choosing a bigger baseline does not necessarily imply getting a better accuracy.

Both these expressions need to be used to determine whether target accuracy can be reached for the values of the parameters a, D_1 and b, set by any given problem.

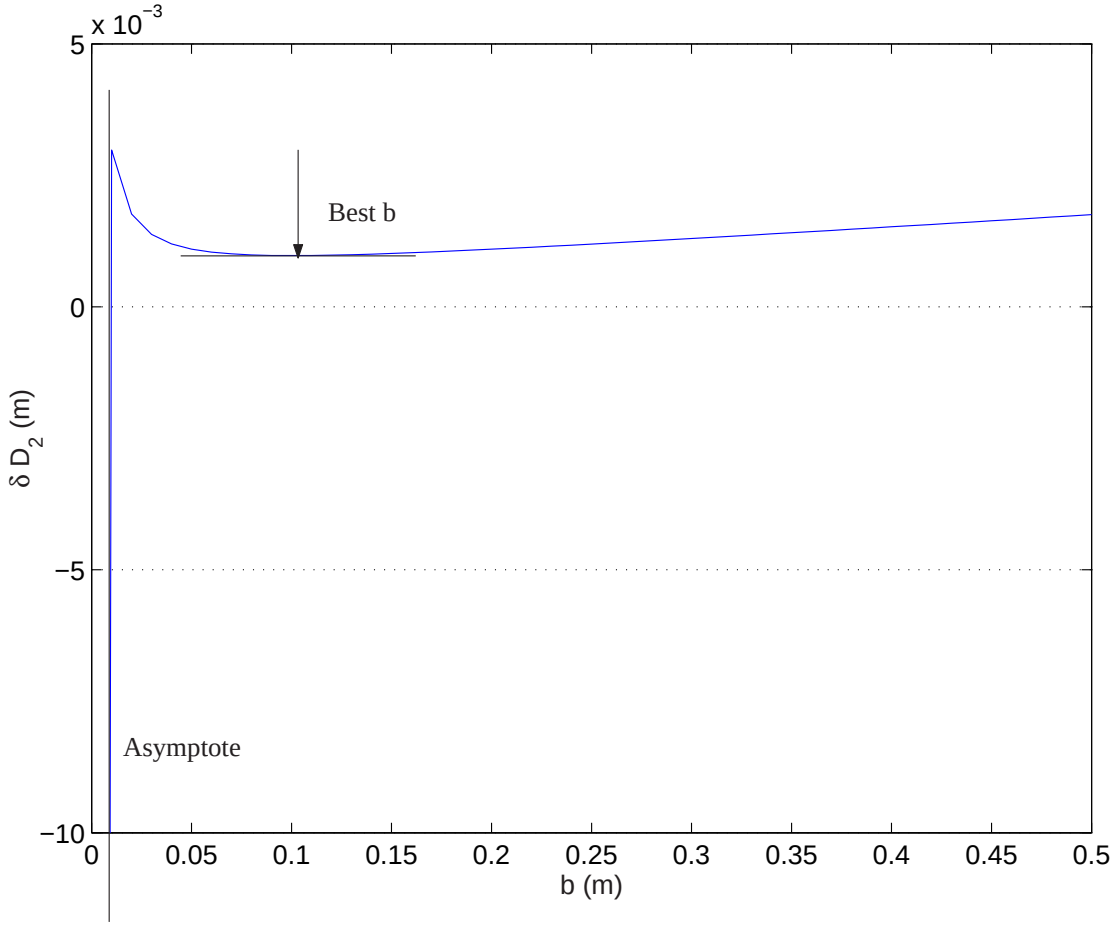


FIGURE 2.5: δD_2 *vs.* b
Graph parameters: $D_1 = 15mm$; $a = 0.1m$; $b : 0 \sim 0.5m$; $p = 4\mu m$; $n = 768$ pixels

2.2.4 Conclusion

Equations 2.11, 2.13 and 2.15 describe the behaviour for a given camera setup, thus making possible to create a family of curves for different D_1 *i.e.* Θ_{MAX} values to chose an acceptable camera configuration. Having found the best parameters set it might be possible to modify the optimum (D_2, b) pair to suit the other experimental constraints.

The accuracy behaviour has been described:

1. *vs.* the camera's focal length D_1 in equation 2.17,
2. *vs.* the baseline b in equation 2.18, and
3. *vs.* the distance D_2 in equation 2.9.

2.3 Non-Parallel Camera Axes

2.3.1 Introduction

The previous section described the case where the two camera axes are *parallel*. This section briefly describes the situation where the two camera focal axes are *not parallel*, it is discussed comprehensively by Mallot [53], so this section simply highlights the principal differences relevant to matching algorithms: the presence of both positive and negative disparities.

2.3.2 Vergence and Vieth-Müller circle

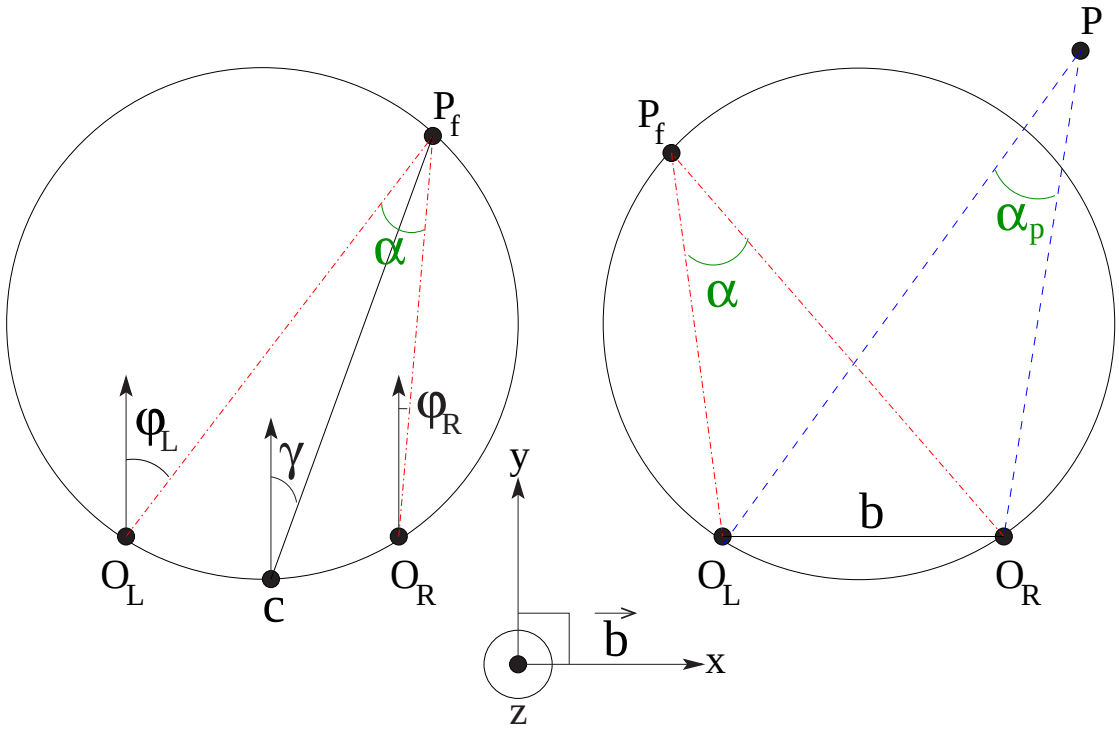


FIGURE 2.6: Left: Axes and fixation point P_f, Middle: Cartesian reference system, Right: Position of a scene point P while fixating P_f.

Figure 2.6 illustrates the axis, fixation point P_f and scene point P configuration:

- P_f is the fixation point, *i.e.* the point where the two camera axis cross,
- O_L, O_R are the centres of the cameras,
- b is the baseline, *i.e.* the distance between the two cameras,

- φ_L, φ_R are the azimuth angles of the cameras, *i.e.* the viewing directions in a head-centered system,
- α, α_f are the vergences of P and P_f
- γ is the version,
- P is a point seen while fixating P_f.

The Cartesian coordinate system is defined by:

- x: the baseline vector ($\overrightarrow{b}$) between the two camera centres, O_L and O_R,
- y: the heading direction normal to the baseline and
- z: a vertical axis upward.
- so that O_L and O_R coordinates are $O_L(-b/2, 0, 0)$ and $O_R(b/2, 0, 0)$

and the Hering vergence and versions are respectively:

- $\alpha = \varphi_L - \varphi_R$
- $\gamma = \frac{1}{2} \cdot (\varphi_L + \varphi_R)$

Note that a vergence, $\alpha = 0$, corresponds to the case where the two cameras optical centres are the same and is not stereovision anymore. In the parallel optical axes case, α is undefined as the two axes do not cross anymore.

For each positive value of α, a Vieth-Müller circle of vergence α is defined by:

- centre: $[\frac{b}{2} \cdot \cot \alpha, 0, 0]$
- radius: $\dfrac{b}{2 \cdot sin\alpha}$

The Vieth-Müller circles are also referred to as iso-vergence lines.

Using β_L and β_R, the azimuth angles of point P in a camera-centered coordinate system, the disparity of a scene point, P, may also be defined by the angular difference:

$$\boxed{\delta = \beta_L - \beta_R} \tag{2.19}$$

Figure 2.7 shows the correspondence between the disparity in angular units ($\delta = \beta_L - \beta_R$) and the disparity in pixels with is the difference in length of the two portions of the camera sensor (d_L and d_R) subtended by the two angles β_L and β_R, P_f is the fixation point and P is a user sample point.

In Figure 2.7 shows three points, P_1, P_2 and P_3 at increasing depths with angular disparities:

- $\delta_1 = \beta_{L1} - \beta_{R1}$
- $\delta_2 = \beta_{L2} - \beta_{R2}$
- $\delta_3 = \beta_{L3} - \beta_{R3}$

In figure 2.7, the two angles labelled λ_2 are equal (λ_1 and λ_3 have not been represented for clarity). P_2 is on the iso-vergence circle containing P_f, then:

$$\alpha_2 = \alpha_f$$

Knowing:

$$\alpha_f + \lambda_2 + \beta_{L2} = \alpha_2 + \lambda_2 + \beta_{R2}$$

leads to:

$$\beta_{L2} = \beta_{R2}$$

i.e. disparities of points on the iso-vergence circle passing through the fixation point P_f have for disparity:

$$\boxed{\delta_2 = \beta_{L2} - \beta_{R2} = 0}$$

Since

$$\beta_{L1} > \beta_{L2} > \beta_{L3}$$

and

$$\beta_{R1} < \beta_{R2} < \beta_{R3}$$

we must have:

$$\delta_1 > \delta_2 > \delta_3$$

and as $\delta_2 = 0$, we have:

$$\boxed{\delta_1 > 0 \qquad \text{and} \qquad \delta_3 < 0}$$

or negative as well as positive disparity values will be present.

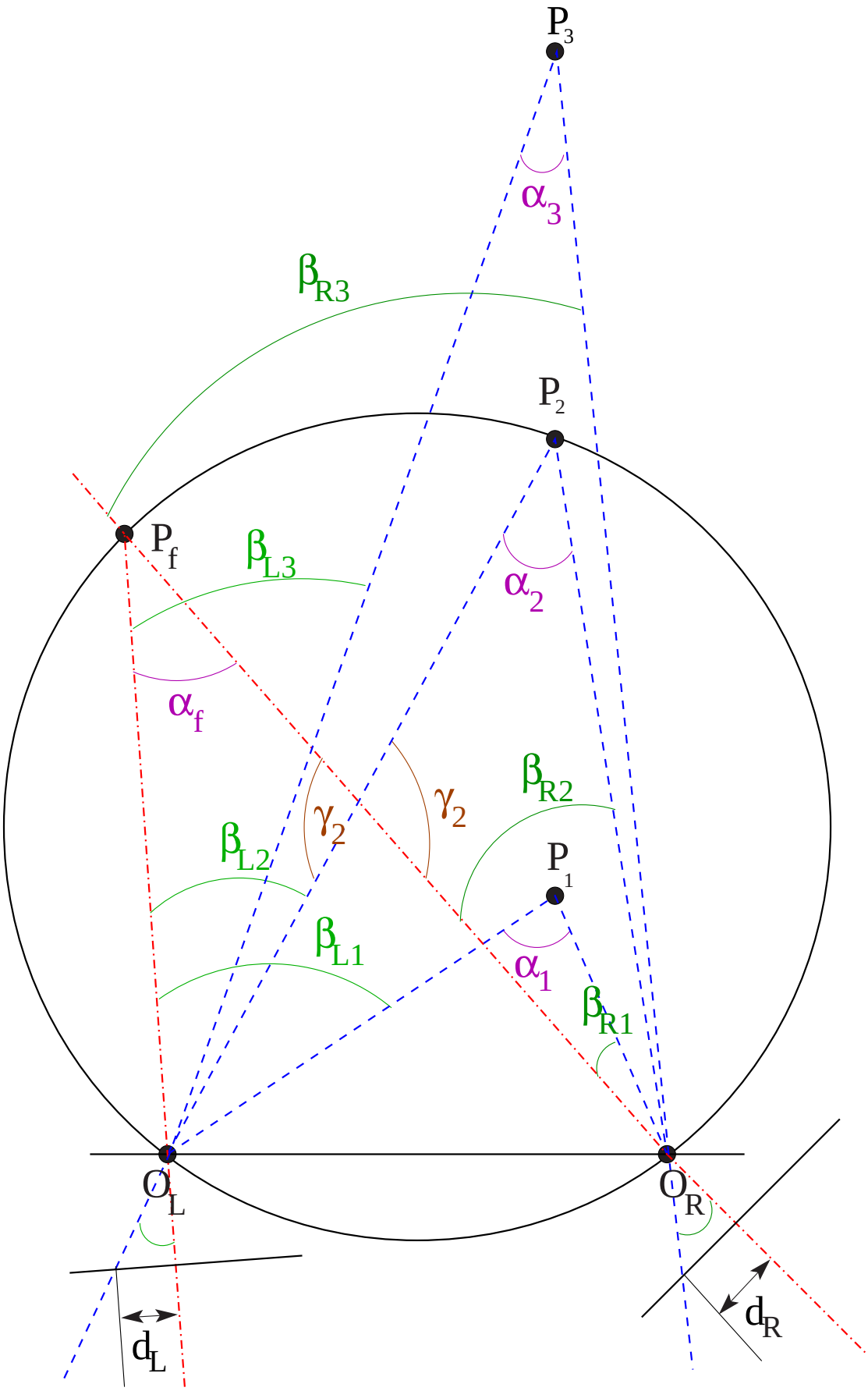


FIGURE 2.7: Disparity for 3 different cases while fixating P_f: a close point P_1, a point on the fixation circle P_2 and a far point P_3

2.4 Biprism study

2.4.1 Introduction

Lee *et al.* proposed the use of a biprism so that stereopairs can be captured by a single camera [1]. A biprism placed in front of a lens and centred on the camera's optical axis bends the rays impinging on the left and right halves of the image plane to produce a pair of images similar to those obtained with two cameras whose optical axes are not parallel.

2.4.2 Equivalent Stereo System

Figure 2.8 shows how a scene point, P, is imaged onto two positions on the image plane so that it appears to be both at P_L and P_R in the scene. The deviation angle of the biprism, δ_{Na}, is a function of n[3], the refractive index of the biprism material, and α, the angle at the base of the biprism. The configuration is illustrated in figure 2.8 (adapted from Lee *et al.* [1]). From Snell's law:

$$n = \frac{\sin(\frac{\alpha + \delta_{Na})}{2})}{\sin(\frac{\alpha}{2})}$$

For BK7 (a commonly used glass), the refractive index, $n = 1.52$. α depends on the cut of the biprism and is also known, so:

$$\delta = 2 \cdot \arcsin \cdot \Big(n \cdot \sin\big(\frac{\alpha}{2}\big) \Big) - \alpha$$

The equivalent baseline, b_{eq}, is a function of the distance between the optical centre of the camera and the biprism, t_z, and δ_{Na} [1]:

$$b_{eq} = 2 \cdot t_z \cdot \tan \delta_{Na}$$

[3]Usually refractive indices are measured using a sodium (Na) lamp whose dominant wavelength is $589nm$.

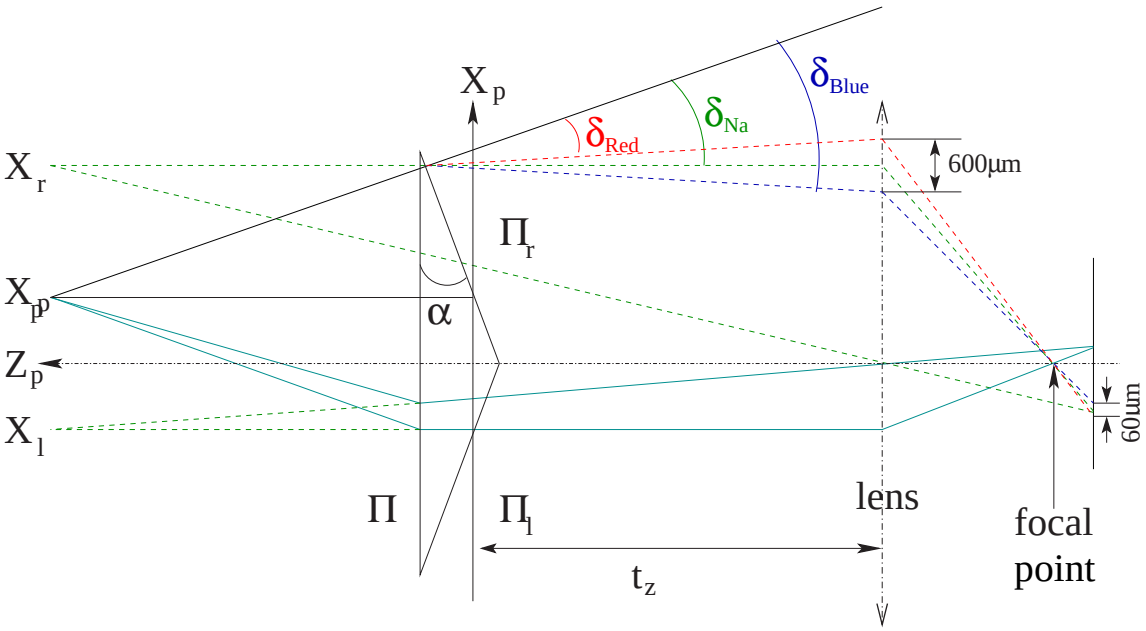


FIGURE 2.8: Biprism configuration adapted from [1]

As an example, following Lee *et al.* and using the values:

$$\alpha = 12.4^o$$
$$\delta_{Na} = 6.6^o$$
$$t_z = 152mm$$

leading to:

$$b_{eq} = 35mm$$

From 2.13, the best accuracy is achieved when a and b are almost equal, so that this setup limits us to small objects. For larger objects, say, $a = 300mm$, start with $b_{eq} = 300mm$:

$$b_{eq} = 2 \cdot t_z \cdot \tan \delta_{Na} = 300mm$$

For a 12.4^o biprism, $t_z = 1283mm$, which, with a camera view angle of 25^o, would require an impractically large prism of $1194mm$. For a 45^o prism, $t_z = 306mm$, which still requires a prism of $285mm$.

A biprism also separates colours. For 'higher dispersion crown', the refractive index varies between 1.533 for a wavelength of $434nm$ and 1.514 for $768nm$ (see [54]) . For a prism with $\alpha = 12.4^o$, δ ranges from $\simeq 6.66^o$ to $\simeq 6.42^o$. At a distance $t_z = 152mm$ the separation is $17.7mm - 17.1mm = 600\mu m \simeq 120$ pixels on a camera with $5\mu m$ pixels. The $600\mu m$ dispersion is at the position of the lens, which

is a few cm in size: the image sensor is a few mm in size so the ratio is roughly one tenth. Thus the dispersion on the image plane is $\simeq 60\mu m = 12$ pixels, *i.e.* the dispersion per colour components is $\simeq 4$ pixels.

This means that a black and white camera will see a thoroughly distorted view of a multi-coloured object and therefore this technique will only work for objects with a narrow spectral range.

For a colour camera, each colour component would be spread by about a third of this, *i.e.* $\sim 200\mu m$. Even though it would be possible to shift the colour components by a known amount to align the centres of the R, G and B bands, the spread over one colour band will still be several pixels.

2.4.3 Accuracy

For a realisable biprism system, this section calculates the accuracy that could be achieved. Choosing values for camera A used in Chapter 4:

$$\begin{aligned} p &= 4\mu m \\ D_1 &= 15mm \\ b_{eq} &= 35mm \quad \Rightarrow \quad a = 35mm \\ &\Rightarrow \quad D_2 = \frac{2 \cdot a \cdot D_1}{\mu} = 342mm \\ &\Rightarrow \quad \delta D_2 = \frac{p \cdot D_2^2}{b \cdot D_1 - p \cdot D_2} = 0.9mm \end{aligned}$$

However due to the prism dispersion, the effective p is increased to 4 pixels which leads to $\delta D_2 = 3.6mm$ or $\sim 10\%$ of a $35mm$ object.

2.4.4 Discussion

The interest in this approach is:

- the seemingly easier calibration of the system,
- noise might be limited by taking the two images with the same camera, as opposed to using two separate sensors,
- an easier epipolar configuration as one only needs to set up the biprism perpendicular to the scanlines once,
- the baseline can be changed by moving the camera relative to the biprism only.

On the other hand:

- the main limitation of the system resides in the fact that it can only handle near objects and with a small equivalent baseline with a rather small field of view,
- the sample calculations show that obtaining reasonable baselines needs a large biprism,
- once two camera are positioned carefully on a rigid base the baseline can be easily be modified (conserving the cameras alignment) using a precise translater device,
- the angle δ is fixed, it depends on the biprism physical characteristics which makes it more difficult to change settings: especially the *field of view* is fixed and the only adjustable parameter is t_z whereas with a dual camera configuration, it is easy to change camera parameters, including the baseline and the angle between the camera axes,
- the dispersion problem decreases significantly the accuracy of the system for a given baseline and camera view angle.

Overall, since the main advantages of a bi-prism based system can also be provided by careful alignment and calibration of a pair of cameras, lower accuracy and dispersion-induced distortions would generally make these systems unattractive.

Part II

Experiments

Chapter 3

Experiments: Introduction

3.1 Introduction

This chapter introduces several stereo algorithms: the ones used in this study as well as other important algorithms described in the literature. The choice of algorithms for this study has been motivated by the initial goal of this study, *i.e.* hardware implementation suitable algorithms. An overview of the code used is also given, followed by a description of the metrics and different stereo pairs used for the comparisons.

3.2 Algorithms

This section presents firstly the algorithms implemented for this study, secondly several variants of them and lastly some other key algorithms. Two main classes of algorithms are presented: correlation based and dynamic programming. My own variants will be introduced in the corresponding experimental chapters.

Binocular stereo has the following definitions:

- binocularly visible points are seen by both right and left camera,
- monocularly visible points are seen by either the right or the left camera and
- occluded points belong to the scene but cannot be seen by any of the two cameras.

A *partial occlusion* is the exact equivalent to a monocularly visible point. However, completely occluded points are not used by algorithms as they cannot be seen at all, so commonly *occlusion* refers to a monocularly visible point, *i.e.* a partial occlusion.

All these algorithms assume the *ordering constraint*, *i.e.* for a pair of matching pixels, all the pixels on one side of a pixel are on the same side of the corresponding pixel on the other image. This assumes mostly continuous surfaces: the classic exception to this constraint is a scene with a pole in front of a background.

3.2.1 Correlation based algorithms

Firstly three standard correlation based variants are presented: two from Faugeras *et al.* [55] and a third one which was simplified so that it was capable of an efficient (small and fast) hardware implementation: the sum of absolute differences (SAD). The Census algorithm has been considered with the correlation algorithms because computation of the Hamming distance on the computed Census transform vectors is similar to a correlation function.

Notation

$I(P)$ is the intensity at point, P: $I_L(P)$ is an intensity in the left image and $I_R(P)$ in the right one. Correlation functions are evaluated over a 'window' of neighboring pixels in each image. A window, $w(P, r)$, is defined by its centre, P, and its radius, r. A radius, r implies a square window of $(2 \cdot r + 1) \times (2 \cdot r + 1)$ pixels.

Figure 3.1 illustrates the correlation process: for a point P on the reference image (left for instance), all correlations with a window in the right image for the whole disparity range are computed and the best value is chosen.

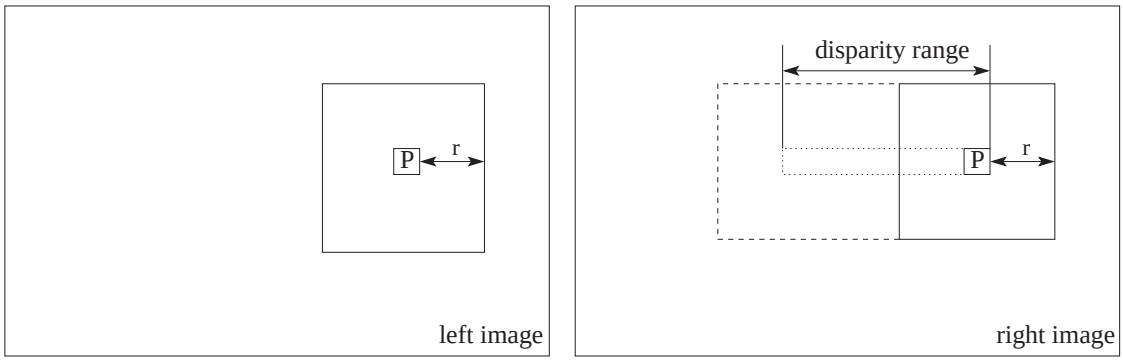


FIGURE 3.1: Correlation between a left and a right image at point P: the best correlation between windows in left and right images is searched for all the values of the disparity.

Corr1: Normalised Square of Differences Correlation (C_1 ***in Faugeras** et al. **[55]***

The first correlation cost, $C_1(P)$, is the normalised intensity difference:

$$C_1(P) = \frac{\sum_{P' \in w(P,\beta)} (I_L(P') - I_R(P'))^2}{\sum_{P' \in w(P,\beta)} I_L(P')^2 \cdot \sum_{P' \in w(P,\beta)} I_R(P')^2} \tag{3.1}$$

Corr2: Normalised Multiplicative Correlation (C_2 ***in Faugeras** et al. **[55])***

The second correlation cost, $C_2(P)$, is a normalised multiplicative correlation function:

$$C_2(P) = \frac{\sum_{P' \in w(P,\beta)} I_L(P') \cdot I_R(P')}{\sqrt{\sum_{P' \in w(P,\beta)} I_L(P')^2} \cdot \sqrt{\sum_{P' \in w(P,\beta)} I_R(P')^2}} \tag{3.2}$$

Sum of absolute differences (SAD)

The third cost simply sums absolute differences without normalisation. This is naturally much faster and has a simple hardware implementation:

$$SAD(P) = \sum_{P' \in w(P,\beta)} |I_R(P') - I_L(P')| \tag{3.3}$$

A variant of the SAD cost is the Sum of Squared Differences (SSD):

$$SSD(P) = \sum_{P' \in w(P,\beta)} (I_R(P') - I_L(P'))^2 \tag{3.4}$$

For hardware implementations, SAD is easier to implement as it only needs a substraction, a comparison, and a possible sign change: SSD needs a substraction and a multiplication. In hardware, a parallel-array multiplier requires O(n)^2 space[1] and a propagation delay approximately twice that of a substraction which requires $O(n)$ space, see Hamacher [56].

Discussion

Normalisation factors were introduced into the first two algorithms to allow for 'real' images taken with, for example, different gain settings. However, as my experiments show, normalisation makes negligible contribution to the matching quantity. Centred variants of the correlation have also been studied by Faugeras *et al.* [55]:

[1]Bit serial multipliers can be built but are very slow - they trade speed for space.

they claim that normalised multiplicative correlation (C_2) has similar performance to the centred versions (C_3 and C_4 the centred normalised difference and centred normalised multiplicative correlations respectively) and perform better than the C_1 criterion. Figure 3.3 shows three correlation variants with the same window radius tested against luminosity difference: the left image was not modified while a fraction of the right image intensity was used ($5\% \sim 100\%$). An illustration of right images with different luminosity levels is given in figure 3.2

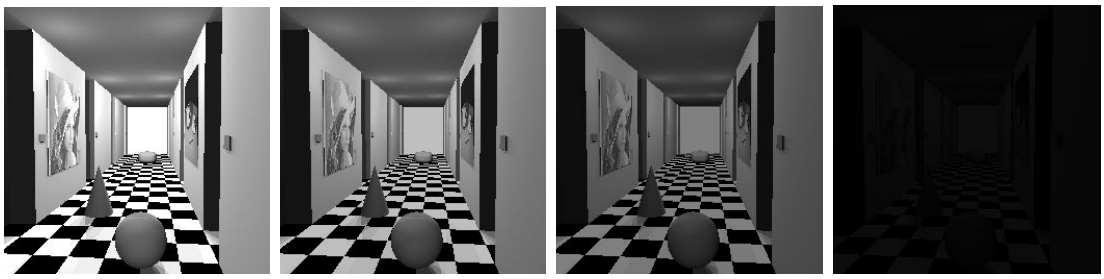


FIGURE 3.2: Exemples of different luminosity levels for the right image of the corridor stereopair, illustrated levels are, from left to right: 100%, 75%, 50% and 5%.

Note that the Corridor images were generated by ray-tracing so that the initial luminosities of the left and right images are as accurate as possible.

Figure 3.3 shows that SAD and Corr1 have fewer good matches with the increasing intensity difference whereas Corr2 maintains a stable percentage of good matches. However, the normalised multiplicative correlation function (Corr2) is much more complex than the SAD function and is not justified in the case of carefully calibrated cameras and especially in the case of a sliding camera taking two shots at two different positions.

The crossing point between SAD and Corr2 - marked 'A' in figure 3.3 - shows that up to a few percent of luminosity difference SAD performs significantly better in the case where the cameras are carefully calibrated.

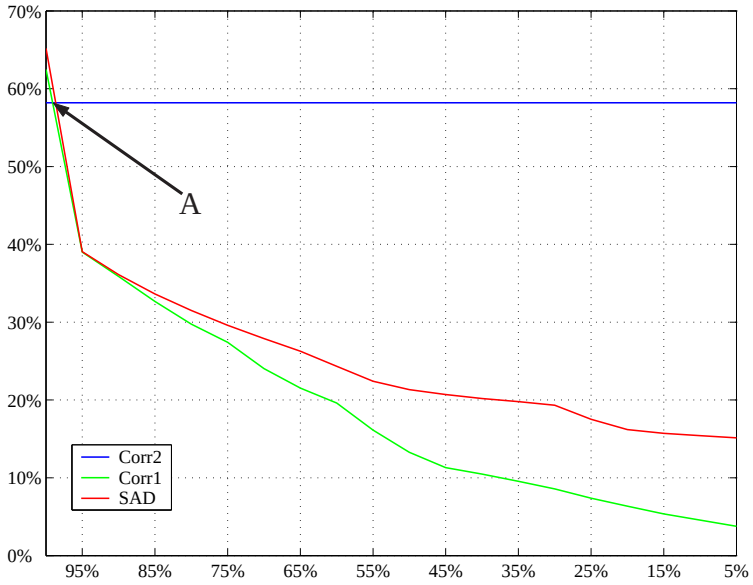


FIGURE 3.3: Plot of the percentage of good matches for Coor1, Corr2 and SAD *vs.* percentage of the original luminosity in the right image, the left image remaining unmodified.

3.2.2 Census algorithm

Zabih and Woodfill describe both the Rank and the Census transforms in [57] and [33]. Lan and Mohr [2] - see section 3.2.4 - working on the original Rank and Census transforms as well as on a modification of Zabih and Woodfill algorithm, claim that the Census version performed better than the Rank one, so in this study only the Census algorithm was implemented:

1. For each pixel, P, in an image, define the **Census transform**, T, consisting of bits describing the relationship between that pixel and those in a window, $w(P, \beta)$, of radius, β, surrounding point, P:

$$T(P, \beta) = \bigotimes_{P' \in w(P,\beta)} \xi(P, P')$$

   where $\otimes$ is a concatenation operator. Each bit of T is defined:

$$\xi(P, P') = \begin{cases} 1 & \text{if } I(P) < I(P'), \\ 0 & \text{otherwise.} \end{cases}$$

   $\xi(P, P')$ represents the ordering (relative intensity) of a pixel at P', relative to a central pixel at P.

   The Census vector, $T(P, \beta)$, is formed over a $(2\beta+1) \times (2\beta+1)$ 'inner' window (see figure 3.4) and thus has a length of $4\beta^2 + 4\beta$ bits - ignoring the central pixel which is always 0 and may be omitted from calculations.

2. Form a **Census vector** by concatenating transforms over an 'outer' window of radius α, (see figure 3.5):

$$V(O, \alpha) = \bigotimes_{P \in w(O,\alpha)} T(P, \beta)$$

   which is zero when the surroundings of both pixels are identical.

3. For each possible value of the disparity, δ, compute the correlation between the Census vectors:

$$C(\delta, w(O_L, \beta)) = V(O_L, \alpha) \ominus V(O_R, \alpha)$$

   where $O_L = (x, y)$ and $O_R = (x - \delta, y)$ are in the same scanline. $\ominus$ is the Hamming distance operator, *i.e.* $C(\delta, w(O_L, \beta))$ is computed by counting bits which differ in the bit vectors, $V(O_L, \alpha)$ and $V(O_R, \alpha)$.

4. Return the disparity value at which $C(\delta, w(O_L, \beta))$ is a minimum.

Note that since the major operations are simply comparing intensities and counting bits, the Census algorithm is potentially very simple and efficient to implement in custom hardware, *e.g.* using FPGA technology [58]. A hardware version has been implemented by Woodfill *et al.* in [59] and [60] and achieves 42 frames per second for a 320×240 pixels image with a maximum disparity of 24 pixels. It has been implemented on the PARTS computer made of 16 Xilinx 4025 FPGAs and 16 one-megabyte SRAMs.

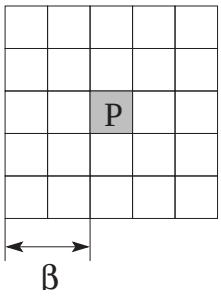


FIGURE 3.4: Census algorithm: window over which the transform, $T(P, \beta)$, is computed, *i.e.* the inner window.

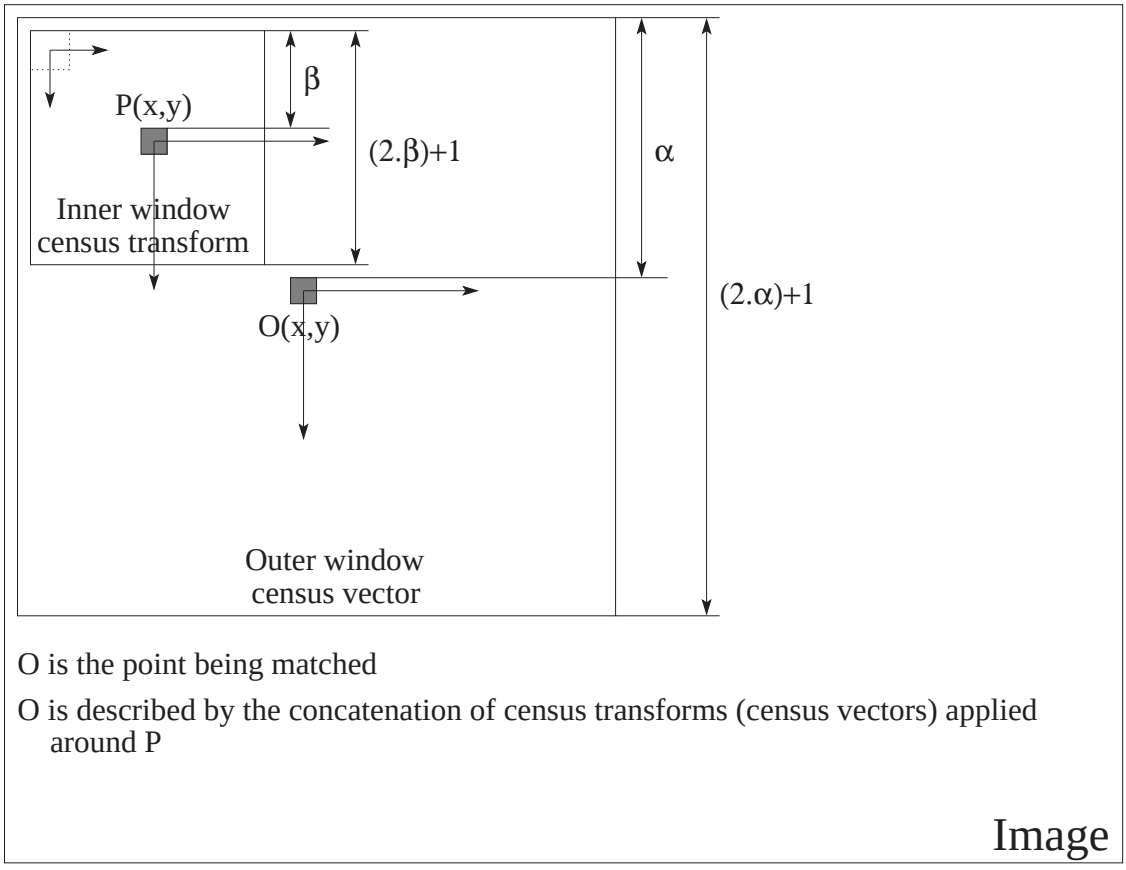


FIGURE 3.5: Inner and outer window used by the Census algorithm

3.2.3 Dynamic Programming algorithm: Pixel-to-Pixel

Birchfield and Tomasi's Pixel-to-Pixel algorithm - in [61] and [62] - was chosen for its ability to find occlusions. A dynamic programming algorithm, it minimises a cost function to find the best path through all possible solutions. The cost function is:

$$\gamma(M) = N_{occ} \cdot \kappa_{occ} - N_m \cdot \kappa_r + \sum_{i=1}^{N_m} d(x_i, y_i)$$

where:

- x_i is the index in the left image and y_i the index in the right image (using the epipolar constraint only indices on the same lines are needed),
- N is the number of occlusions (N_{occ}) or matches (N_m), κ is the cost for an occlusion (κ_{occ}) or reward for a match (κ_r) and
- $d(x_i, y_i)$ is the dissimilarity measure.

The dissimilarity measure - see also Birchfiel *et al.* [63] - is used to decide whether $I_L(x_i)$ and $I_R(y_i)$ are images of the same scene point. It is defined by:

$$d(x_i, y_i) = \min(\overline{d}(x_i, y_i, I_L, I_R), \overline{d}(y_i, x_i, I_R, I_L))$$

where:

$$\overline{d}(x_i, y_i, I_L, I_R) = \min_{y_i - \frac{1}{2} \leq y \leq y_i + \frac{1}{2}} |I_L(x_i) - \widehat{I_R}(y)|$$

where $\widehat{I_R}(y)$ is the linearly interpolated greyscale value.

The algorithms also looks for intensity gradients to find depth discontinuities. An intensity gradient is defined as a minimum intensity jump over a minimum number of pixels, for instance a gradient of at leat 5 grey levels over 3 pixels. A pixel x in the left scanline is said to lie to the left of an intensity gradient if the intensity variation between $x + 1$, $x + 2$ and $x + 3$ is at least 5 grey levels.

Figure 3.9 illustrates the definitions of left and right occlusions: the white dots correspond to occlusion positions. It is assumed that an intensity gradient corresponds to each depth discontinuity - see figure 3.6 - and an occlusion lies on the same side of its intensity gradient, *i.e.*:

- If $(x_i, ..., x_j)$ is a left occlusion, then x_j lies to the left of an intensity gradient $(1 \leq i \leq j < k)$.
- If $(y_i, ..., y_j)$ is a right occlusion, then y_j lies to the right of an intensity gradient $(1 \leq i \leq j < k)$.

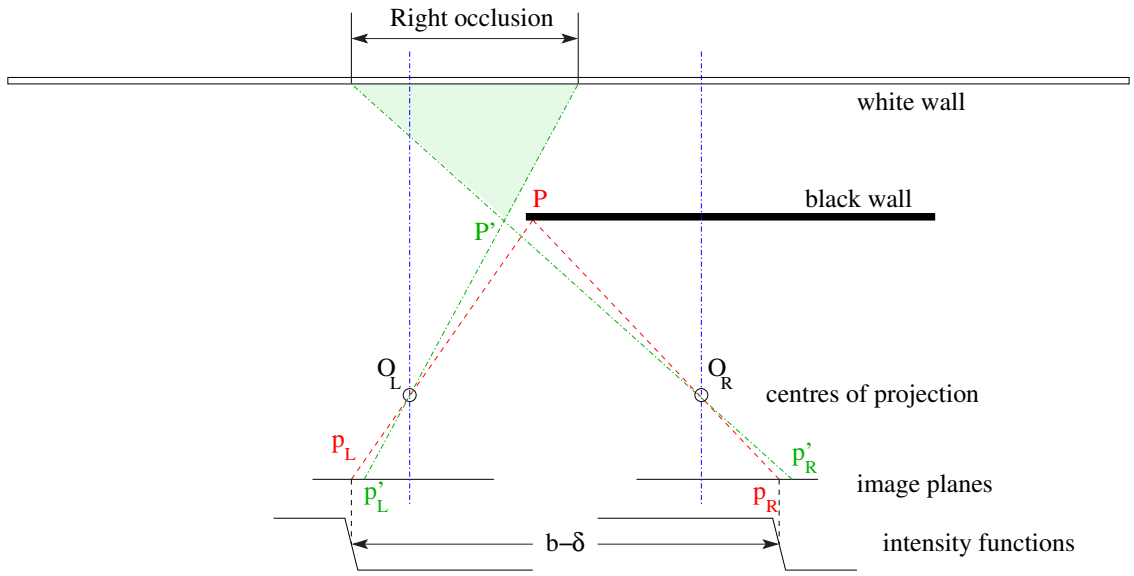


FIGURE 3.6: Pixel-to-Pixel greyscale gradients illustration.

Figure 3.7 shows a diagram of the cost array, where the cost is calculated for each (x, y) cell, where x is the pixel position on the current epipolar line in the left image and y is the pixel position on the same line in the right image. The black cells represent cells that do not need to be computed: when the maximum disparity is known (3 in the example implies $\delta = y - x \leq 3$).

For efficient storage, the array in figure 3.7 can be transformed to a dense array (see figure 3.8) indexed by (δ, y) where y is the pixel position on the current epipolar line in the reference image and δ ranges over all possible disparities. The computed costs are saved in an array from which the lowest cost path is inferred, it has the length of a scanline and the height of the maximum possible disparity.

The output of the cost array is a **match sequence**: an example appears in figure 3.9, which shows the matching points for a line in the left and the right images. Occlusions are marked by the white dots: when a match occurs, a line links pixels from left and right images: the disparity is proportional to the slope of this line.

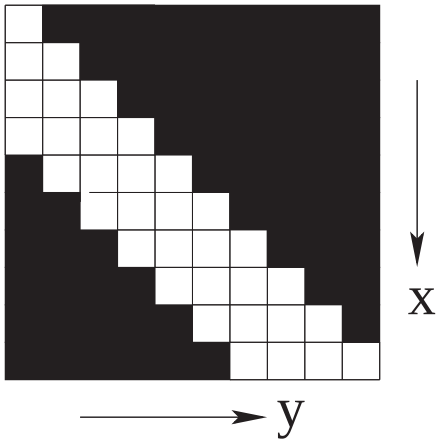


FIGURE 3.7: Pixel-to-Pixel cost array (x,y)

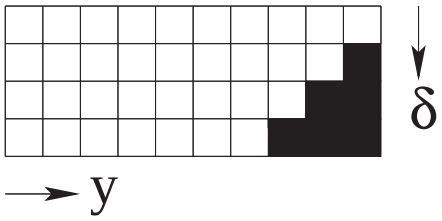


FIGURE 3.8: Pixel-to-Pixel reduced cost array (δ, y)

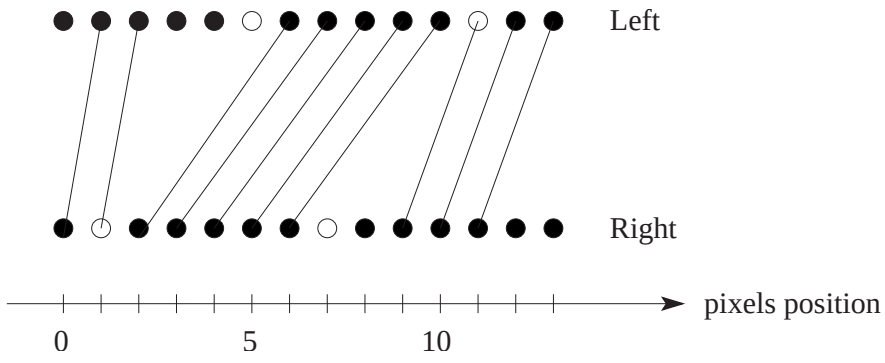


FIGURE 3.9: Pixel-to-Pixel match sequence. The white dots show where occlusions are found.

3.2.4 Other algorithms

Introduction

Recently, Scharstein and Szeliski - augmenting their first taxonomy [44] - have produced an extensive one with an objective comparison of more than thirty algorithms in [43]. Their work is based on four sets of stereo pairs: Map, Sawtooth, Tsukuba and Venus, all of which were used in this study also for consistency and are described in section 3.5.

In their evaluation, they used the root mean square, RMS, of the error, between the computed disparities $d_C(x, y)$ at position (x, y) and the ground truth disparities $d_T(x, y)$, given as:

$$R = \frac{1}{N} \cdot \sum_{x,y} \Big(|d_C(x, y) - d_T(x, y)|^2 \Big)^{\frac{1}{2}}$$

The percentage of bad matching pixels was defined as:

$$B = \frac{1}{N} \cdot \sum_{x,y} |d_C(x, y) - d_T(x, y)| > \delta_d$$

where δ_d was chosen as 1 pixel. Bad matches were also classified as:

- bad matches for non occluded zones,
- bad matches for textureless zones and
- bad matches for depth discontinuities.

The comparisons of correlation based algorithms takes into account a *truncation* value taken between 1 and ∞ *i.e.*no truncation. Scharstein and Szeliski introduced truncation in 1996 [64], [65] and [66]; they claim it mostly helps for discontinuities where a window has pixels from the near and the far object. Truncation fixes a maximum threshold to the matching cost (the correlation value for instance) which limits the influence of wrong matches. A good range is given to be between 5 and 50 usually around 20 and has to be just larger than the level of noise. They compared SAD and SSD algorithms with different truncation levels, with and without the use of Birchfield and Tomasi's sampling insensitive dissimilarity measure [63] as well as using a 9×9 min window.

Note that a 9×9 window corresponds to a radius of 4, which is the optimum window size for the correlation algorithms as shown in section 5.2 - figure 5.1 shows that radii larger than 4 do not significantly improve the percentage of good matches.

In most cases the difference between SAD and SSD is negligible. Typical bad matches rates are between a few percent and 60%: the best results are seen for truncation values[2] around 20 or a bit less.

Including Birchfield and Tomasi's dissimilarity has a stronger positive effect with smaller truncation values and always give comparable or better results than the standard method. This strengthens the case for use of SAD over SSD as it is computationally cheaper. My study also shows that SAD's performance is very close to that of normalised additive or multiplicative correlation functions (Corr1 and Corr2), making SAD the optimum choice for a hardware implementation of a correlation algorithm. Scharstein and Szeliski studied the effect of the truncation value as well as Birchfield and Tomasi's dissmilarity measure: in my study, correlation algorithms were tested against different window sizes.

Scharstein and Szeliski also assessed dynamic programming algorithms, scanline optimisation and graph cut algorithms with and without Birchfield and Tomasi's dissimilarity measure. The influence of the truncation and dissimilarity measure was similar to that observed for the correlation algorithms. Some of the algorithms tested were not included in this study as the reported processing times are in the 10 to 30 minute range whereas this study focussed on candidates for real time implementation. Thus even though the graph cut approach is reported as the best solution, especially in the presence of occlusions, it is extremely slow and it was not considered likely to have an efficient hardware implementation [58].

***Faugeras** et al.*

Faugeras *et al.* [55] have analysed correlation algorithms for a Mars rover application. Their report describes several types of correlations implemented on several platforms with 4 to 6 DSPs each. They used Fua's method [67] and [68] to detect occlusions and claim to solve the problem completely: a trinocular algorithm can compute an error free depth map by combining the data from the three cameras.

They used algorithms with normalised additive and multiplicative correlations as well as zero mean variants. They describe an implementation which optimises the correlation window computation.

The choice of correlation area based algorithms was designed for a robust (over long periods of time) navigation solution for a Mars rover. Navigation tests were been made indoors and outdoors. Their experiments tested a whole system's performance with a stream of images and did not focus on the correspondence problem

[2]The truncation value depends on the image intensity range (0-255 for 8 bits for instance) as well as the way the matching cost is computed (SAD or SSD for instance).

alone, whereas in this work, I have focussed on objectively assessing the performance of matching algorithms using carefully selected pairs of test images.

Kanade and Okutomi's Adaptive Window Algorithm

Kanade and Okutomi's adaptive window algorithm [69] uses a statistical model to determine which the window size which produces an estimate of the disparity with the least uncertainty. It starts from an initial estimate of the disparity map and adapts the window size depending on local variations of intensity and disparity in an iterative algorithm.

To illustrate the algorithm - using the standard white square on a black background stereo pair - Figure 3.10, from Kanade [69], shows selected points and figure 3.11 shows the final windows used around those points.

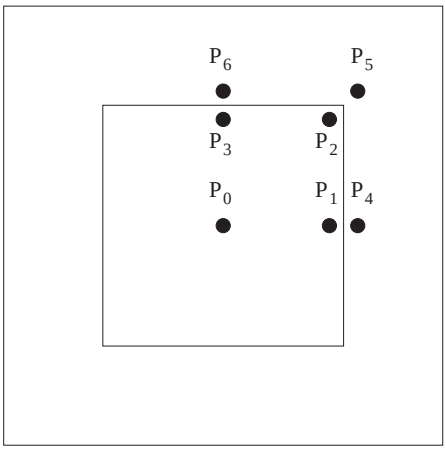


FIGURE 3.10: Selected points for the adaptive window algorithm description.

Table 3.1 compares the adaptive window algorithm with a fixed window algorithm for 3 different window sizes - using SSD[3] costs:

[3]Sum of Square Differences, see section 3.2.1 for a comparison with SAD

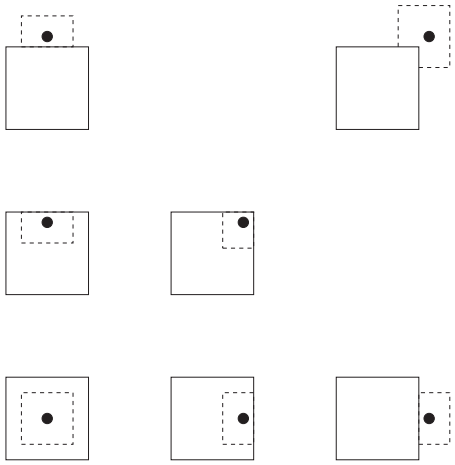


FIGURE 3.11: Dotted lines show the windows used by the adaptive window algorithm for the points given in figure 3.10

Window	Mean Error (pixels)
3×3	0.22
7×7	0.20
15×15	0.34
Adaptive Window	0.08

TABLE 3.1: Adaptive window algorithm versus SSD (Sum of Square Differences) with fixed window radii of 1, 3 and 7.

Lan and Mohr's Partial Correlation Algorithm

Lan and Mohr [2] introduced a variant of Zabih's approach - Census algorithm [33] - to deal with partial occlusions. The idea is that there exists an affine relation between the *local* left and right intensities except where there is an occlusion: their method tries to adjust the inlier and outlier weights inside the correlation window.

Lan and Mohr introduced partial occlusion because standard correlation methods usually take into account single populations, whereas in stereovision, matching windows contain pixels coming from different parts of the scene - as illustrated in figure 3.12. This phenomenon is called partial occlusion even though it may have other sources.

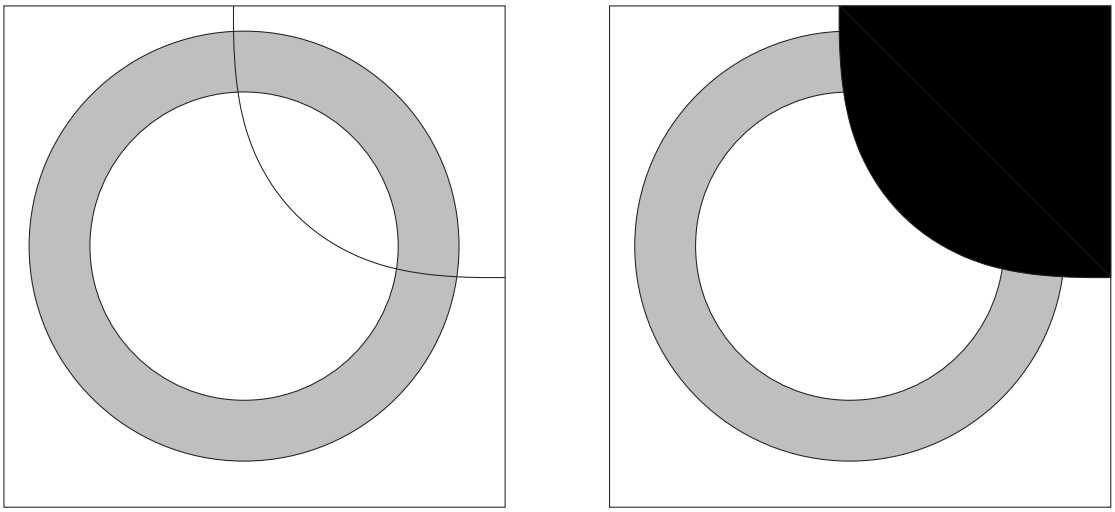


FIGURE 3.12: Partial occlusion phenomenon - as illustrated by Lan and Mohr [2]. The left and right window are seen from two different positions: some pixels in the right window come from another part of the scene. The circle on the left images illustrates the position of the occluding black circle on the right image.

They start from the hypothesis that the signal locally follows an affine relation - illustrated in figure 3.13:

$$I_R = k \cdot I_L + constant$$

They use robust statistics - based on the least median of squares regression - to find the outlier part of the correlation between I_R and I_L - see figure 3.13. The occluded part in each window is found and the correlation computed on the remainder of the window. This report compares four different algorithms: standard zero mean sum of squared differences (ZSSD), Zabih's rank transform [33], their

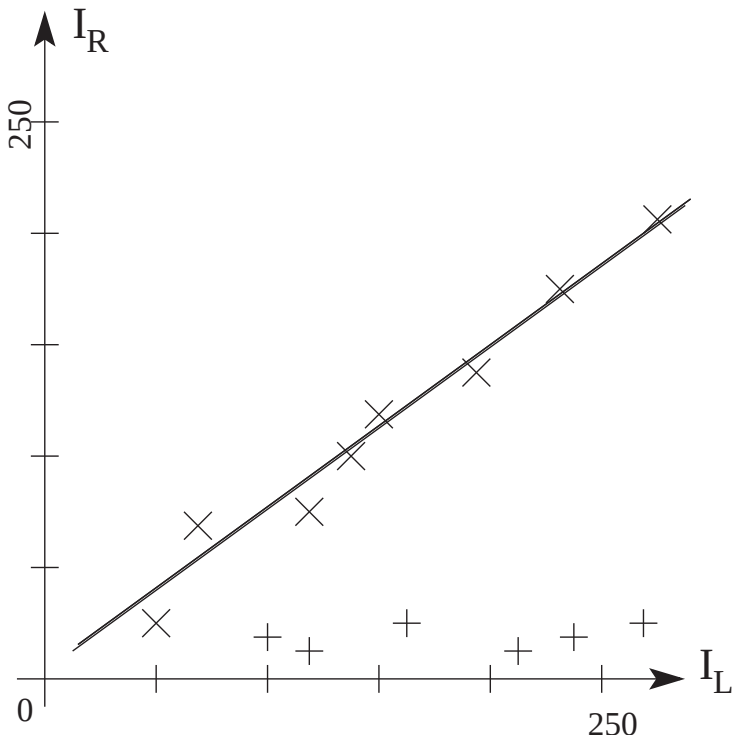


FIGURE 3.13: Affine relation: values following the affine relation between I_L and I_R are marked '$\times$': outliers are marked '+'

modification of Zabih's Census transform and their partial correlation algorithm, in most cases, this variant of the ZSSD algorithm outperforms the original approach and is slightly better than the rank or the Census transforms, but is slightly worse if points in non occluded regions are chosen.

Lan and Mohr's Census Variant

In the same report [2], Lan and Mohr discuss a variant of Zabih's Census algorithm [33]. In the inner part of the transform, instead of using 0s and 1s depending on $I(P) < I(P')$, they use the probability that $I(P) < I(P')$: for neighboring pixels P_1 and P_2; the bigger the intensity difference between P_1 and P_2 the more confidence there is in the intensity ordering. They model the probability that $I(P_1) < I(P_2)$ by a gaussian distribution $G(s, \sigma)$ where s is the observed value for $prob(I(P_1) < I(P_2))$ and σ is a measure of the standard deviation. Replacing the original Hamming distance by this method affects computation speed - they report a slow down factor of 100.

Lan and Mohr compared the standard ZSSD (Zero Mean Sum of Square Differences), their statistical approach called RZSSDC (Robust Centred ZSSD), the

original Rank transform and their modified Census transform. Table 3.2 gives their results for a full image and table 3.3 for a selection of occluded pixels in the image, showing the advantage of their method on these regions.

Algorithm	Disparity Error			
	$0 \sim 1$	$1 \sim 2$	$2 \sim 3$	$3 \sim \infty$
Original Rank transform	11946	2512	299	968
Lan and Mohr's Census transform	12960	2237	105	426
Lan and Mohr's RZSSDC	11845	1985	193	1702
ZSSD	13140	1941	116	528

TABLE 3.2: Original Rank transform, ZSSD and Lan and Mohr's Census and RZSSDC comparison on a full image.

Algorithm	Disparity Error			
	$0 \sim 1$	$1 \sim 2$	$2 \sim 3$	$3 \sim \infty$
Original Rank transform	77	14	0	49
Lan and Mohr's Census transform	89	12	0	39
Lan and Mohr's RZSSDC	118	10	0	12
ZSSD	56	0	0	84

TABLE 3.3: Original Rank transform, ZSSD and Lan and Mohr's Census and RZSSDC comparison on a selected occluded points subset.

Gimel'farb's Symmetric Dynamic Programming Stereo(SDPS)

Gimel'farb proposes a dynamic programming approach to the stereo problem based on a Markov chain modelling of possible transitions - see figure 3.14 - to form a *continuous* graph of profile variants (GPV) [70]. Then, the reconstructed profile has to maximise the likelihood ratio - chosen as the log-likelihood ratio - with respect to a purely random one [71].

Two matching points in the left and right images have for coordinates (x_L, y_L) and (x_R, y_R) respectively, where $y_L = y_R$ following the epipolar constraint. Nodes $N = (x, p, s)$ are defined by:

- $x = \frac{x_L + x_R}{2}$: the cyclopean coordinate,
- $p = x_L - x_R$: the disparity, and
- v: the visibility:
  - BVP (B) : binocularly visible points, *i.e.* seen by both left and right cameras,
  - MVP (M_L, or M_R) : monocularly visible, *i.e.* seen by the left or the right camera only.

The different transitions in the model only depend on their visibility states. Also, the Markov chain producing the purely random profile is defined by the two following transition probabilities [71]:

- $P(v_i = B | v_{i-1} = B)$, and
- $P(v_i = M | v_{i-1} = M)$, where M is either ML or MR.

because [71]:

$P_{ML|ML} = P_{MR|MR} \equiv P_{M|M}$ because ML and MR cases are equivalent by symmetry [71]
$P_{B|ML} = P_{B|MR} \equiv P_{B|M} = 1 - P_{M|M}$
$P_{ML|B} = P_{MR|B} \equiv PB|M = 0.5 \cdot (1 - P_{B|B})$

Finally, $P_{B|B}$ and $P_{M|M}$ are noted:

- for the purely random profile: $P_{B|B}$ and $P_{M|M}$ respectively, and
- for the profile reconstructed from a stereopair: $P^o_{B|B}$ and $P^o_{M|M}$ respectively.

For simplicity, only the case where $P^o_{B|B} + P^o_{M|M} = P_{B|B} + P_{M|M} = 1$ is considered [72], *i.e.* the algorithm is defined by: $SPDS(P_{B|B}, P^o_{B|B})$.

Note that the distribution of the purely random profile "controls" the smoothness of the reconstructed profile by the way of maximising the log-likelihood between the two profiles.

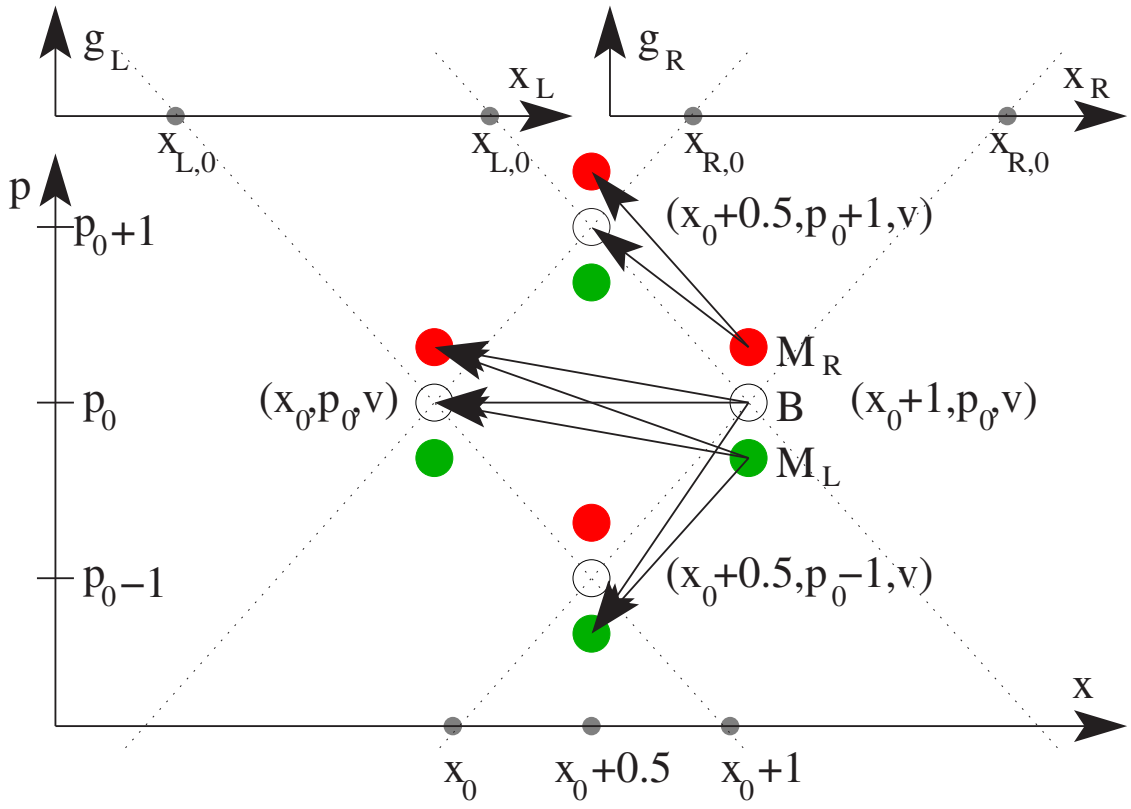


FIGURE 3.14: Graph of Profile Variants (GPV): possible transitions in the graph.

Figure 3.14 represents the possible transitions, where:

- g_L and g_R are the grey values for the two stereo channels, assumed to be epipolar,
- (x, p) represent the point in the profile variants, and
- v is the visibility.

Gimel'farb and Lipowezky provide an accuracy comparison between the SDPS algorithm and correlation based algorithms [72]. They use the four sets used by Scharstein and Szeliski as well as in this study: Map, Sawtooth, Tsukuba and Venus - images of a few hundred by a few hundred pixels in size with maximum disparity range of a few tens of pixels - as well as a digitised rectified, *i.e.* epipolar, aerial

stereo pair of an urban scene for which both dimensions are above a thousand pixels with a maximum disparity range of [50, 150] - containing 392 ground control points.

Image Set	% of pixels Abs. Err. <= 1		Mean of Abs. Err.		Standard Deviation		Error Max	
	SDPS	SAD	SDPS	SAD	SDPS	SAD	SDPS	SAD
Map	89.1	91.2	1.32	1.55	3.14	4.90	22	24
Sawtooth	93.7	94.6	0.60	0.41	1.36	1.46	11	15
Tsukuba	91.4	89.1	0.59	0.70	1.15	1.74	9	14
Venus	94.9	89.1	0.45	0.66	0.71	2.04	7	16

TABLE 3.4: Comparison of SDPS *vs.* SAD(r=4) on the four sets: Map, Sawtooth, Tsukuba and Venus.
Note that SAD results have been extracted from this study for the comparison.

Table 3.4 shows the comparison between the SDPS dynamic programming algorithm and SAD(r=4) on four common stereo pair between this study and Gimel'farb and Lipowezky's study. This shows a generally better behaviour for the SDPS algorithm especially in terms of outliers, with smaller mean of absolute errors, standard deviation and maximum errors, even if the initial percentage of pixels with errors of 0 or 1 pixel is about the same.

Algorithm	Mean of Abs. Err.	Std. Dev	Error Max
$SDPS(0.10, 0.90)$	4.68	7.92	55
$SDPS(0.25, 0.75)$	3.59	6.35	39
$SDPS(0.50, 0.50)$	3.02	4.94	36
Cross Correlation 7×7	5.51	11.28	50
Cross Correlation 21×21	3.31	8.16	49

TABLE 3.5: Performance of SDPS with 3 different sets of parameters and cross correlation with two window sizes.

Table 3.5 shows again a better overall behaviour of SDPS against a standard cross correlation method. In addition, Gimel'farb and Lipowezky observe that the

correlation algorithm tends to blur the buildings over a wider area than SDPS does.

This study found the same kind of relative behaviour between Pixel-to-Pixel - another dynamic algorithm - and the tested correlation algorithms. The reported results for SDPS lack processing times - preventing a more complete comparison.

Zabih** et al. **Graph Cut

Graph cut is an optimised technique for solving complex energy minimisation problems. Zabih *et al.* [34] describe the graph cut algorithms that also deal with the occlusion problem. The energy function is:

$$E(f) = E_{data}(f) + E_{occ}(f) + E_{smooth}(f)$$

where:

- E_{data} comes from pixel differences between the images,
- E_{occ} imposes a cost for making a pixel occluded,
- E_{smooth} tends to give pixels in the same neighbourhood similar disparities.

Versions of the graph cut algorithm using two approaches for the smoothness term: several older graph cut variants [73,74], Zitnick and Kanade's algorithm [75,76] and a standard correlation algorithm are all compared, see table 3.6. One of the two approaches is much more promising and analysed in greater detail as it outperforms almost all other tested algorithms.

Method	Errors %	Gross errors %	False neg. %	False pos. %
Zabih *et al.* [34]	6.7	1.9	45.6	1.1
Zabih *et al.*(swap algorithm) [34]	20.7	13.6	50.6	3.4
Boykov, Veksler and Zabih [73]	6.7	2.0	82.8	0.3
Zitnick and Kanade [76]	12.0	2.6	52.4	0.8
Correlation	28.5	12.8	87.3	6.1

TABLE 3.6: *Errors*: the algorithm did not compute the correct disparity,
Gross errors: for disparities within ± 1 of the correct disparity, or a pixel labelled as occluded,
False: show the error rate for occlusions.

In their study, Scharstein and Szeliski [43] report a few percent of bad matches on non occluded pixels using Zabih *et al.*'s algorithm on the Tsukuba, Sawtooth and Venus stereo sets, whereas other algorithms tend to have more than 5% of bad matches.

Rectangular Subregioning

Sun's algorithm [77] pre-processes the image to obtain rectangular windows. The regions are obtained in two steps, first horizontal stripes are merged until the disparity range in all neighbouring stripes differ by more than a threshold value, then the same process is repeated with vertical stripes.

This leads to smaller rectangular windows for which the disparity range is known and is smaller than the full range of disparities possible for the image. With the disparity range, each of these rectangles defines a volume for which the solution has to be found. A two stage dynamic programming approach is used to compute the best path using zero mean normalised correlation as a cost for the matching.

Scharstein and Szeliski's study does include, at the end, results for Sun's algorithm. Tests are performed on the four image sets: Tsukuba, Sawtooth, Venus and Map. Sun algorithm results are reported in table 3.7 as well as results for other algoritms discussed in this study as a comparison.

Algorithm	Tsukuba			Sawtooth		
	nonocc.	text.	disc.	nonocc.	text.	disc.
Sun	11.10 20	10.70 18	41.99 20	5.51 20	5.56 20	27.39 19
SSD (r=10)	5.23 15	3.80 10	24.66 17	2.21 11	0.72 10	13.97 15
Pixel-to-Pixel	5.12 14	7.06 17	14.62 10	2.31 13	1.79 12	14.93 17
Mühlmann	9.76 19	13.85 20	24.39 16	4.76 18	1.87 13	22.49 18
	Venus			Map		
	nonocc.	text.	disc.	nonocc.	N/A	disc.
Sun	4.36 15	4.78 12	41.13 19	4.17 19		27.88 19
SSD (r=10)	3.73 13	6.82 15	12.94 8	0.66 8		9.35 10
Pixel-to-Pixel	6.30 17	11.37 18	14.57 10	0.50 7		6.83 8
Mühlmann	6.48 18	10.36 17	31.29 18	8.42 20		12.68 16

TABLE 3.7: *nonocc.*: for bad pixels in non occluded regions,
text.: for bad pixels in textureless regions,
Note that the map image pair has no textureless regions - see 3.5.6.
disc.: for bad pixels in discontinuities,
The small number on the right gives the ranking in Scharstein and Szeliski's study.

3.2.5 Using Colour to Improve Matching

Colour and greyscale information are strongly correlated:

$$GreyLevel = \frac{R + G + B}{3}$$

which partially explains why most algorithms only use the greyscale information. Also it is faster to process one dimension than the three used for most colour spaces - RGB, normalised RGB or HSI for instance. Nevertheless colour presents interesting properties that should help matching: it is obvious than a red pixel should not match a blue one even if they have similar grey levels.

In this study, colour experiments - see section 8 - used two approaches:

- combined where the cost function is evaluated on each of the three colour coordinates and summed and

- separated where the cost function is evaluated on each colour coordinate and the component with the strongest match is used.

The two following sections describe work using colour with success:

- firstly, Koschan *et al.* using colour instead of greyscale values with a $\sim 25\%$ improvement,

- secondly, Koschan *et al.* pyramidal colour block matching, and

- lastly, Jordan and Bovik using colour to reinforce a result found on greyscale.

Klette and Koschan

Klette, Koschan, Schlüns and Rodehorst have reported a matching improvement by using colour information [78]. Koschan describes a block matching algorithm - practically implemented in a vision system described in [79] and [80] - using a chromatic/achromatic segmentation in the HSI (Hue, Saturation, Intensity) colour space [41, 81]. Results were $25 \sim 30\%$ better than using greyscale values only. In

this approach, achromatic pixels are defined as:

$$
\begin{aligned}
&\text{* Case 1:} && I > 95 \text{ or } I \leq 25 \\
&\text{* Case 2:} && 80 < I \leq 95 \text{ and } S < 18, \\
& && 60 < I \leq 80 \text{ and } S < 20, \\
& && 50 < I \leq 60 \text{ and } S < 30, \\
& && 40 < I \leq 50 \text{ and } S < 40, \\
& && 25 < I \leq 40 \text{ and } S < 60,
\end{aligned}
$$

where I is the intensity and S the saturation.

Since only the intensity and saturation are needed, the complex conversion from RGB to Hue is avoided. Blocks of size $n \times m$ - 8×8 in this example - are used; a block is defined as achromatic if 60% of the pixels inside the block are achromatic.

The similarity measure between blocks is the mean square error (MSE) defined as:

$$MSE_{colour}(x, y, \Delta) = \frac{1}{n \cdot m} \cdot \sum_{i=-k}^{k} \sum_{j=-k}^{k} D_x\big(F_R(x+i, y+j), F_L(x+i+\Delta, y+j)\big)$$

where x and y are the pixel coordinates, Δ is the disparity and D_x is:

$$D_C(f_1, f_2) = \sqrt{(R_1 - R_2)^2 + (G_1 - G_2)^2 + (B_1 - B_2)^2}$$

$$D_A(f_1, f_2) = |\frac{R_1 + G_1 + B_1}{3} - \frac{R_2 + G_2 + B_2}{3}|^2$$

for the chromatic and achromatic cases respectively.

Klette, Koschan, Schlüns and Rodehorst [78] describe the same kind of algorithm using the (I_1, I_2, I_3) space initially suggested by Ohta, Kanade and Sakai [82]. It is defined by:

$$
\begin{cases}
I_1 = \frac{R+G+B}{3} \\
I_2 = \frac{R-B}{2} \\
I_3 = \frac{2 \cdot G - R - B}{4}
\end{cases}
$$

This space - combined with the colour block matching approach - was reported to perform better than RGB, XYZ or HSI.

Note that the block matching algorithm has been implemented in parallel in [83].

Koschan Pyramidal Block Matching

Koschan *et al.* describe a pyramidal approach for a block matching algorithm [3, 4, 84]. The block matching algorithm described in the previous section is modified to use a quad pyramid:

- disparity at level $s+1$, $D(s+1)$ can be derived from the disparities at the previous level, $D(s)$, using the algorithm at level $s+1$,
- the disparity search space at level $s+1$ is derived from the disparity at level s using a tolerance factor D_T, thus defining D_Δ the width of the reduced search space $[D_{min}, D_{MAX}]$, *i.e.* controlling the smoothness of the disparity map.

i.e. :

$$D_\Delta(s) = 2^{s-1} \cdot D_T$$

$$D_{min}(s) = \begin{cases} D(0) - D_\Delta(s) & \text{for } s = 1 \\ D_{min}(s-1) - D_\Delta(s-1) & \text{for } s > 1 \end{cases}$$

$$D_{MAX}(s) = \begin{cases} D(0) + D_\Delta(s) & \text{for } s = 1 \\ D_{MAX}(s-1) + D_\Delta(s-1) & \text{for } s > 1 \end{cases}$$

Koschan *et al.* illustrate that without using the pyramidal approach, the block matching algorithm has difficulties matching correctly the two eyes of a face [3] ; *i.e.* the left eye is matched properly, but the right eye is matched with the left because of the repetitive pattern of the eye. This effect can also be seen in the given example of reconstruction at the end of chapter 4 in figure 4.13: half of the face is matched properly but the second eye is mismatched. Using this pyramidal approach Koschan *et al.* illustrate that because of the constrained range for the disparity both eyes are now matched correctly, they also claim that the depth of small structures like the ears is estimated more correctly.

Note that in the specific case of the human face another solution to avoid the mismatching eye artefact, it is also possible to put the cameras vertically instead of horizontally - Philippe Leclercq, Jiang Liu, Mark Chan, Alexander Woodward, Georgy Gimel'farb and Patrice Delmas, Comparative Study of Stereo Algorithms for 3D Face reconstruction, submitted to ACVIS'04 Brussels.

After describing the pyramidal approach as efficiently implementable in parallel - as well as the block matching algorithm alone, see previous section - Koschan *et al.* in [4] give the running times for their algorithm in table 3.8.

Image size in pixels	Block Matching 1 PU	Block Matching 10 PUs	Pyramidal Block Matching 1 PU	Pyramidal Block Matching 10 PUs
256x256	0.96 s	0.13 s	0.25 s	0.03 s
768x566	10.14 s	1.07 s	3.51 s	0.64 s

TABLE 3.8: Comparison of the running times for the block matching algorithm and the block matching algorithm using the pyramidal approach for 1 and 10 processing units (PUs). Results have been computed on a SGI Power Challenge with twelve 75MHz R8000 processors.

Table 3.8 illustrates the gain of parallel implementation for both the block matching algorithm alone or the pyramidal block matching algorithm.

Jordan and Bovik

Jordan and Bovik [85] describe an experiment evaluating the use of colour to help the matching of a standard algorithm - the Laplacian of Gaussian filtered pairs (a feature based algorithm). It was compared with the use of chromatic gradients to further describe the zero crossing. They chose this method rather than finding the zeros for each colour band - red, green and blue - as it is computationnaly cheaper and practically the position of the zero crossing in each colour band is not much different from the intensity ones because of the strong correlation between the colour bands and the intensity information.

The use of colour gradients is based on the presumption that close to an edge it is more likely that the colour will change. The colour space used was the normalised RGB space because the intensity information is already used by the standard intensity zero crossing algorithm and the normalised RGB colour space is more robust against intensity changes than the RGB colour space. The gradients used are:

$$\begin{cases} D_{rg}(x,y) = \frac{R(x,y)-G(x,y)}{R(x,y)+G(x,y)+B(x,y)} \\ D_{gb}(x,y) = \frac{G(x,y)-B(x,y)}{R(x,y)+G(x,y)+B(x,y)} \\ D_{br}(x,y) = \frac{B(x,y)-R(x,y)}{R(x,y)+G(x,y)+B(x,y)} \end{cases}$$

The sign of the variation is given by:

$$\frac{\partial}{\partial x}(g_\sigma * D_{rg}) = \big(\frac{\partial}{\partial x} g_\sigma\big) * D_{rg}$$

where g_σ is a two dimensional rotationally symmetric Gaussian. The orientation is given by:

$$\tan^{-1}\Big(\frac{\frac{\partial}{\partial y}(g_\sigma * D_{rg})}{\frac{\partial}{\partial x}(g_\sigma * D_{rg})}\Big)$$

The additional information derived from colour enables choice of candidates within the disparity range that have:

- the same intensity contrast sign,
- roughly the same orientation: $\pm 30 deg$,
- the same chromatic gradient sign for each of the normalised difference spectra.

So that after finding all intensity zero crossings, their algorithm can determine crossings with a unique solution and validate it.

For one of the test scenes, city, they observed:

- a $\sim 40\%$ decrease in crossings having at least one candidate, which shows that the chromatic information eliminates incorrect matches,
- a $\sim 60\%$ decrease in the total number of candidate matches, which shows that the chromatic information helps disambiguate potential candidates,
- a $\sim 70\%$ increase in unique matches,
- a $\sim 0.86\%$ increase in the correct unique matches - even though small, it is related to the strong increase of unique matches.

The two other sets, a Rubik's cube and a set of hand tools, show similar behaviours - although less strong for the cube and more significant for the tools set. A following article [86] describes in greater detail the use of chromatic characterisation for egde matching - feature based stereo. The algorithm described there uses a disparity gradient constraint described by Pollard, Mayhew and Frisby [87]. The RGB colour space is kept and is used colour band by colour band:

$$R_{\sigma_x}(x,y) = \frac{\partial}{\partial x}(g_\sigma * R(x,y)) = \big(\frac{\partial}{\partial x} g_\sigma\big) * R(x,y)$$

The orientation is given by:

$$\tan^{-1}\left(\frac{R_{\sigma_x}(x,y)}{R_{\sigma_y}(x,y)}\right)$$

Finally the chromatic gradient magnitude is:

$$\sqrt{R_{\sigma_x}(x,y)^2 + R_{\sigma_x}(x,y)^2}$$

Two uses of colour are examined, firstly at the end of the algorithm to remove ambiguity between several candidates and secondly as part of the primary cost function in the algorithm itself. The use of this colour gradient to characterise the possible candidates produced results similar to those of Klette, Koschan, Schlüns and Rodehorst discussed previously. The use of the colour gradients within the algorithm itself significantly improves the number of good matches especially for small window radii, but also decreases the processing time as fewer potential matches have to be processed.

Mühlmann** et al. **Efficient SAD Implementation

Mühlmann *et al.* divide the stereo algorithms in two [88, 89]:

- real-time reconstructing and
- improving accuracy regardless of processing time reconstructions.

Following Klette *et al.*'s claimed improvement of the signal to noise ratio using colour (see section 3.2.5), they used as a matching cost:

$$C(P) = \sum_{P' \in w(P,\beta)} |I_{L_{red}}(P') - I_{R_{red}}(P')| + |I_{L_{green}}(P') - I_{R_{green}}(P')| + |I_{L_{blue}}(P') - I_{L_{blue}}(P')|$$

Parabolas were fitted to costs *vs.* disparity curves and the minimum used to estimate disparities to sub pixel accuracy, see figure 3.15: only situations where one or two pixels are below a given threshold are considered, if a third smallest value is present the match is marked as bad. This was followed by median filtering - see Smith's sorting network [90].

The whole discussion of implementation takes into account hardware issues - such as the processor cache - to describe the impact of the order of nesting variables - length, width and disparity - in loops, as well as a discussion on the use of Single Instruction Multiple Data, *e.g.*MMX or SSE, architectures. On a dual processor 800MHz Pentium III system with 512MB RDRAM using one thread for calculations they report:

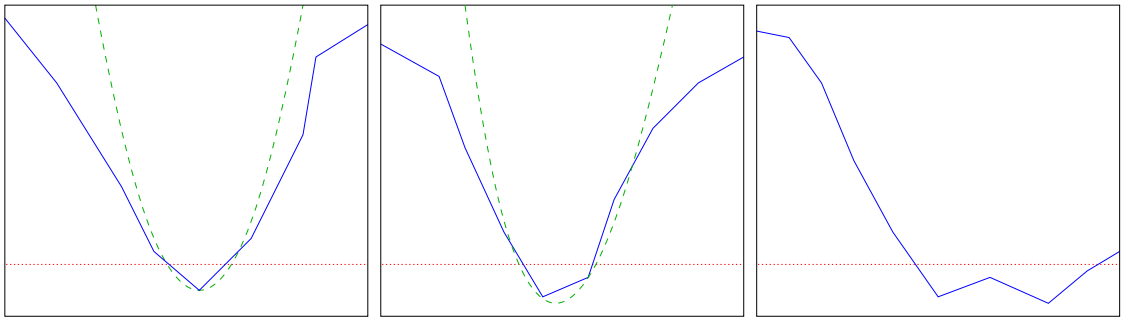


FIGURE 3.15: In the left and middle diagrams, one or two minima lie below the minimum threshold, so a parabola is fitted: its minimum provides the disparity of the current pixel. In the right diagram, there are too many values below the threshhold and the pixel is marked as bad and non matched.

image pair	image size, maximum disparity	ms	$\frac{ns}{pixel \times \delta_{max}}$
QSIF	$160 \times 120, 32$	50	82
Tsukuba	$384 \times 288, 20$	218	98
Pentagon	$512 \times 512, 21$	552	100
PAL	$720 \times 576, 50$	1890	91
Image0005[4]	$2048 \times 1536, 551$	124900	72

TABLE 3.9: Execution time for different size images and disparity ranges.

Tsukuba was also used in this study: its processing time is several seconds - for a window radius up to 4 - for a non optimised implementation of the SAD algorithm, thus the optimisations produced a speedup factor of about 20. Mühlmann *et al.* describe their algorithms as either a real time solution for stereo matching or as a fast first guess for more complex algorithms that need an approximately correct value as a starting point. Unfortunately they do not provide any correct match statistics.

3.2.6 Active Illumination

Introduction

Active illumination is the idea of projecting patterns onto the scene to help the algorithm to match the left and right images by increasing the local information in the images. It obviously helps in homogenous regions where no matching is otherwise possible.

The two following sections introduce:

- Kanade's *et al.* monochromatic active illumination for a four camera set working in real-time.
- Koschan's *et al.* using active *colour* illumination

For instance, recently Scharstein and Szeliski in [91], have used structured light to label each pixel. They use two different pattern generators: Gray codes and sine waves but which respectively use 80 and 100 patterns. Their results are reproduced in table 3.10 for two of their own sets: Cones and Teddy.

	Cones		Teddy	
Algorithm	t=1 (in%)	t=2 (in%)	t=1 (in%)	t=2 (in%)
SSD	17.8	9.3	26.5	12.8
Dynamic Programming	17.1	9.8	30.1	10.5
Graph Cut	12.6	7.0	29.3	11.4

TABLE 3.10: Percentages of pixels which disparity error is greater than the given threshold t.

***Kanade** et al.*

Kanade *et al.* describe a real time stereo system using four cameras as well as active illumination [92]. A projected light pattern of frequency modulated sinusoidally varying intensity helps the matching and to obtain dense reliable depth maps instead of interpolating between reliable pixels.

The system is composed of an 8x8 matrix of iWarp components - 20 MFlops[5] each - and is capable of extracting stereo depth data in real-time with an accuracy of less than 1mm in average for distances between 1.5 to 3.5m away from the cameras.

[5]Millions of Floating point Operations per Second

The error analysis results is made by locating planar patches in the image as well as a cylinder of known cross sectional radius. The furthest plane is 1.7m away from the camera and the cylinder 3.3m. Average error is always less than 1mm with a maximum error of a few mm and a standard deviation less than half a mm, see table 3.11

Element	Patch size in pixels	Avg error (μm)	Max error (mm)	Std Dev (μm)
Plane 1	20925	550	2.24	400
Plane 2	12405	420	1.91	310
Plane 3	993	520	2.97	420
Plane 4	1340	370	1.75	320
Cyl. 1	25200	640	4.35	540
Cyl. 2	35150	640	3.17	500

TABLE 3.11: Error fit for the planar patches and cylinder.

In conclusion, Kanade *et al.* mention that instead of a regular modulation of the active light a random one would be preferable to eliminate the last chances of repetitive pattern. Also some problems with dark surfaces or surfaces almost parallel to the projection: the use of multiple projectors/patterns, either time or colour sequenced may reduce the last causes of errors.

Chapter 10 of this study uses a colour pattern consisting of squares of different colours projected onto the measured object.

***Koschan** et al.*

Following Kanade *et al.* conclusion on the better use for a colour pattern instead of sinusoidal varying intensity pattern, Koschan presents the use of active *colour* illumination in [3,4] combined with a pyramidal block matching approach. Koschan *et al.* use a rainbow like pattern to illuminate the scene, its spectrum S is defined in the 3 RGB bands as:

$$\begin{cases} S_R = \sin\left(\frac{i}{n} \cdot \pi\right) \cdot \left(\frac{G_{MAX}}{2} - 1\right) + \frac{G_{MAX}}{2} \\ S_G = \sin\left(\left(\frac{2}{3} + \frac{i}{n}\right) \cdot \pi\right) \cdot \left(\frac{G_{MAX}}{2} - 1\right) + \frac{G_{MAX}}{2} \\ S_B = \sin\left(\left(\frac{4}{3} + \frac{i}{n}\right) \cdot \pi\right) \cdot \left(\frac{G_{MAX}}{2} - 1\right) + \frac{G_{MAX}}{2} \end{cases}$$

where $i = 0, ..., n$ is the row index of the generated colour spectrum S_{RGB} and G_{MAX} is the maximum intensity in every colour channel. An illustration of the spectrum is given in figure 3.16.

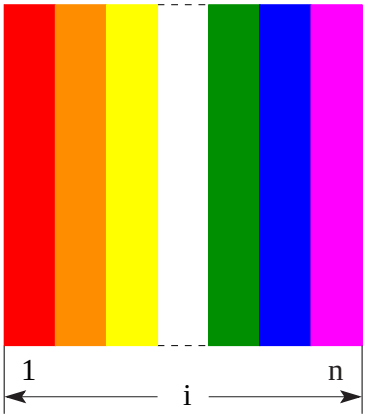


FIGURE 3.16: Sorting Network to compute the median of 9 elements ($P1$ to $P9$). Each cell has as output the minimum (bottom left) and the maximum (bottom right) of the two numbers on its inputs (at the top).

Results are given for a synthetic pair of a cube with uniform shading and no texture thus presenting very ambiguous matching, in this case the projected colour rainbow really improves the matching - see table 3.12 - as it is the only source of information for the algorithm to perform matching.

Koschan *et al.* use the cube again but also make additional measurements on a synthetic image pair representing the bust of Beethoven [4]. The results compare the

Difference in pixels	Good Matches Percentage	
	without active illumination	active colour illumination
0	7.9	62.2
1	12.2	25.8
≥ 2	79.8	12

TABLE 3.12: Koschan results - from [3] - using pyramidal block matching with and without the use of active colour illumination.

algorithm performance with and without active colour illumination in the following cases:

- On the original image set, see table 3.13.
- On the set corrupted with Gaussian noise ($\sigma = 10.0$), see table 3.14.
- On the set corrupted with different contrast, see table 3.15. The right image intensity I_R was transformed into I'_R before projecting the colour pattern using:

$$I'_R = \frac{3}{4} \cdot I_R + \frac{1}{8} \cdot G_{MAX}$$

  where G_{MAX} is the maximum intensity.

- On the set corrupted by intensity difference, see table 3.16. I_R was transformed into I'_R before projection of the colour pattern using:

$$I'_R = \left(\frac{I_R}{G_{MAX}}\right)^{\frac{1}{\chi}} \cdot G_{MAX}$$

  where G_{MAX} is the maximum intensity and $\chi = 1.5$

Tables 3.13, 3.14, 3.15 and 3.16 show that the use of active colour illumination increases the quality of matching in all cases of image corruption as well as for the original images.

Difference in pixels	Using intensity only in %	Using active colour illumination in %
0	59.6	63.3
1	34.0	31.8
≥ 2	6.3	4.8

TABLE 3.13: Koschan results - from [4] - using pyramidal block matching with and without the use of active colour illumination on original set.

Difference in pixels	Using intensity only in %	Using active colour illumination in %
0	39.8	44.3
1	42.5	43.5
≥ 2	17.7	12.2

TABLE 3.14: Koschan results - from [4] - using pyramidal block matching with and without the use of active colour illumination when the right image is corrupted by Gaussian noise ($\sigma = 10.0$).

Difference in pixels	Using intensity only in %	Using active colour illumination in %
0	14.7	37.7
1	20.9	42.6
≥ 2	64.4	19.7

TABLE 3.15: Koschan results - from [4] - using pyramidal block matching with and without the use of active colour illumination with different contrast.

Difference in pixels	Using intensity only in %	Using active colour illumination in %
0	9.6	28.6
1	18.4	36.7
≥ 2	72.0	34.7

TABLE 3.16: Koschan results - from [4] - using pyramidal block matching with and without the use of active colour illumination with different intensity.

3.3 Software

A modular software has been designed in ANSI C to run all the experiments, see figure 3.17. The mainly used image format is ppm/pgm, an array of floats with a comment header is also used to keep the precision of the results, graphic formats are used for reading the input stereopair and for display only. All image sets and algorithms are described by a configuration file - see also appendix A - also describing file names and directories. Once a stereopair is loaded, it is possible to change its parameters (convert to greyscale for instance) before choosing the algorithm to process it with. If a ground truth disparity map exists it is possible to compare the processed disparity map with it and create histograms of the errors. Functions have been written to format the file name depending on the algorithm and chosen parameters as well as file path.

Note that most of the functions have been documented in the source files using ROBODoc - version 4.0.6 [93]. As an example, the given 'GNUmakefile' compiles the software as well as executes ROBODoc if it is installed.

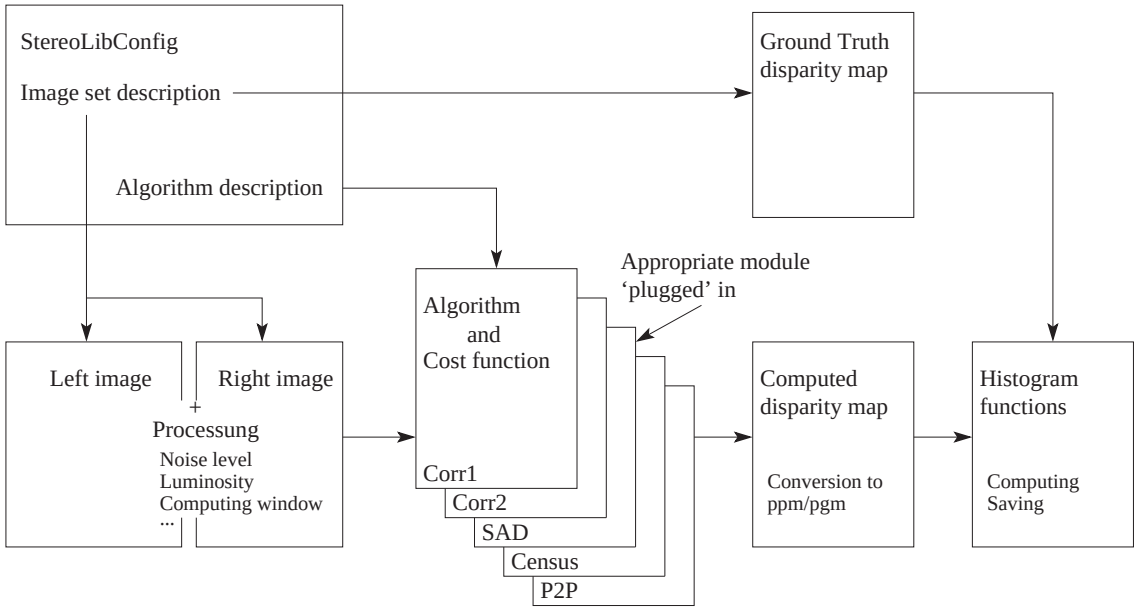


FIGURE 3.17: StereoLib's organigram

3.4 Metrics

In order to compare computed disparity maps objectively, one needs to *measure* the accuracy as distinct from subjectively estimating it by simply viewing images. A

typical distribution showing percentage of pixels exhibiting a given disparity error is shown in figure 3.18. Note that no sub-pixel estimation techniques were employed. An appropriate disparity error clearly depends on the actual application in which the dense disparity map will be used: in this work, the percentage of 'good' matches has been reported as those with a disparity error less than 0.5 pixels. Less stringent criteria, such as that used by Scharstein and Szeliski [43], who include all disparities within 1.5 pixels of the correct match as 'good', may well be acceptable in many applications. Lan and Mohr [2] define several classes:

- good match: up to one pixel error,
- near miss: between one and two pixels error,
- bad matches: between two and three pixels error and finally
- false matches: error bigger than three pixels.

My stringent criterion produces numbers of good matches which may seem low when compared with other work. This work is concerned with assessing relative algorithm performance so that the choice of acceptable error is somewhat arbitrary as long as a consistent criterion is applied. The fraction of good matches is clearly the most important metric but the mean and the standard deviation highlight characteristics of the algorithms or the way they are implemented. For example, a small peak to one side of the main peak (*cf.* figure 3.18) is due to aliasing. Such peaks appear often: when there is more than one match, the the direction in which the program scans the possible disparities causes the first one to be chosen resulting in a bias in the calculated disparity. The Census algorithm is particularly susceptible to such artefacts because the cost function is a sum of small integer values (Hamming distances).

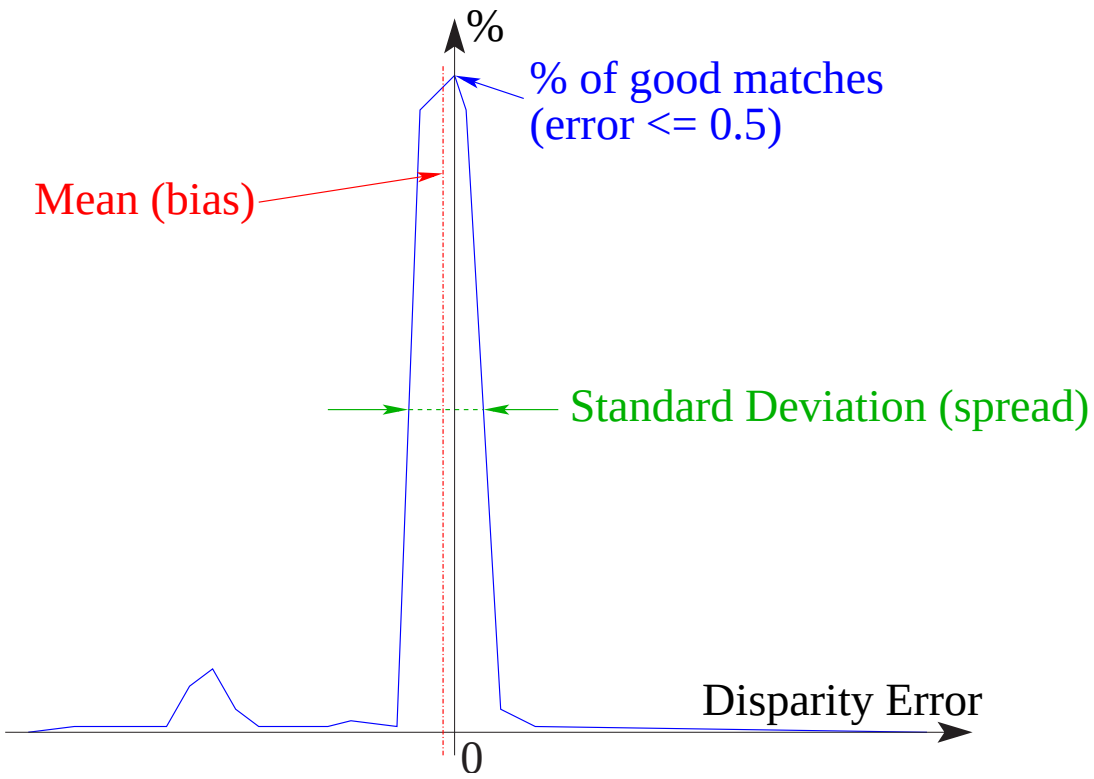


FIGURE 3.18: Metrics: typical distribution of percentages of pixels having a specified disparity error.

3.5 StereoSets

3.5.1 Introduction

This section presents the stereosets used in this study. It describes the characteristics of every stereo pair including its working window to avoid border effects, as well as any other relevant information (*i.e.* added noise, different baselines, ...).

3.5.2 Working window

To fairly assess all the stereo matching algorithms, any border effects were eliminated by leaving a border of 20 pixels around each image to allow experiments with very large window radii.

If the epipolar constraint is met for all images (whether by careful alignment or rectification), matching only occurs on the same scan line. As described in 2.3, the disparity range (Δ_{min} - Δ_{max}) can include both positive and negative values.

Note that in the case of parallel axis, the lower limit is forced: $\Delta_{min} = 0$.

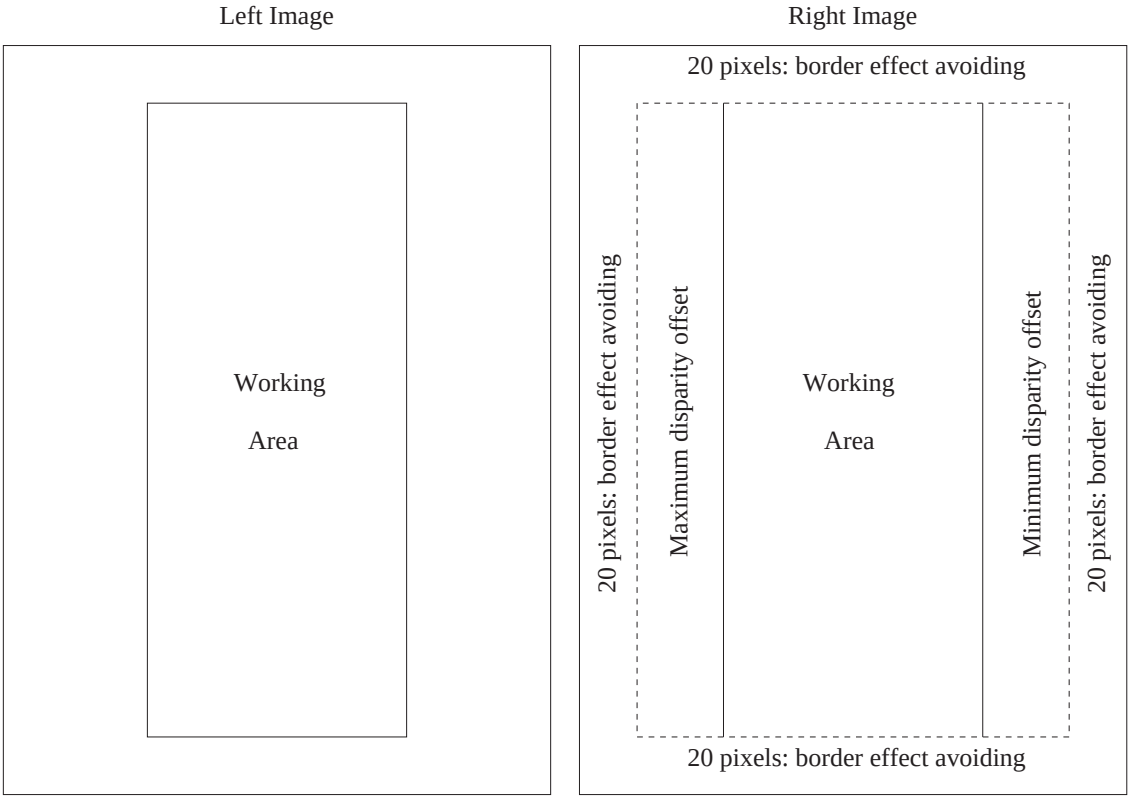


FIGURE 3.19: Sketch showing how the working window is set to avoid border effects and take the disparity range into account.

For each image set used, the working window is outlined in the left and right images in unbroken lines: dashed lines in the right window delineate both maximum and minimum disparity limits. Refer to the code in *StereoSetsSize.c* which has been used for both computing the working windows and to create the graphical representation of these borders.

3.5.3 MRTStereo

The *MRTStereo* program [94] is an addon to a ray tracing software package (*MRT* = Modular Rendering Tools) that enables to create stereo sets from scene descriptions. I used two sets computed with MRTStereo: the "Corridor" set and the "Madroom" set. The main benefits of using MRTStereo are:

- the ability to specify the baseline between the two cameras for the study on varying the baseline,
- ray tracing produces images with no noise from the camera sensor or other sources,
- the ability to choose the image resolution - which impacts the accuracy as shown in study of the "CoinStack" set and the "Shell" set.

Gerdes provides more information on his web site [94].

3.5.4 Corridor

Default Set

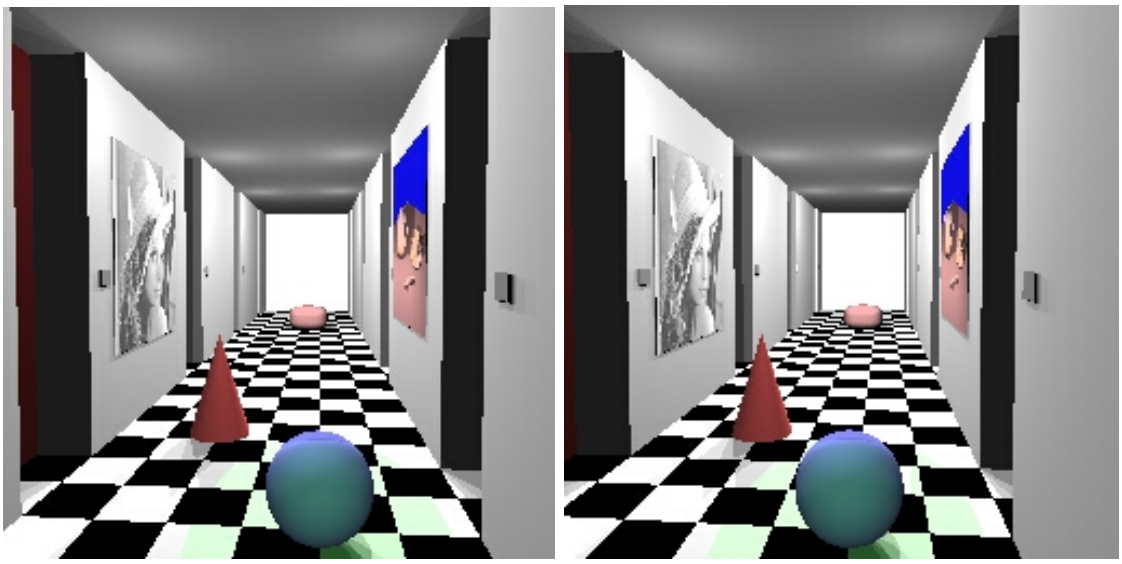
The *Corridor* set was computed using the MRTStereo tool. The image size is 256×256 and default baseline used produced a maximum disparity of 20 pixels. This set has very little colour (most pixels are 'grey' *i.e.* $R = G = B$) apart from some objects and textures on the wall. The main difficulty in this set is the complete absence of texture on the back wall. For further reference, the 3D corridor scene description files are given in appendix B.

Different Baselines Sets

For the study of the impact of the baseline on stereo algorithms, the Corridor set was computed with increasing values for the baseline, from 10 to 90. Figure 3.21 shows on top a reference left image followed by right images with increasing baselines (10, 50 and 90) and the corresponding left and right occlusion maps. Increasing the baseline can improve the accuracy (see chapter 9 for more details) but also increases the number and size of occlusions which is a central problem in stereo algorithms.

Additive White Gaussian Noise Sets

The Corridor set was produced by ray tracing and is free of the noise present in real images. This set was corrupted with increasing levels of additive white gaussian noise to assess the robustness of matching algorithms to noise. A full description of the noise corruption process used is given in appendix D.



Left and right images for the "Corridor" set, baseline of 20 and no added noise.

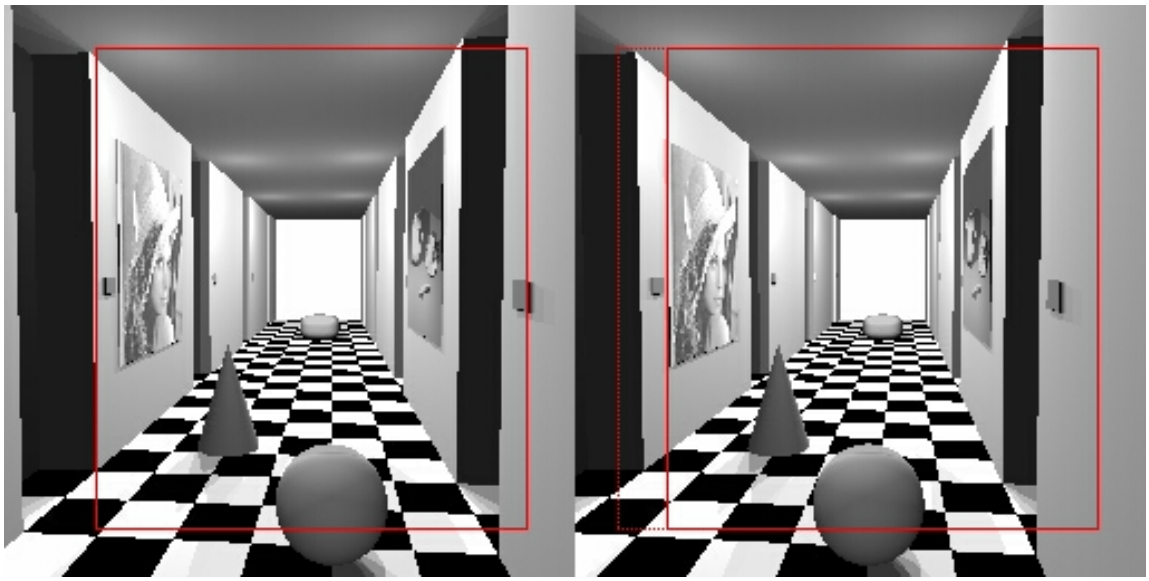
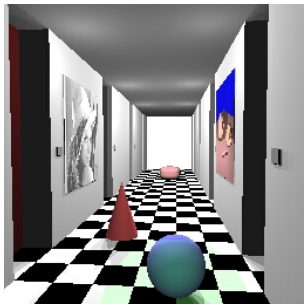
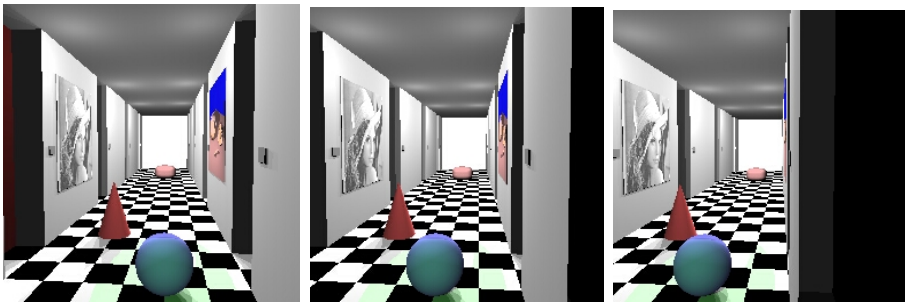


FIGURE 3.20: Corridor: working window
Image size: height 256, width 256,
minimum disparity 2.5 pixels, maximum disparity 21.7 $\sim$ 22 pixels,
Image window: (c:41, l:19) - [c:194, l:216] : 41904 pixels

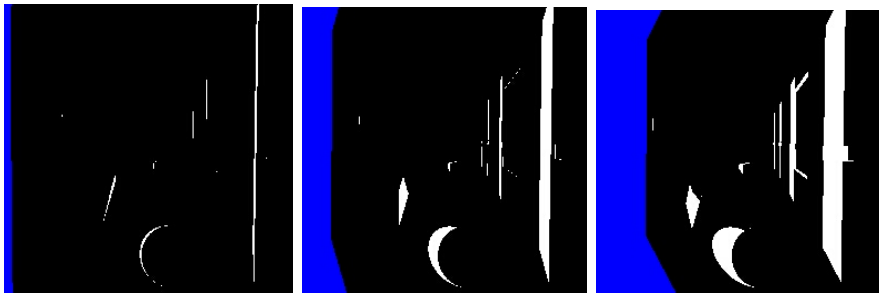


Reference left image, baseline=0

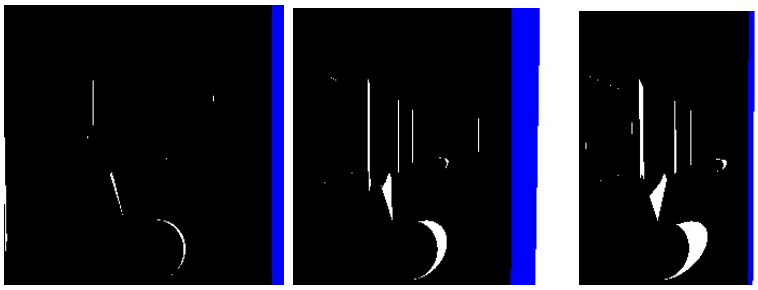
baseline=10 baseline=50 baseline=90



Right images



Left occlusion maps



Right occlusion maps

FIGURE 3.21: "Corridor" set with differing baselines

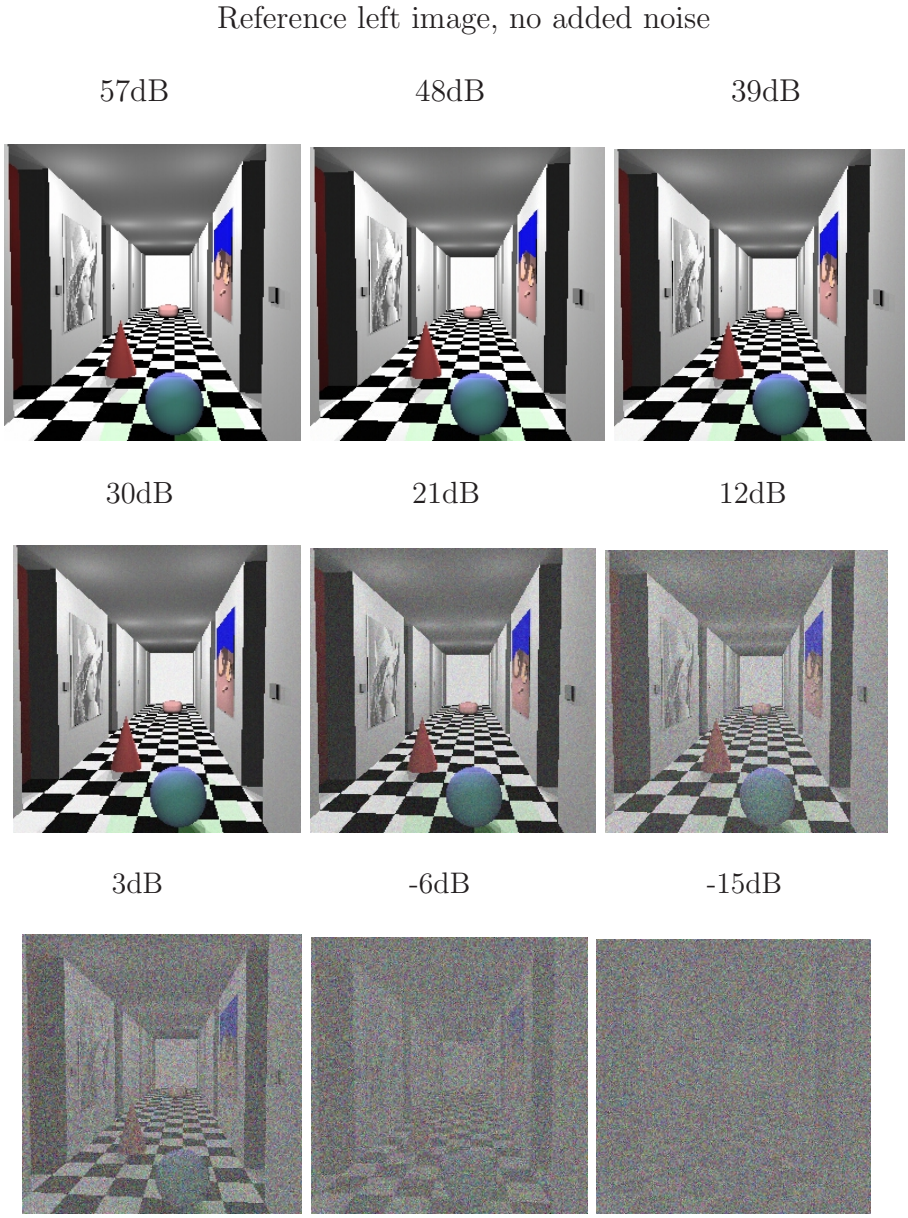
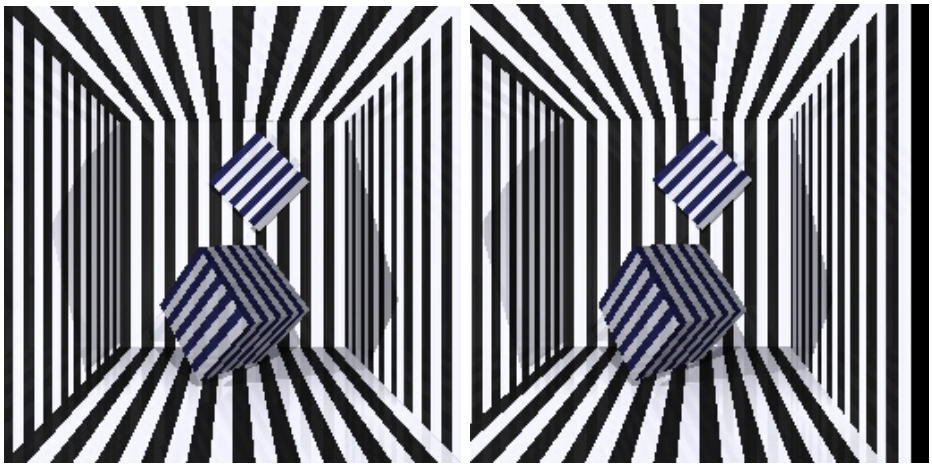


FIGURE 3.22: "Corridor" set with different levels of noise added
See appendix D for a precise description of the significance of x dB of noise.

3.5.5 Madroom

The "Madroom" set is also created with *MRTStereo*. This set has been kept because it represents a real challenge for any stereo algorithm as it presents lots of constant width stripes that make it hard for any algorithm to find accurately. The results on this set are usually complete outliers.



Default left and right images for the "Madroom" set.

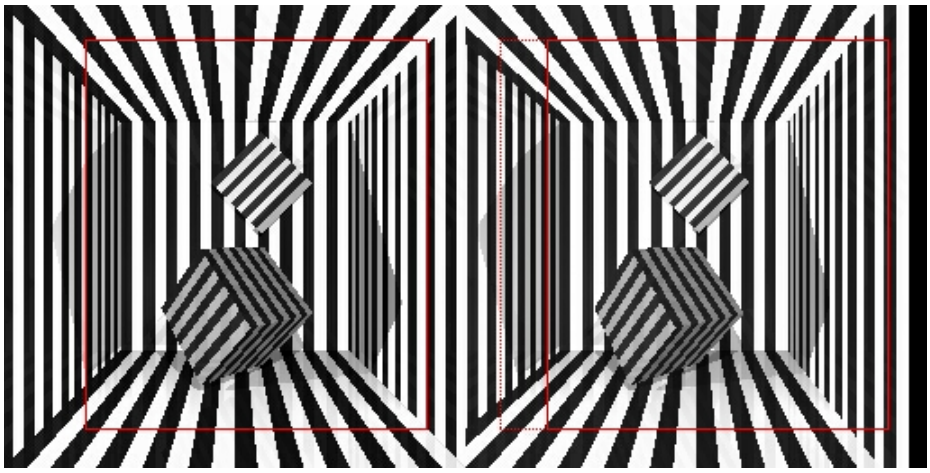
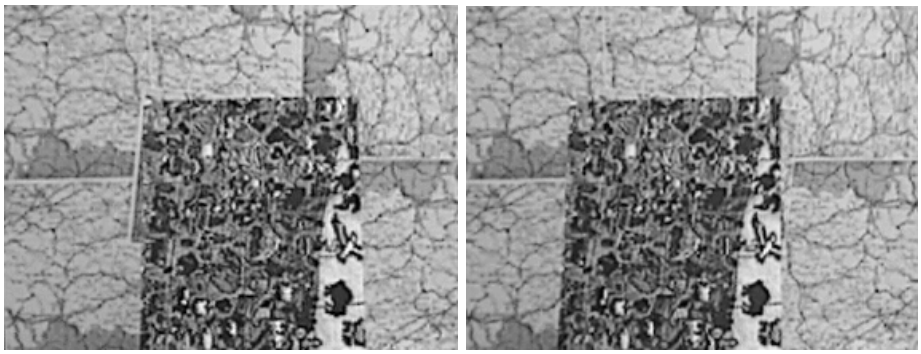


FIGURE 3.23: Madroom:
Image size: height 256, width 256,
minimum disparity 12.668714, maximum disparity 25.500000 ($\sim$ 26),
Image window: (c:45, l:19) - [c:190, l:216] : 41040 pixels

3.5.6 Map

The "Map" stereo set usually gives very good results, it is a greyscale pair of a map on top of other maps. It presents fine, non repetitive texture and very small occlusions, so that all algorithm always perform very well on it.



Default left and right images for the "Map" set.

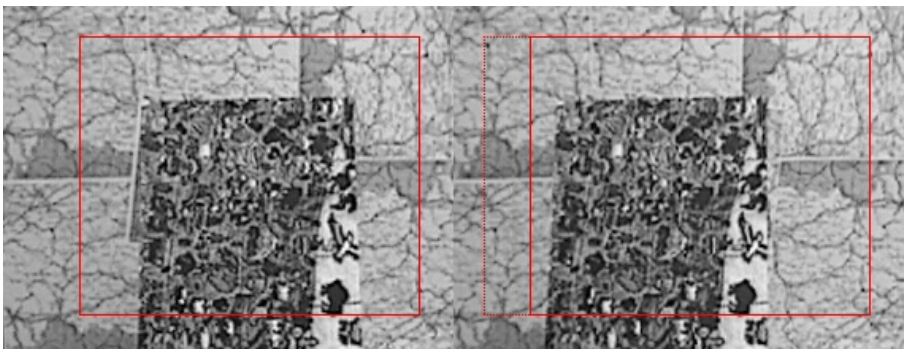


FIGURE 3.24: Map:
Image size: height 216, width 284,
minimum disparity 4.375000, maximum disparity 28.125000 ($\sim$ 29)
Image window: (c:48, l:19) - [c:215, l:176] : 37840 pixels

3.5.7 Sawtooth

The "Sawtooth" set provides both greyscale and colour textures with sloping walls.

Default left and right images for the "Sawtooth" set.

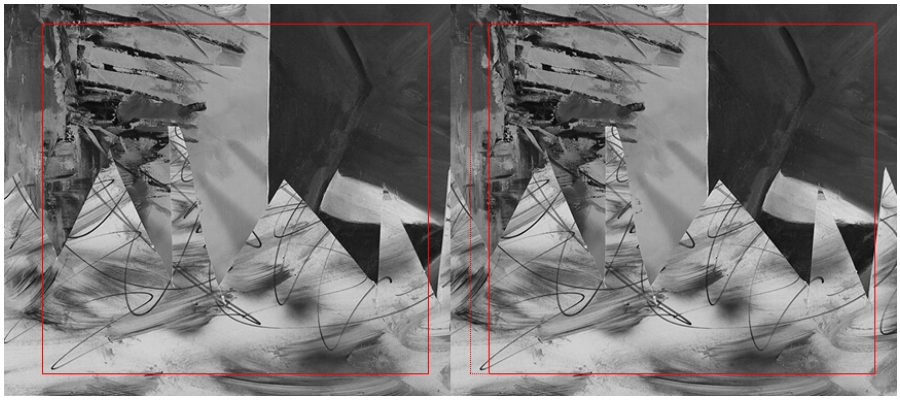
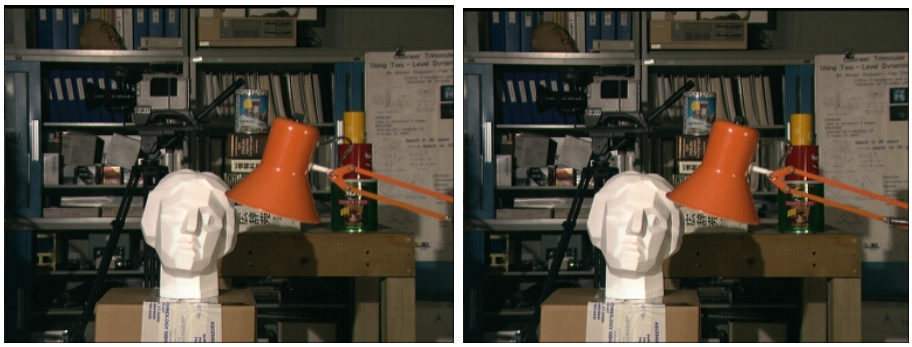


FIGURE 3.25: Sawtooth:
Image size: height 380, width 434
minimum disparity 3.875000, maximum disparity 17.875000 ($\sim$ 18)
Image window: (c:37, l:19)- [c:376, l:340] : 127840 pixels

3.5.8 Tsukuba

The "Tsukuba" set is a real scene with several distinct layers. Usually algorithm cannot find accurately the lamp's arms as either the window blends the disparity gap either cost function try to provide "smooth" disparities.



Default left and right images for the "Tsukuba" set.

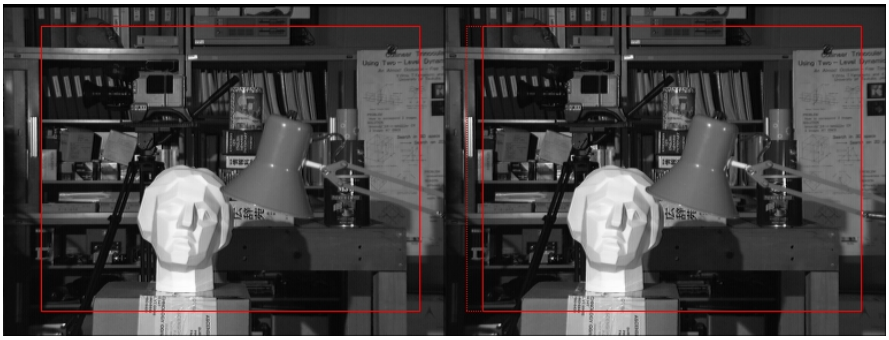


FIGURE 3.26: Tsukuba:
Image size: height 288, width 384,
minimum disparity 0.000000, maximum disparity 14.000000 ($\sim$ 14)
Image window: (c:33, l:19) - [c:330, l:248] : 81840 pixels

3.5.9 Venus

The "Venus" set is a superposition of sloping planes with colour textures, either writing or photo or painting.

Default left and right images for the "Corridor" set.

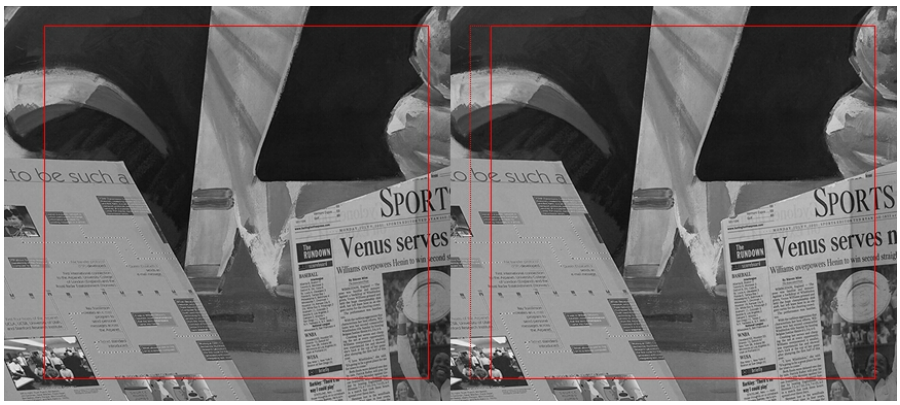


FIGURE 3.27: Venus:
Image size: height 383, width 434,
minimum disparity 3.000000, maximum disparity 19.750000 (~ 20)
Image window: (c:39, l:19) - [c:374, l:343] : 128282 pixels

3.5.10 CoinStack

The idea of a stack of coins as a stereo pair has been used for different camera calibration purpose. Coins are usually of precise known dimensions, in our case, we used several coins from New Zealand, their complete details can be found at the Reserve Bank of New Zealand [95].

The used background is a copy of a 20 New Zealand dollars note. The idea is that notes have very fine texture for forgery prevention purpose. This fine texture should be easily matched by algorithms and therefore provide a good reference disparity for the image background.

The copy of a ruler has also been made either on the copy of the note itself or on a transparent strip and can be used to measure the field of view of the camera. Using modern photocopiers the scale on the copied paper or transparent is exactly the same size than the original ruler.

Exemples of a stack of coins.

3.5.11 IGN Aerial epipolar stereo pair with ground truth

This set has been given by the French national geographic institute (Laboratoire MATIS, Institut Géographique National, IGN). The two images in figure 3.28 have been rectified to satisfy the epipolar constraint. Two disparity maps - see figure 3.29 - were provided: one generated by laser and the other one by manual entry, with an accuracy of one tenth of a pixel.

From these large images (3526×4800) a smaller subset was extracted:

- the given disparity map do not cover the full initial set, and
- a smaller subset is faster to process.

Figure 3.30 illustrates the chosen part of the main stereo pair as well as the windows used for processing and figure 3.31 the disparity map for the chosen section.

After a careful check these two images look very similar:

- only two cars have moved between the two images in the lower right corner of the photos, and
- a some pedestrians around the church.

These images have been taken with verging camera axes and have both positive and negative disparites - see section 2.3. Also the existence of a laser generated disparity map makes it possible to compares its results *vs.* binocular stereo algorithms - see chapter 11.

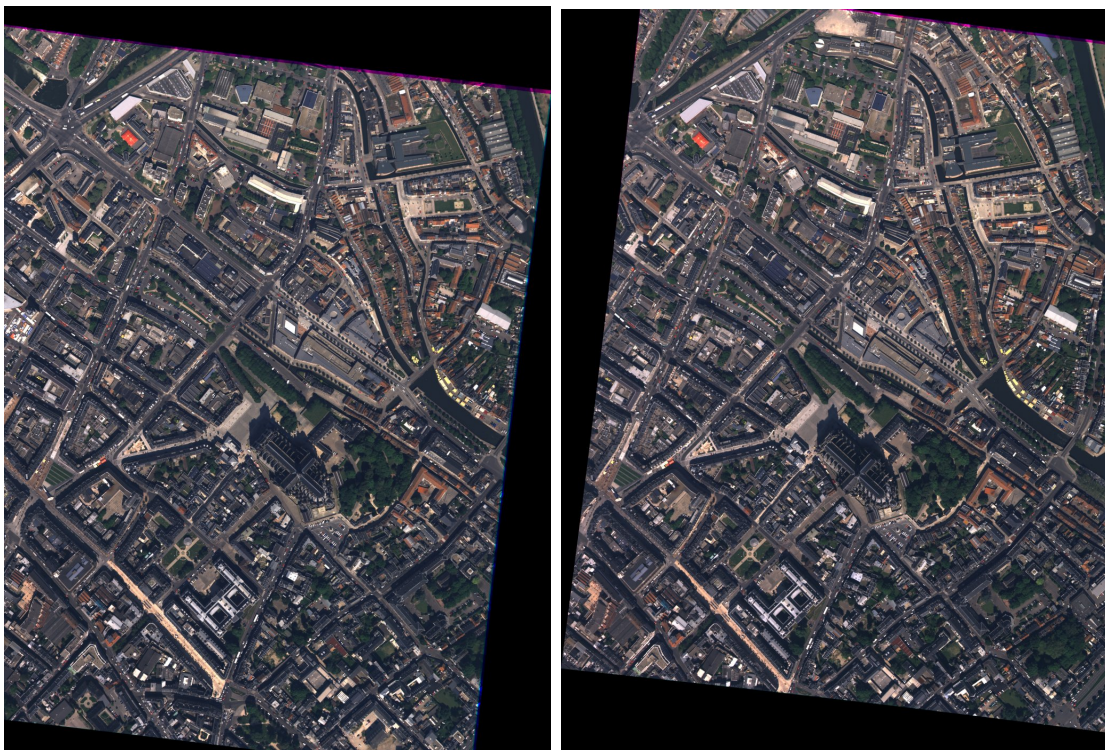


FIGURE 3.28: IGN main set:Image size: height 3526, width 4800

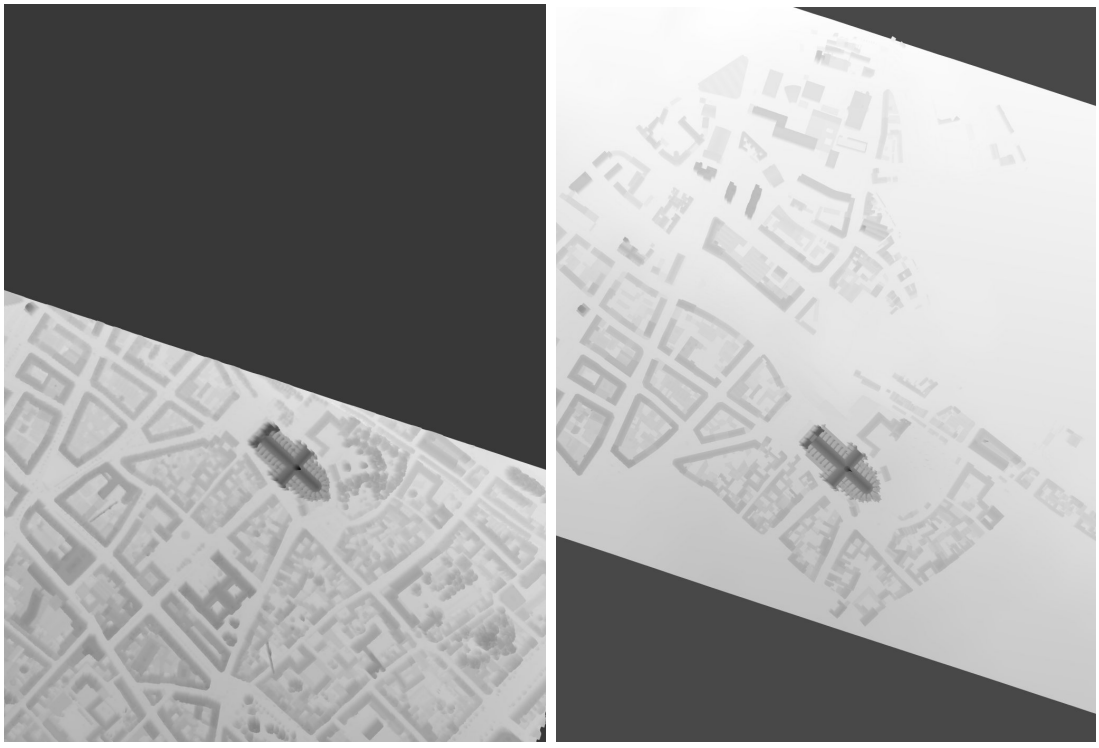


FIGURE 3.29: IGN main set:
Left: laser range scanner disparity map - Right: manual input disparity map.

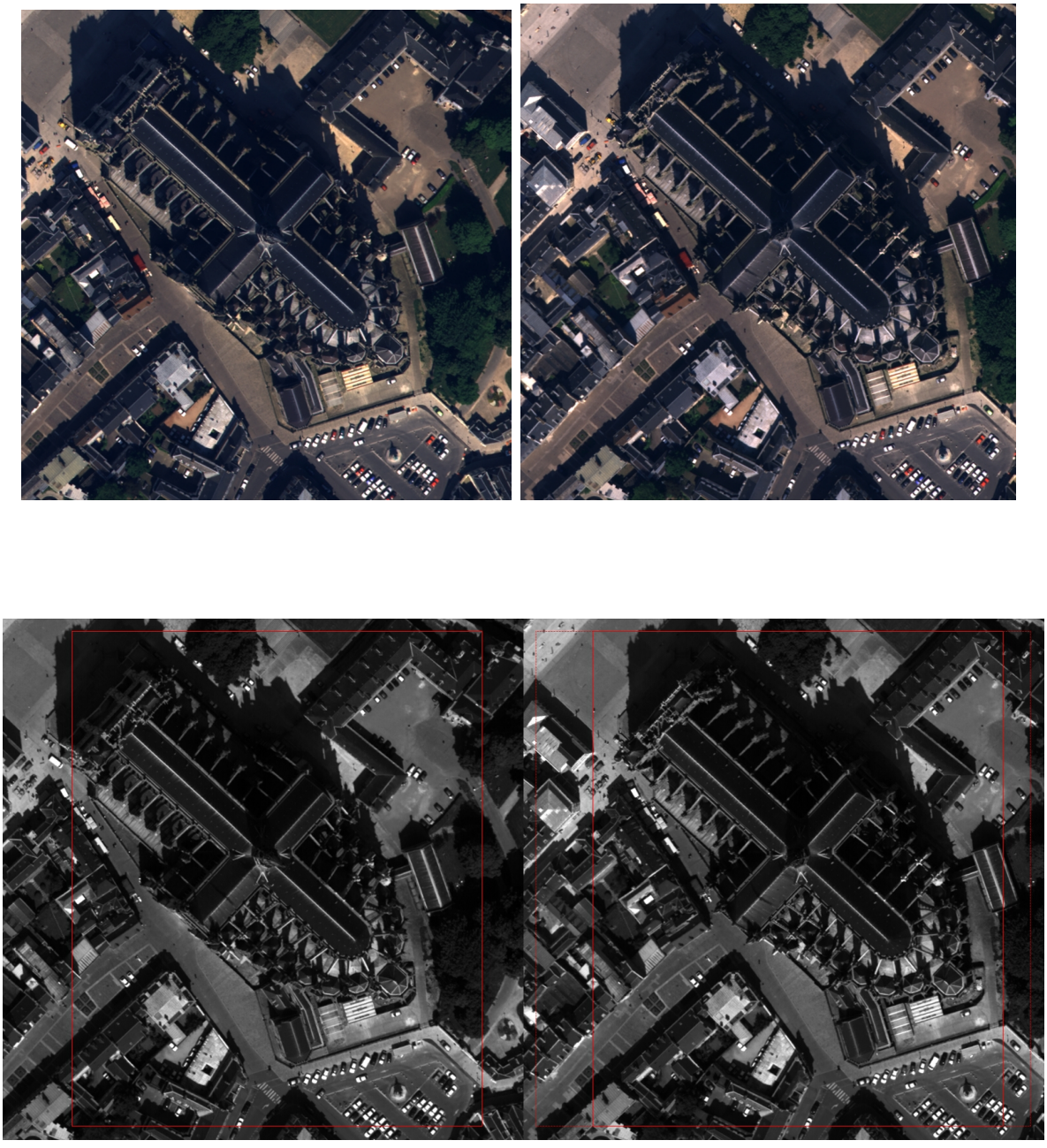


FIGURE 3.30: Amiens _ 01: working window
Image size: height 800, width 800,
minimum disparity -42 pixels, maximum disparity 87 pixels,
Image window: (c:106, l:19) - [c:361, l:760] : 274360 pixels

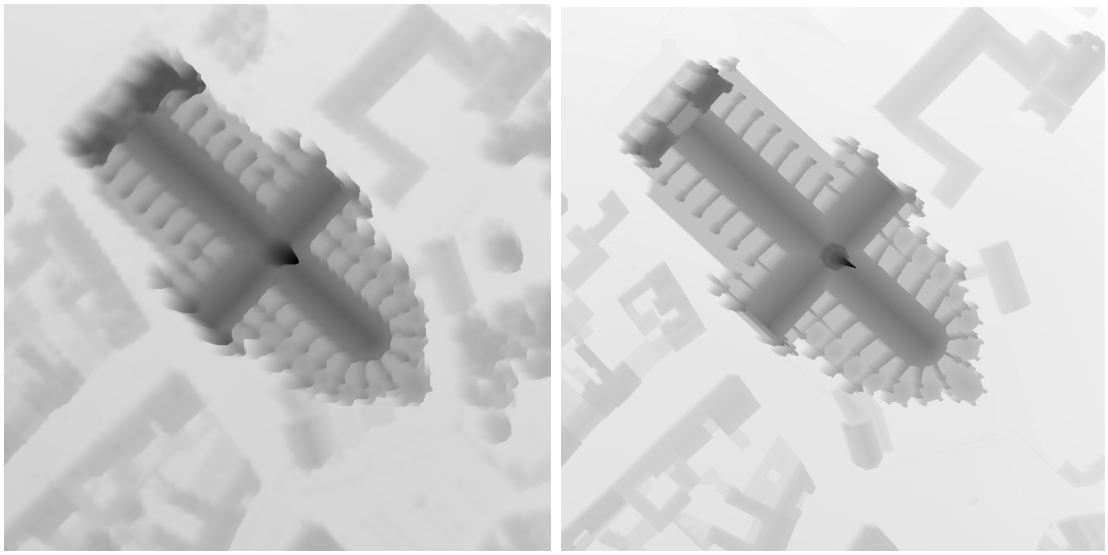


FIGURE 3.31: Amiens _ 01 set:
Left: laser range scanner disparity map - Right: manual input disparity map.

Chapter 4

Camera Description and Setup

4.1 Introduction

This chapter describes the set up of an experiment: it describes how to

- practically describe a given camera,
- choose the camera parameters,
- set the camera and establish the scene configuration according to these parameters and
- reconstruct a scene from experimental results.

Firstly, a camera parameter has to be determined: the ratio D_1/p. If enough information is available, it is possible to estimate the pixel size p and use it to compute D_1. However, a procedure is described, which obtains an experimental value for D_1/p, which appears in all formulae and is enough to describe the camera for a stereo application.

Note that in general neither D_1 nor p can be separately determined experimentally These two variables can only be directly determined if:

- D_1 is known because
  - the camera has a fixed documented focal length or
  - the range of the adjustable zoom is known and forced to the maximum or minimum value - which is rather suited for high-end cameras which provide accurate feed back on the zoom setting - or
- p the camera manufacturer clearly states either which chip is used or the chip's pixel size, but for most cameras this information is not readily available.

Secondly, the user has to establish a target accuracy for measurements on the object. Using the camera description, it is possible to check whether the target accuracy can be achieved or not. If achievable, it will lead to the choice of a view angle for the camera (Θ_{MAX}) (and thus the value of D_1/p, *cf.* equation 2.2 where D_{chip} is $n \times p$), from which the baseline, b, between the two camera positions as well as the distance at which to place the object to be measured (D_2) are calculated.

Finally, having defined the camera and set the experimental configuration, a practical way of calibrating the camera is given to achieve the desired view angle and the right object distance. A short description of a procedure for reconstructing a scene from a computed disparity map is also given.

4.2 Camera Characterization from a Stereo Pair

4.2.1 Introduction

Digital camera manufacturers usually give very little information on the chip they use inside their devices, especially the 'consumer' versions which potentially could be used to construct economic, portable stereo photogrammetry systems. The size of a pixel and the position of the camera's optical centre are needed for determining the optimum configuration. This section sets out a straight forward procedure for measuring the actual camera parameters. In most cases it will only be possible to measure the ratio D_1/p, but this is adequate for all the formulae. For zoom lenses, most manufacturers give a focal length range, (*e.g.* $3.1mm \sim 46.5mm$) but the accuracy of this information is unknown and there are generally no mechanisms to set the lens precisely to any desired focal length.

The procedure described here only depends on the ability to measure a depth and a distance in the scene, for example from known object sizes, a ruler or a grid on the background. It also provides the distance between the object and the optical centre of the camera: with complex lenses, it is very difficult to determine where that centre is located inside the camera. Examples of the procedure with two different cameras are given in the following two sections.

4.2.2 Camera A: PAL format (768×576), 1/4" sensor

Reported Camera Pixel Size

This camera has a sensor labelled '1/4 inch' in the sales brochures, which is reported by Parsons[1] as a 3.2 mm × 2.4 mm sensor.

The manufacturer's statement on this camera is a resolution of 800×600, but the maximum resolution available for the user's picture capture is 768×576[2] *i.e.* a square pixel size of:

$$3.2 \div 800 = 4\mu m \quad \text{horizontally and}$$
$$2.4 \div 600 = 4\mu m \quad \text{vertically.}$$

Note that even though this information is available for this case, it was hard to find and is not supported by definitive statements from the manufacturer, therefore the following procedure was needed to confirm the reported figures. Neither the pixel size, p, nor focal length, D_1, can be determined separately but their ratio can be determined and suffices in all cases.

Description of a sample scene setup: 'CoinStack'

Two of 'CoinStack' sample images (numbers 0 and 4), shown in figure 4.1, were used in the procedure. A stack of 3 visible New Zealand coins (whose thicknesses are defined by the Reserve Bank of New Zealand [95]) sits on a banknote with 3 hidden coins below the largest to separate the stack from the background.

Two points at two different depths were chosen and their disparities in pixels counted: for example, a point on the T of 'Twenty' and another on the 5 of the topmost 5 cent coin. The two points are surrounded by white circles on the two images in figure 4.1 and are labelled 'T' and '5' in the following discussion. Using Reserve Bank of New Zealand specifications [95], the height of the stack is:

$$\Delta D_2 = 19.5mm$$

The baseline can be read from the position of the right hand edge of the images

[1] See Parsons' history of sensors for a description of the norms and a relation between the labels and actual digital sensor dimensions [96].

[2] For some cameras, the user's picture window is surrounded by masked out pixels used for the camera's self-adjustment of the sensor's black level. In this case the 'complete' resolution of the sensor is known but a value to be trusted is the actual number of pixels obtained when capturing a picture, *i.e.* 768×576 in this case. Note that the difference is minor and represents few tens of pixels on several hundreds, *i.e.* a difference of about 5 %.

FIGURE 4.1: left: CoinStack 0 image, right: CoinStack 4

on the ruler - $104.0mm$ for image 0 and $123.5mm$ for image 4 - giving:

$$b = 19.5mm.$$

It is easy to precisely measure b: in this example every millimetre is represented by about 7 pixels. The field of view of one camera can be read from the ruler:

$$FoV = 106.0mm$$

The two points on image 0 have the following coordinates:

Point	Image 0	Image 4	Disparity (pixels)
P_T	$(218, 390)$	$(75, 390)$	$d_T = 143$
P_5	$(523, 267)$	$(369, 267)$	$d_5 = 154$

Note that good precision can be obtained by using viewing software with a zoom option to determine the points and their disparities.

Using equation 2.5:

$$\begin{cases} d_T \cdot p = \dfrac{D_1 \cdot b}{D_2} \\ \\ d_5 \cdot p = \dfrac{D_1 \cdot b}{D_2 - \Delta D_2} \end{cases}$$

The value of ΔD_2 is known from the height of the coin stack. p, D_1 and b can be eliminated by dividing one equation by the other, leading to a value for the distance

from the optical centre of the camera to a known scene point:

$$D_2 = \frac{\Delta D_2 \cdot d_5}{d_5 - d_T} = 273mm \tag{4.1}$$

D_2 is practically hard to measure as the optical centre lies within the camera itself. Experimentally it is easier to measure the distance from the object to a known point on the camera and deduct the distance between this point and the optical centre: this speeds up subsequent camera setups because the position of the optical centre (depending on the chosen focal length) usually changes by few millimetres. So a user can almost 'preset' the camera at the desired position for the following experiments.

Knowing b and calculating D_2, we can obtain the *critical* camera parameter,

$$\boxed{\frac{D_1}{p} = \frac{d_T \cdot D_2}{b}} \quad = 2.0 \times 10^3$$

Assuming Parsons' data is correct, *i.e.* $p = 4\mu m$, then $D_1 = 8.0mm$ for the setting of the zoom lens used to capture these images.

Theoretical Accuracy Check

A quick approximation to the depth accuracy would be:

$$d_5 - d_T = 11 \quad \text{pixels represent} \quad \Delta D_2 \simeq 19.5mm$$

i.e.

$$\delta D_2 = 1.77mm \cdot pixel^{-1}$$

Using equation 2.7, with both distances, *i.e.* D_2 and $D_2 - \Delta D_2$:

$$\boxed{\delta(D_2 - \Delta D_2) \simeq 1.66mm < 1.77mm < \delta D_2 \simeq 1.92mm}$$

which shows that the quick approximation to the accuracy lies between the accuracies obtained theoretically, confirming the validity of the theoretical analysis.

Experimentally, it is easy to measure accurately two points in the scene and derive the camera parameters and distances from these two points. In this case, readily available objects (coins) were used so that the depth difference between the two points was known.

4.2.3 Camera B: 4MPixels (2288×1712), 1/1.8" sensor

The same process was used with the second camera: a stereo pair was taken of a scene with known depths and corresponding disparities, see figure 4.2; the baseline is $b = 20mm$.

This coin stack contained 4×1 dollar coins: the height of the stack, $\Delta D_2 = 10.96mm$. The two chosen points were: on the background at the corner of the 'T' of 'Twenty' and on the crown on the 1 dollar coin; each is surrounded by a white circle in the two images in figure 4.2.

Point	Left Image	Right Image	Disparity (pixels)
P_T	$(571, 1155)$	$(1046, 1157)$	$d_T = 475$
P_c	$(902, 546)$	$(1392, 548)$	$d_c = 490$

The camera is sold by the manufacturer as a:

- 4.13 million pixels
- with a 1/1.8" sensor given by Parsons [96] to be 7.2 mm $\times$ 5.3 mm.

Practically, the camera returns $2288 \times 1712 = 3917056$ pixels pictures in its biggest resolution.

Not knowing the exact gross resolution but only the gross number of pixels, one can guess the pixel size as:

- the sensor's area is $7.2 \times 5.3 = 38.16mm^2$
- one pixel area is $\frac{38.16}{4.13 \cdot 10^6} = 9.24 \cdot 10^{-6}mm^2$
- one pixel is $\sqrt{9.24 \cdot 10^{-6}} \times \sqrt{9.24 \cdot 10^{-6}} \simeq 3.0\mu m \times 3.0\mu m$

Note this strengthen the fact that except rare cases where a camera is completely known, one has to use *trusted* values like the maximum returned picture resolution in raw mode and calibrate using the D_1/p ratio, manufacturers usually do not give enough information to be sure of the pixel size or a precise enough feedback for focal length.

D_2 and D_1/p are given by:

$$\boxed{D_2 = \frac{\Delta D_2 \cdot d_c}{d_c - d_T} \simeq 358mm} \tag{4.2}$$

$$\boxed{\frac{D_1}{p} = \frac{D_2 \cdot d_T}{b \cdot (b)} \simeq 8.5 \times 10^3}$$

If p is known - $p = 3.0\mu m$ - then $D_1 = 25.5mm$.

FIGURE 4.2: Left: CoinStack2 image 0; Right: CoinStack2 image 4

4.2.4 Summary

This section sets out a method for describing a camera when the required technical specifications are not available. A simple experimental procedure determines the ratio, D_1/p, which appears in all formulae, from a pair of images. Values of D_1 and p separately are not needed.

For cameras equipped with zoom lenses, this protocol could be used to determine ranges for (D_1/p), as well as the associated half view angle, Θ,

$$\tan \Theta_{MAX} = \frac{n \cdot p}{2 \cdot D_1}$$

i.e.

$$\boxed{\Theta_{MAX} = \arctan \frac{n \cdot p}{2 \cdot D_1}}$$

which could be used as a guide to setting up the experiment.

Note that in both cases it is difficult to have clear statements of the camera's details: one can only rely on the maximum number of pixels actually captured by the camera and experimentally determine the D_1/p value. In most cases only partial information is given and only verifiable values are to be trusted.

4.3 Experiment Setup

4.3.1 Introduction

Now that the camera has been described - its D_1/p ratio is known - one needs to determine parameters for the configuration to be used for photogrammetry: the baseline, b, and the object distance, D_2, which achieve the target accuracy (or to know that this accuracy is not achievable with that camera).

4.3.2 Description

This part describes an experiment's set up through the example of the shell measurement. The shell to be measured has an extent of about 20mm and the target accuracy is $\delta_{target} = 0.5mm$ using Camera A.

We know that $3.1mm \leq D_1 \leq 46.5mm$ from the camera specifications. Θ_{MAX} is given by equation 2.2. The common field of view is expressed as a function of the object extent: $a = \rho \cdot 20mm$ where $\rho \geq 1$ indicates how many times the object extent will be seen in the common field of view. A larger ρ leads to a smaller maximum disparity and faster processing.

The best accuracy obtainable for given camera parameters is given by equation 2.15:

$$\delta D_{2_{limit}} = \frac{4 \cdot p \cdot \rho \cdot a \cdot D_1}{\mu^2}$$

where $\mu = (n-1) \cdot p$. Substituting for μ, we obtain:

$$\boxed{\delta D_{2_{limit}} = \frac{4 \cdot \rho \cdot a \cdot D_1}{(n-1)^2 \cdot p}} \tag{4.3}$$

One needs to be able to constrain the search for values for the different adjustable parameters of the experiment. From equation 4.3, D_1/p and ρ can be determined:

$$\frac{D_1}{p} = \frac{\delta_{target} \cdot (n-1)^2}{4 \cdot \rho \cdot a}$$

$$\rho = \frac{\delta_{target} \cdot (n-1)^2 \cdot p}{4 \cdot a \cdot D_1}$$

Using $\rho_{min} = 1$, we can determine

$$(D_1/p)_{max} = \frac{\delta_{target} \cdot (n-1)^2}{4 \cdot a} = 3676$$

$(D_1/p)_{min}$ is the minimum value achievable with a given camera. In most cases, this will have to be determined experimentally using the procedure outlined in section 4.2.

Determining the experiment's parameters

Computing $\delta_{target} - \delta D_{2_{limit}}$ within the defined search range leads to figure 4.3. Two considerations affect the choice of values for D_1 (or D_1/p) and ρ:

- choosing larger values for D_1/p (equivalent to larger values for D_1, since p is always fixed) may reduce image distortions due to wide view angles,
- choosing larger values of ρ reduces the maximum disparity of the object and decreases processing time which usually has a time complexity linear in the disparity.

From here on, 'user' will denote the values chosen by the user for D_1 (or D_1/p) and ρ from the possible values, for instance:

- for camera A, D_1 should be large enough to avoid distortions, so choose $D_1user = 10mm$ (or D_1/p large enough $\simeq 2000$),
- to reduce processing time, use the largest ρ possible - $\rho_{user} \simeq 2$ for the chosen D_1user.

Fixing the 'user' values, it is possible to compute $D_{2_{user}}$ and b_{user} from equations 2.11 and 2.13 knowing ρ_{user} and D_1/p.

Thus, reading from figure 4.3 the experimental configuration should have:

$$\begin{cases} (D_1/p)_{user} & = 3.3 \times 10^3 \\ \rho_{user} & = 2 \\ D_{2_{user}} & = 17.1mm \\ b_{user} & = 4.0mm \end{cases}$$

4.3.3 Summary

This section described the procedure for obtaining practical values for D_1/p, ρ, D_2 and b through the shell example. The target accuracy, δ_{target}, is used to determine if the camera is suitable and leads to the choice of $(D_1/p)_{user}$ (effectively the setting of the zoom lens when one is used) and ρ_{user} which leads in turn to the values for $D_{2_{user}}$ and b_{user} using the formulae given in the geometry chapter.

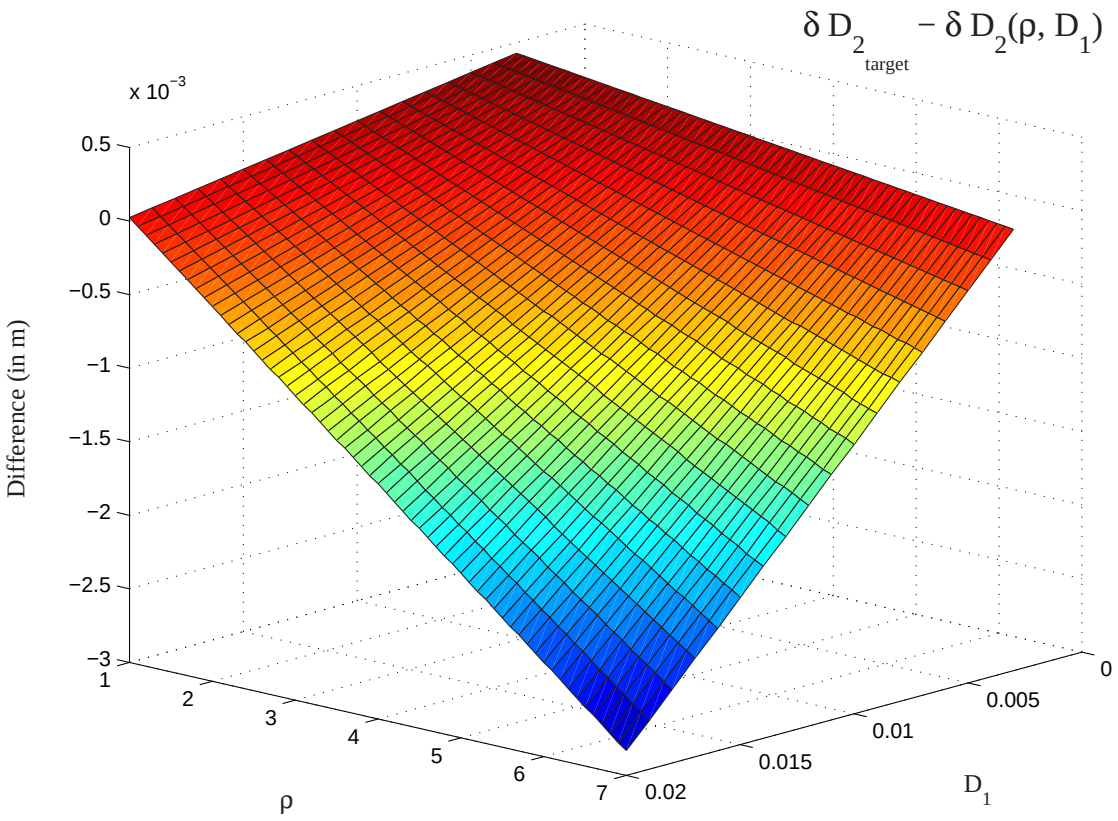


FIGURE 4.3: $\delta_{target} - \delta D_2$ as function of ρ and D_1: Note that in this particular case, a value for p was known, enabling the directly physically meaningful (and smaller - values of D_1/p tend to be $> 10^3$) D_1 to be used: in general, D_1/p would have to be used.

4.4 Camera Setup

4.4.1 Introduction

Digital cameras manufacturers commonly provide a zoom lens without a means of setting the focal length precisely. Adjustment is usually made with a continuous backward/forward control with no feedback indicating the actual focal length. Thus one needs to experimentally adjust the camera settings - particularly the focal length[3]. In all cases, the use of compound lenses makes the location of the optical centre uncertain so that the distance D_2 between the optical centre and the object still needs to be set experimentally.

This section describes a practical use of the theory to set the values determined in the previous section 4.3 where we determined D_1 (or D_1/p) and ρ and then computed D_2, the distance between the object and the optical centre of the lens. The position of the optical centre for the lens system was determined by the procedure in section 4.2.2.

4.4.2 Experimental Procedure

Photograph 4.4 shows the experimental setup:

- the camera is mounted on a tripod on a table,
- the object is mounted on a second tripod on the ground.

It is possible to adjust:

- the levels of the heads of both tripods, see photo 4.5, the tripod on which the camera is mounted has a vertical rack and pinion to adjust and lock its vertical position,
- the object is attached to a horizontal rack and pinion on top of the lower tripod, which is used to precisely adjust the baseline.

4.4.3 Satisfying the Epipolar Constraint

To satisfy the epipolar constraint, the rack and pinion on which the object is installed needs to be aligned so that it moves parallel to the camera's scanlines. The first step is to rotate the head of the lower tripod. A small sheet of graph paper with a 5

[3]Some high-end cameras do have a precise digital focal length adjustment, in which case the setting of D_1 can be skipped

FIGURE 4.4: Camera and rack and pinion tripods setup

mm grid allows easy location of some points, two arrows are used as references and two boxes are drawn around two chosen points (see figure 4.6). On the first (rough)

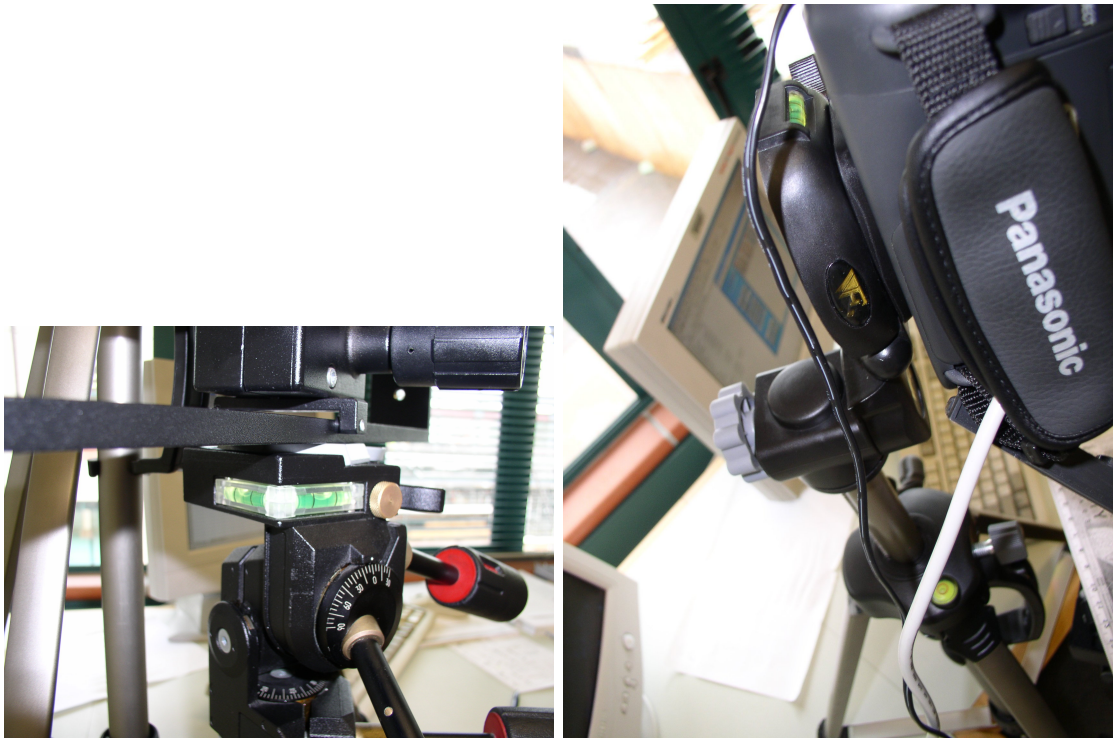


FIGURE 4.5: Left: levels on the sliding bar tripod; right: levels on the camera tripod

alignment, the coordinates for the two points on the first image were $(..., 303)$ and $(..., 306)$ and on the second image $(..., 312)$ and $(..., 315)$. Thus the two images were not epipolar, but from the position of the two shots and the coordinate differences, it is possible to calculate the angle to rotate the tripod's head.

The angular misplacement was $306 - 303 = 3$ pixels for a baseline of a few centimetres. Assuming a pixel size of $4.0\mu m$ for camera A leads to a correction angle,

$$\Theta_{corr} = \arctan\left(\frac{3 \cdot 4.0\mu m}{4cm}\right) \simeq 0.02^o$$

The angular error is a small fraction of a degree: a precise rotation table with a vernier would make it possible to adjust the angle the first time. With this equipment, on the other hand, the tripod's head has a rotation scale marked every 5 degrees, which made the adjustment more tedious. After several trials, the two images in figure 4.7 were obtained: the coordinates for the two points on the first image were $(..., 311)$ and $(..., 310)$ and on the second image $(..., 311)$ and $(..., 310)$ so the object was now aligned parallel to the camera's scanlines.

Note that the adjustment was fairly quick using camera A with 720 pixels on one scanline (a still capture resolution of 720×540 was used for the pictures in figures 4.6 and 4.7), but the manual adjustment became very tedious with camera

B which had 2288 pixels on one scanline, making more precise angle adjustment hardware almost essential for a higher resolution camera.

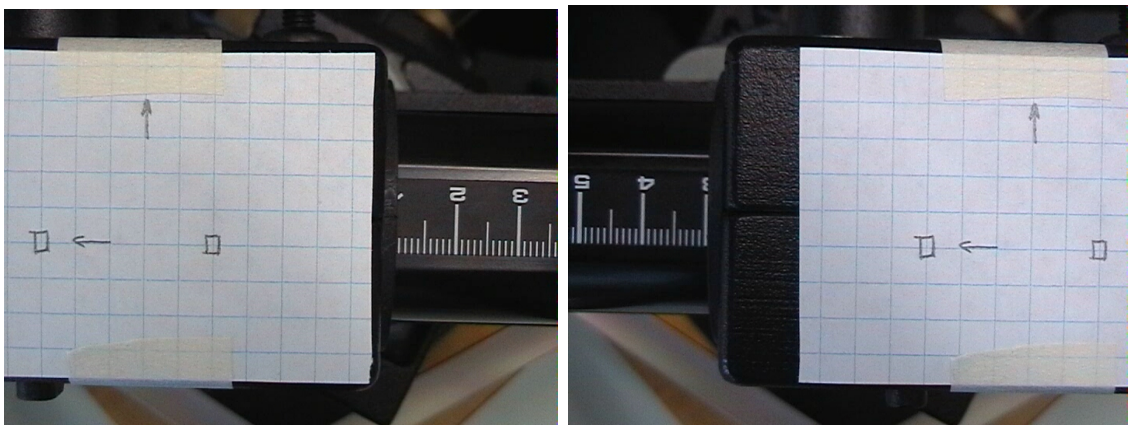


FIGURE 4.6: First (rough) alignment of the rack and pinion supporting the object with the camera's scanlines

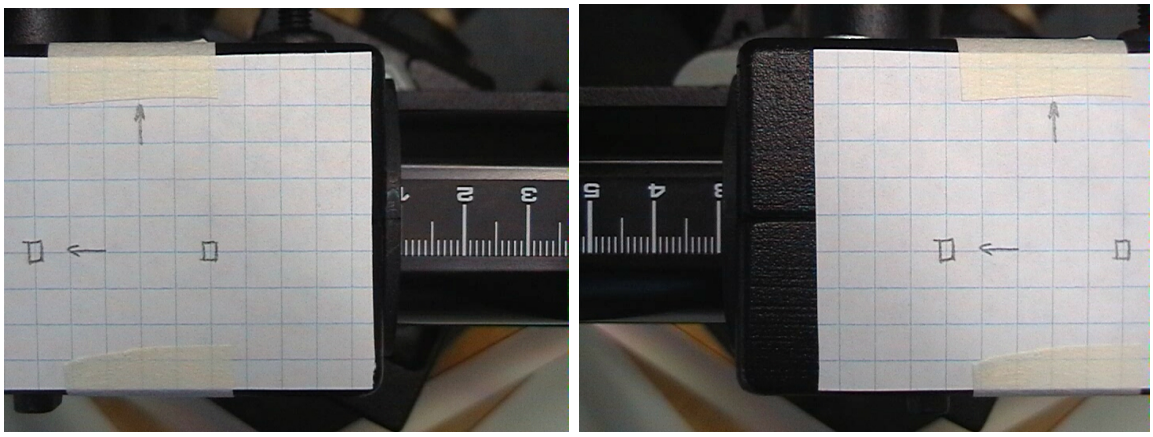


FIGURE 4.7: Rack and pinion aligned with the camera's scanlines

4.4.4 Experimental Focal Distance and Object Distance Setup

This section describes three different ways to adjust the camera parameters (D_1 and D_2) to satisfy the constraint found in section 4.2.

- Method 1 - *one reference point on the background* - is the simplest but incomplete and needs several iterations before obtaining the desired values.
- Method 2 - *two reference points at different known depths* - enables one to obtain the right D_2 in one step but needs iteration to obtain the desired view angle.

- Method 3 - *using two calibration shots* - is designed to adjust the camera parameters in a practical way in one step.

Note that it will be necessary to choose a reference point on an object - the point at which the target accuracy will be achieved. For a 'deep' object (one with a large extent in the direction of the camera optic axes), the accuracy of measured depths at points further from the camera centres than the reference point will decrease as $1/D$. This point will need to be chosen so that the whole of the object (or the region of interest in a general scene) fits into the common field of view.

These two conditions have to be taken into account when setting the accuracy target and the object plane position. In this example, the reference point was chosen in the background plane on which the shell was placed (ensuring that all depths had at least the target accuracy). This assumption suited the shell because it is a shallow object: it fits easily into the narrowing common field of view as the distance from the optical centre decreases.

In all three methods, we know

- b - as it is part of the experiment parameters (see also 2.2.3),
- D_2 - it is computed from sections 4.2.2 or 4.2.3 for instance, and
- the ratio D_1/p, see sections 4.2.2 or 4.2.3 for instance.

Method 1 - One reference point on the object plane

The first possibility is to calculate the expected disparity of a *reference point* (denoted 'ref') on the object plane:

$$d_{ref} = \frac{b \cdot D_1}{D_2 \cdot p}$$

By comparing the measured disparity of the reference point (denoted 'meas'), d_{meas}, with the expected disparity, d_{ref}, it is possible to calculate the direction in which to move the object plane:

$$\begin{cases} \text{if} \quad d_{meas} < d_{ref} \quad \Rightarrow \quad D_{2_{meas}} > D_{2_{ref}} \quad \Rightarrow \quad D_{2_{meas}} \searrow \\ \text{if} \quad d_{meas} > d_{ref} \quad \Rightarrow \quad D_{2_{meas}} < D_{2_{ref}} \quad \Rightarrow \quad D_{2_{exp}} \nearrow \end{cases}$$

At this point, one only knows in which direction to move the object but not by how much, so D_2 has to be adjusted iteratively: moving the object will also modify the field of view at the new depth thus both dimensions (D_2 and Θ_{MAX}) have to be adjusted step by step:

- Adjust D_2 in the direction indicated until the required depth is reached.
- Adjust the field of view Θ_{MAX} to the desired value.
- Changing Θ_{MAX} also changes D_2 but usually by a few mm only. If the new values don't satisfy the accuracy target, then change D_2 and then Θ_{MAX} again until a satisfactory configuration is reached.

Method 2 - Two reference points at different known depths

If a known *second* point on the object is available, it is possible to calculate how far to move the object plane to obtain the desired value for D_2 ($D_{2_{ref}}$). For instance, in the "CoinStack" images, the height of the stack is known - see figure 4.8 as an illustration. Let A and B be two known points on the object. From a first set of images, measure their disparities d_A and d_B:

$$d_B = \frac{D_{1_{exp}} \cdot b}{D_{2_B}} \tag{4.4}$$

where D_{2_B} is distance from the background to the optical centre of the camera and $D_{1_{exp}}$ is the distance from the optical centre of the camera to the image plane.

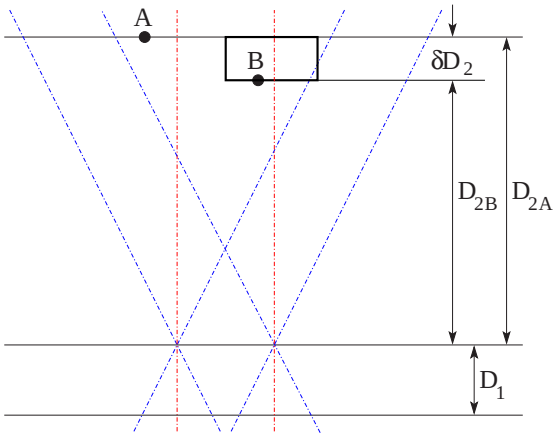


FIGURE 4.8: Method 2: Configuration to determine the position at which an object should be placed when the depths of two different points are known.

Similarly:

$$d_A = \frac{D_{1_{exp}} \cdot b}{D_{2_A}} \tag{4.5}$$

Note that actual values for D_1 and b are not needed as their product appears in both equations and can be eliminated by dividing one by the other to give:

$$\frac{d_A}{d_B} = \frac{D_{2_B}}{D_{2_B} - \Delta D_2}$$

leading to:

$$\boxed{D_{2_B} = \frac{\Delta D_2 \cdot d_A}{d_A - d_B}}$$

By comparing D_{2_B} with $D_{2_{ref}}$, the object can be positioned at the right distance from the camera optical centre.

Here, the accuracy is determined by errors in d_A, d_B and ΔD_2. d_A and d_B are measured on an image to within $\pm\frac{1}{2}pixel$ leading to an accuracy of 0.5% if d_A and d_B are about 100 pixels. ΔD_2 is read from the tripod's rack and pinion position to within $\pm 0.1mm$, *i.e.* 1% if D_2 is $13.5mm$. So, in this example, the accuracy of D_{2_B} is 2.5%.

This method places the object at the desired distance from the camera's optical centre but the field of view angle has to be determined iteratively once the object is at the right distance.

Note that changing the camera's focal length to adjust the view angle also moves the optical centre and D_2 might have to be adjusted again, checked and redetermined. The next section presents a technique for adjusting the field of view angle and distance in one step.

Method 3 - Using two calibration shots

In this method, one moves the camera so as to take two images at two depths (D_{2_a} and D_{2_b}) whose separation, $\Delta D_{2_{ab}} = D_{2_a} - D_{2_b}$, can be measured accurately. This situation is represented in figure 4.9. Compared to the previous method, this one uses measurement of the extent of the field of view at two different depths instead of at a single depth: this enables adjustment of all parameters in one step.

The fields of view at the two depths, $D_{a_{exp}}$ and $D_{b_{exp}}$, are read: for example, by using the image of a ruler (as in the 'CoinStack' example). The measured values are: $D_{a_{exp}}$, $D_{b_{exp}}$ and $\Delta D_{2_{ab}}$, see figure 4.4.4 for the two example shots. Looking at the ruler on the two images shows that $D_{a_{exp}} = 51.0mm$ and $D_{b_{exp}} = 54.5mm$; using reference marks on the camera tripod gives $\Delta D_{2_{ab}} = 13.5mm$.

Thus, with this configuration, the view angle is:

$$\boxed{\Theta_{exp} = \arctan\Big(\frac{D_{b_{exp}} - D_{a_{exp}}}{\Delta D_{2_{ab}}}\Big)} \quad = 14.5^o \tag{4.6}$$

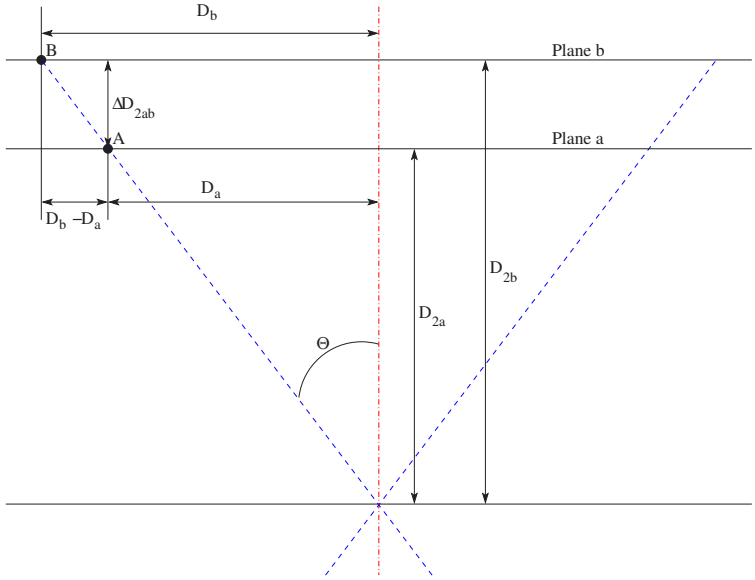


FIGURE 4.9: Method 3: Configuration to determine the position at which an object should be placed

FIGURE 4.10: Two shots at two different depths: only the depth difference needs to be recorded

Using equation 2.4, the reference (expected) value of the view angle is:

$$a = 2 \cdot D_2 \cdot \tan\Theta_{ref} - b \quad \Rightarrow \quad \boxed{\Theta_{ref} = \arctan\left(\frac{(\rho \cdot a) + b}{2 \cdot D_2}\right)} \quad = 13.2^o \qquad (4.7)$$

If both values are clearly different, *i.e.* if the field of view at the desired depth leads to impossible values on the graph in figure 4.3 for ρ, it is necessary to adjust the focal length of the camera so that the *new* field of view:

From figure 4.9:

$$D_{b_{exp}} = D_{a_{exp}} + \Delta D_{2_{ab}} \cdot \tan \Theta_{exp}$$

becomes:

$$D_{b_{ref}} = D_{a_{ref}} + \Delta D_{2_{ab}} \cdot \tan \Theta_{ref}$$

At this stage $D_{a_{ref}}$ is unknown, but can be derived from:

$$D_{a_{exp}} = D_{2a} \cdot \tan \Theta_{exp}$$
$$D_{a_{ref}} = D_{2a} \cdot \tan \Theta_{ref}$$

eliminating D_{2a} from these two equations brings:

$$D_{a_{ref}} = \frac{D_{a_{exp}} \cdot \tan \Theta_{ref}}{\tan \Theta_{exp}}$$

and finally:

$$\boxed{D_{b_{ref}} = \frac{D_{a_{exp}} \cdot \tan \Theta_{ref}}{\tan \Theta_{exp}} + \Delta D_{2_{ab}} \cdot \tan \Theta_{ref}} = 49.2mm \tag{4.8}$$

Knowing both $D_{b_{exp}}$ and $D_{b_{ref}}$, it is now possible to adjust the camera's focal length so that the half field of view becomes $D_{b_{ref}}$ - reading it from a ruler on the image for instance (see figure 4.11).

Note that $D_{b_{ref}}$ is calculated at the *current* distance, D_{2_b}, so it is not necessary to move the camera at this point, only adjust the zoom lens to obtain the desired focal length. See figure 4.11 for the photo of the calibration image after adjusting the focal distance: on the ruler the field of view is $\simeq 116mm - 17mm = 99mm \simeq 2 \cdot D_{b_{ref}}$.

FIGURE 4.11: Calibration image after adjusting the focal distance.

The *current* depth:

$$D_{2_b} = \frac{D_b}{\tan\Theta_{exp}} \quad = 210mm$$

If the *current* D_{2_b} value leads to impossible values for D_1 (in general D_1/p) on the graph in figure 4.3 it is necessary to move the object by:

$$\Delta D_2 = D_{2_b} - D_{2_{ref}} = \frac{D_{b_{exp}}}{\tan\Theta_{exp}} - D_{2_{ref}}$$

i.e.

$$\boxed{\Delta D_2 = \frac{D_{b_{exp}} \cdot \Delta D_{2_{ab}}}{D_b - D_a} - D_{2_{ref}}} \quad = 39mm \tag{4.9}$$

After all these adjustments, it is possible to take to shots with a baseline of $40mm$ and check that the common field of view is $\rho \cdot a = 2 \cdot 20 = 40mm$, see figure 4.12.

FIGURE 4.12: Two shots with a baseline of 40mm to check the common field of view with the desired depth and focal distance

4.4.5 Summary

This section described methods for configuring a camera and an object to provide a required depth accuracy, as well as a alignment of the object movement with the camera's scanlines when a single camera is translated to take both images.

4.5 Reconstruction

Now we need to address the question: how does one compute a 3D representation of the scene knowing the disparity map given as output of the stereovision algorithm? This is the reconstruction step.

Equation 2.5 may be used to express distance, D_2, as a function of the disparity, d, and the ratio of focal length to pixel size, $\frac{D_1}{p}$:

$$D_2 = \frac{D_1 \cdot b}{p \cdot d} \tag{4.10}$$

The disparity map is an inverted representation of the scene in the sense that closer parts of the scene are represented with lower values than farther parts of the scene. Distances derived from experimentally determined disparities using equation 4.10 are referenced to an origin lying in the line connecting the optical centres of the cameras. Appropriate translations to origins in the scene are readily made along with the 'reflection' that removes the inversion in the disparity map when necessary. In applications like obstacle avoidance, the camera optical centres may be a 'natural' origin, making further corrections unnecessary. A simple program can take a disparity map, the camera parameters and distance to the scene origin and compute a 3D representation of the scene.

Figure 4.13 shows an example (taken from Chan *et al.* [49] of a reconstruction. It includes the original left and right images, the computed disparity map using the SAD algorithm with a window radius of 4 and the 3D Model derived from the computed disparity map.

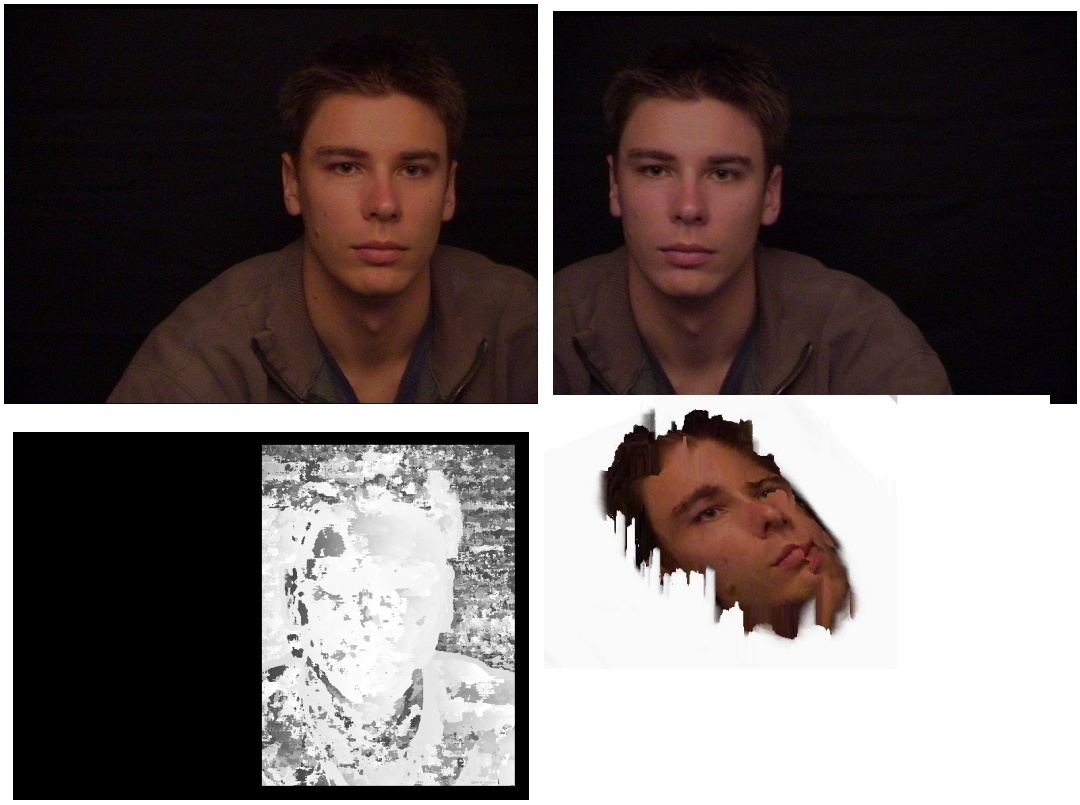


FIGURE 4.13: Top: Left and right image pair.
Bottom: disparity map using SAD with a radius of 4 and the corresponding reconstructed 3D Model.

Note that in this figure , the reconstructed model has been rotated for easier visualization of the 3D structure (of the nose for instance). The pixels not belonging to the face have been masked out median filtered and the values belonging to the face have been thresholded, so that outliers were removed and contrast improved. If this is not done, one incorrect value can lead to an apparently flat face with one point stepping out of it.

Chapter 5

Assessing Stereo Algorithms

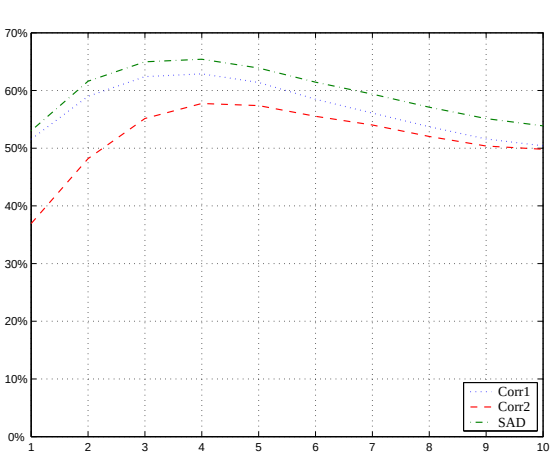
5.1 Introduction

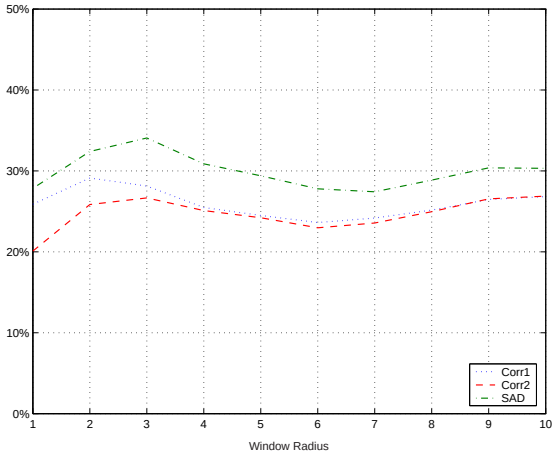
This experiment measured the performance of several stereo algorithms: Corr1, Corr2, SAD, Census and Pixel-to-Pixel. Several sets of images were used: Corridor, Madroom, Map, Sawtooth, Tsukuba, and Venus. Full details of these experiments have been presented at the Image and Vision Processing Conference New Zealand 2002, [97]: this article is reproduced in appendix C so this chapter simply highlights the key results.

5.2 Correlation Algorithms

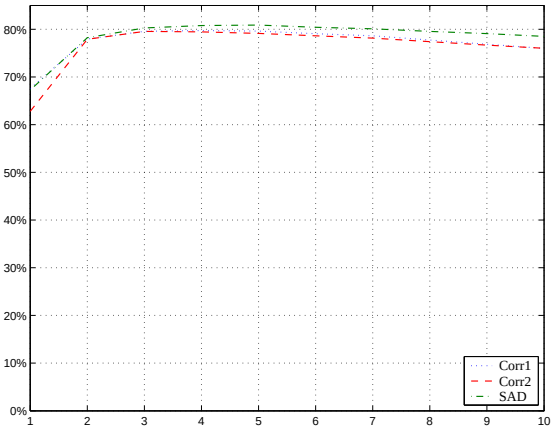
Figure 5.1 shows the results for the three correlation algorithms (Corr1, Corr2 and SAD) in terms of percentages of good matches versus the increasing window radius.

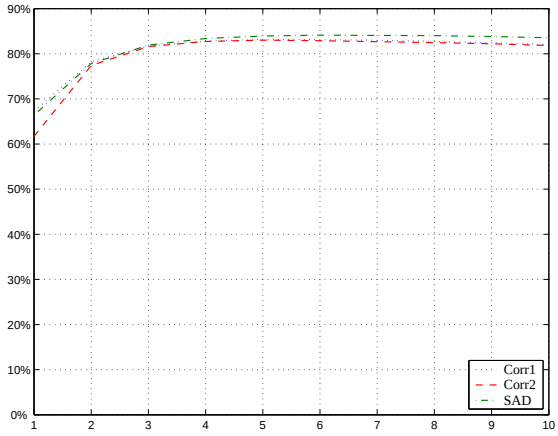
All three algorithms have very similar behaviours. The biggest differences are seen for the Madroom set which is a clear outlier with considerably lower matching success ($\sim$ 30% *vs.*70% or better) for all algorithms because of its stripes configuration. SAD has the same performance as the two other algorithms and would therefore be preferred as it is computationally cheaper - see section 5.5.



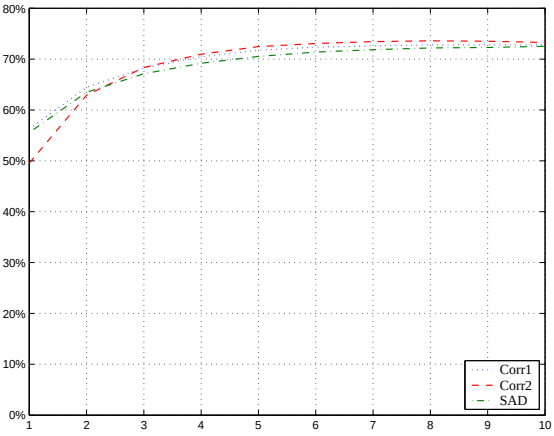


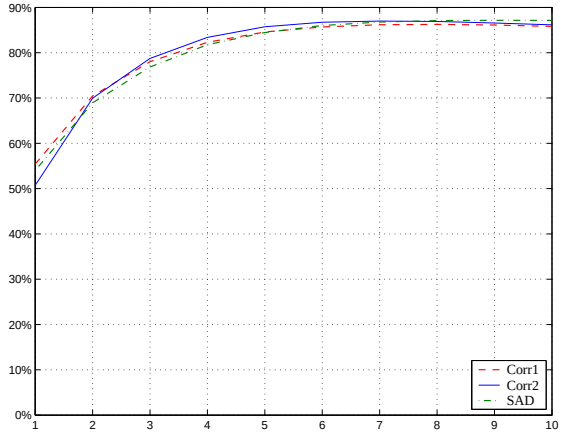
Left: Corridor - Right: Madroom





Left: Map - Right: Sawtooth



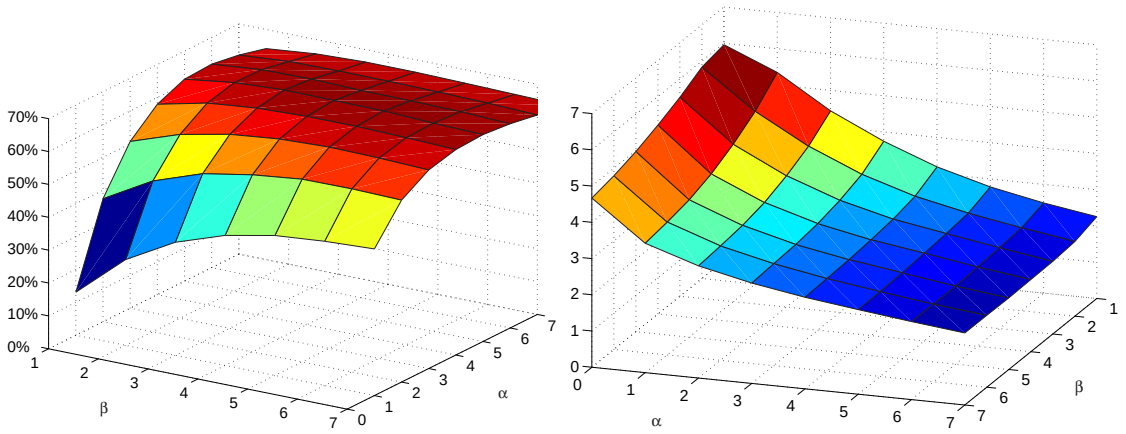


Left: Tsukuba - Right:Venus

FIGURE 5.1: Correlation Algorithms: Percentage of good matches *vs.* the correlating window radius.

5.3 Census Algorithm

Census algorithm performance was plotted against both inner and outer window radii (α and β respectively), see figure 5.2. After reaching reasonable sizes for the two windows, the percentage of good matches plateaus and increasing the window radii does not significantly change the good matches. On the other hand, increasing the window sizes does decrease the standard deviation of the distribution, *i.e.* produce fewer large disparity errors.



Left: Percentage of good matches - Right: standard deviation

FIGURE 5.2: Both graphs are plotted versus the inner and outer window radii (β and α respectively).

5.4 Pixel-to-Pixel Algorithm

The Pixel-to-Pixel algorithm has two main parameters, a cost for an occlusion and a reward for a match (κ_{occ} and κ_r respectively). These two values do not change the algorithm timings, only the behaviour of the cost function. Figure 5.3 shows the good match percentage for the Corridor set versus the parameters (κ_{occ} and κ_r). Pixel-to-Pixel results show a large plateau for which almost any couple (κ_{occ}, κ_r) produces similar results - the optimum being found at ($\kappa_{occ} = 5$, $\kappa_r = 6$).

5.5 Timings and Discussion

Figure 5.4 shows results from timing the algorithms with different parameter choices and table 5.1 gives their complexity. Pixel-to-Pixel on the right has a single because

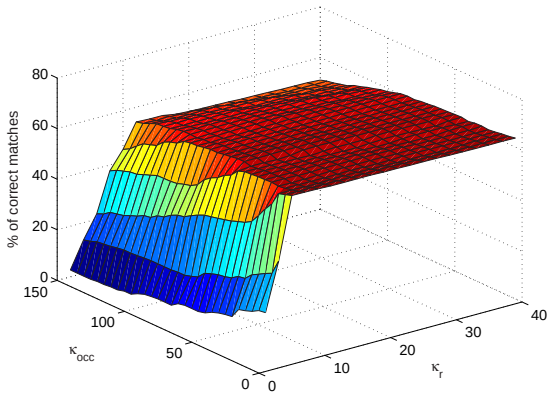


FIGURE 5.3: Pixel-to-Pixel algorithm, percentage of good matches versus (κ_{occ}, κ_r) Corridor stereo pair.

Algorithm	Parameters	Time (sec)	Complexity per point
Census	$\alpha = 4,\ \beta = 3$	58.4	$O(\alpha^2\beta^2\Delta)$
Census	$\alpha = 7,\ \beta = 5$	361	
Corr1	$w = 4$	3.9	$O(w^2\Delta)$
	$w = 10$	20.1	
Corr2	$w = 4$	3.4	$O(w^2\Delta)$
	$w = 10$	16.6	
SAD	$w = 4$	2.0	$O(w^2\Delta)$
	$w = 10$	9.6	
Pixel-to-Pixel	$\forall(\kappa_{occ},\ \kappa_r)$	1.8	$O(\Delta^2)$

TABLE 5.1: Execution time and complexity of the chosen algorithms (w - window radius; Δ - maximum disparity)

its computation time does not depend on the value of either of its two parameters (the reward and occlusion cost). Census, Corr1, Corr2 and SAD were timed with two different parameter sets, the first for the best number of good matches, the second for the lowest standard deviation.

- Correlation and Census algorithms have running times depending on the window sizes: in practical cases a trade off between result quality and computational speed will be needed. All these algorithms have the same behaviour: once a reasonable window size is reached, any increase in radius does not significantly increase the good match percentage but does decrease the standard deviation of the given distribution.

- Census is handicapped in software because of its two nested loops to perform

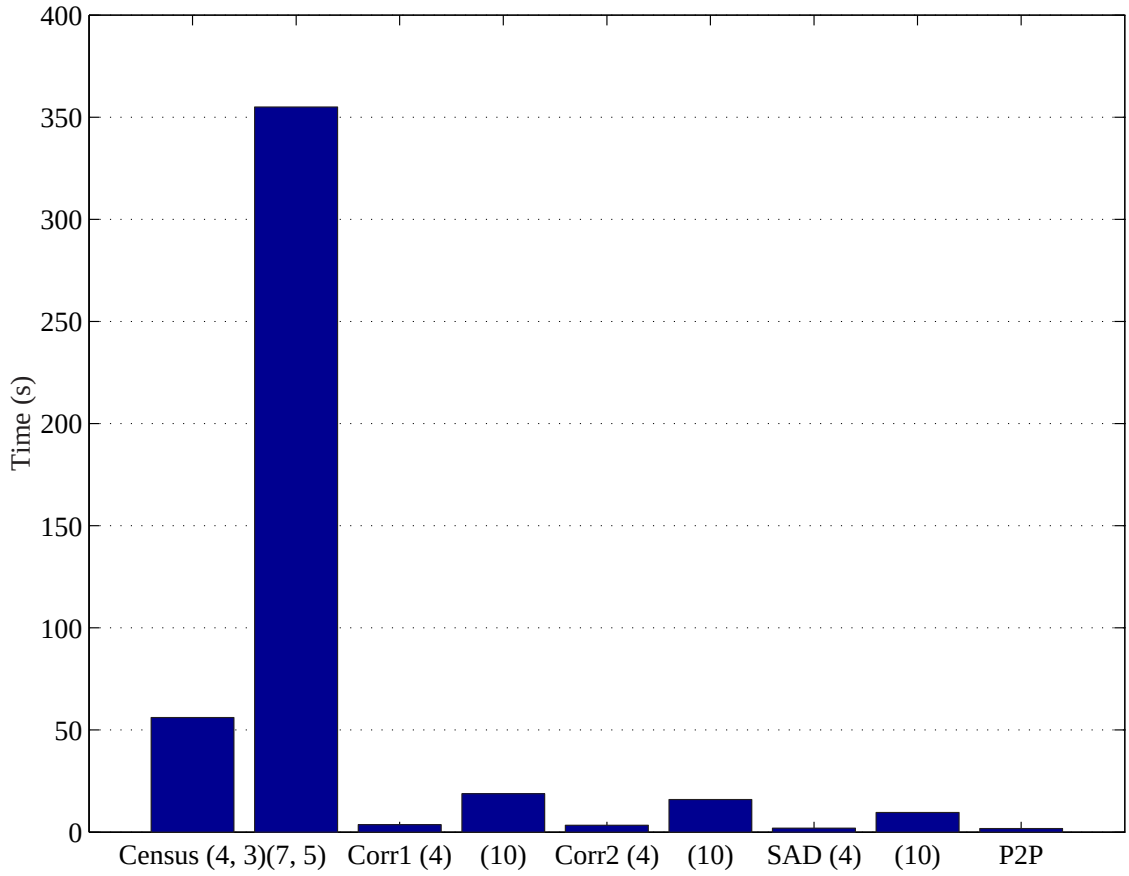


FIGURE 5.4: Execution time for the indicated algorithms and parameter choices.

the transform - see table 5.1, its complexity depends on the product of the squares of its two windows parameters, thus making it slow for traditional coding. It is better suited for hardware implementation where the nested loop computations could be performed in parallel.

- Pixel-to-Pixel is the fastest of the tested algorithms and presents the best quality results. SAD with a window radius of 4 has about the same timing as the Pixel-to-Pixel algorithm. Corr1 and Corr2 are about twice as slow as SAD or Pixel-to-Pixel.

5.6 Summary

This chapter presents the basic performance of the algorithms on the original images using greyscale data. These results were used as benchmarks when assessing the colour and noise experiments described in later chapters. It appears that on traditional architectures, dynamic algorithms perform as well or better on both matching

and speed criteria as other approaches. Although Zabih *et al.*'s 'Graph Cut' algorithm (see 3.2.4) has the best performance in Scharstein and Szeliski's survey [43], it is computationally very intensive - ruling it out as a candidate for the real-time hardware implementations that were the original motivation for this study.

Census does not perform well compared to other algorithms: because of its potential for simple fast hardware implementation, two experiments were conducted to see whether it's performance could be improved - see chapter 6.

Chapter 6

Census Transform Variants

6.1 Introduction

This chapter describes two variations to Woodfill's Census algorithm [33]. Both versions modify the transform in the inner window: the first one weights the different bits according to their distance to the centre of the inner window while the second one uses the middle of each inner window's line instead of the whole window's centre as a reference. The first modification does not improve the matching quality and loses the advantages of the bit patterns given by the original Census transform. The second one keeps bit patterns but enables them to be computed more efficiently in a hardware implementation. It also improves matching accuracy.

6.2 Census Hamming Distance Variant

The motivation behind this experiment was to emphasize the importance of pixels close to the centre of the window by modifying the cost function. The original algorithm uses the Hamming distance - adding 1 if corresponding bits of the left and right vectors differ. In this variant, the contribution of differing bits of the inner window (β radius) transform is weighted depending on the distance from the centre of the window.

The vector is created in the same way:

$$\xi'(P(x,y),P'(x',y')) = \begin{cases} 1 & \text{if } I(P) < I(P'), \\ 0 & \text{otherwise.} \end{cases} \tag{6.1}$$

However, when comparing the n^{th} bit in the left and right vectors (V_{left} and V_{right}),

the contribution to the cost is:

$$\begin{cases} \frac{1}{\sqrt{x^2+y^2}} & \text{if } V_{left}(n) \neq V_{right}(n), \\ 0 & \text{otherwise.} \end{cases} \tag{6.2}$$

where x and y are the coordinates of the n^{th} bit relative to the centre of the inner window. The Hamming distance is replaced by the sum of all these weights.

Figure 6.1 shows the difference in good matches between this variant and the original Census algorithm: positive numbers imply that this variant performs better than the original algorithm. In all cases, the difference between the original Census and this weighted variant are very small: the difference is at most a few percent. The small improvements (when observed) do not justify the use of this variant - especially as it is computationally heavier.

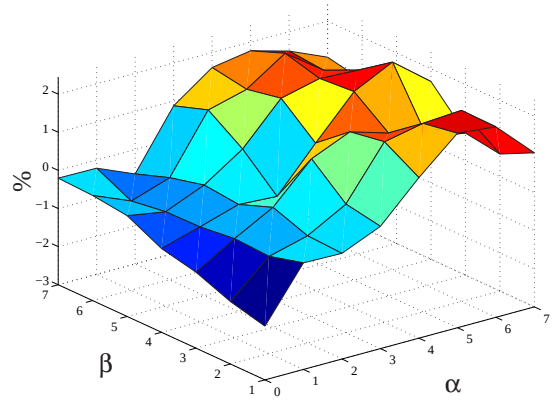
6.3 Line-based Transform Variant

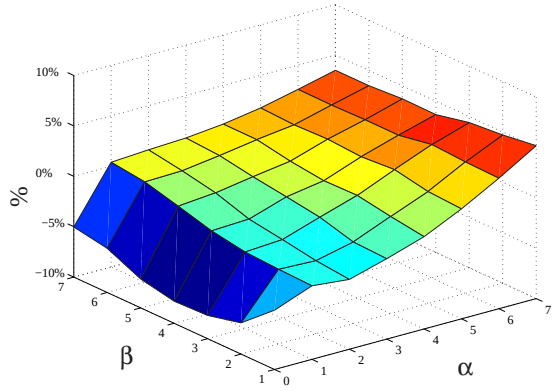
This variant alters the Census transform itself and does produce a small - but significant - increase in matching accuracy for windows above a minimum size. It is based on the observation that, for corrected images, only pixels along individual epipolar scan lines should be matching candidates and that neighbouring scan lines should be used to provide further evidence for a match at a particular disparity value. Therefore, in neighbouring scan lines, pixels are compared to pixels in those lines rather than pixels in the current scan line, *i.e.*the pixel at the centre of the current window. One can observe that the altered transform changes the basis for aggregation of matching costs from the typical square window of simple area-based matching algorithms to a set of lines more typical of dynamic algorithm approaches.

In this line-based transform, the rank transform is derived from the ordering of pixels within a window relative to the pixel at the centre of the same line in the window rather than the centre of the full window, see figure 6.2. For rectified images (*i.e.* those aligned along epipolar lines), this ensures that pixels are only compared with pixels in the same scan line for matching purposes.

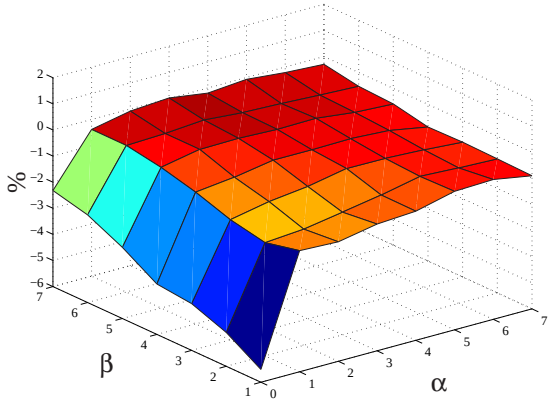
Formally, the rank transform keeps the same core, except that it is computed line by line: the ξ defined in equation 3.5 is replaced by:

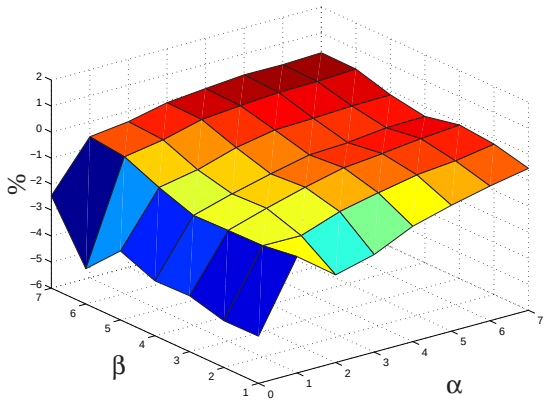
$$\xi'(P(x,y), P'(x,y')) = \begin{cases} 1 & \text{if } I(P) < I(P'), \\ 0 & \text{otherwise.} \end{cases} \tag{6.3}$$



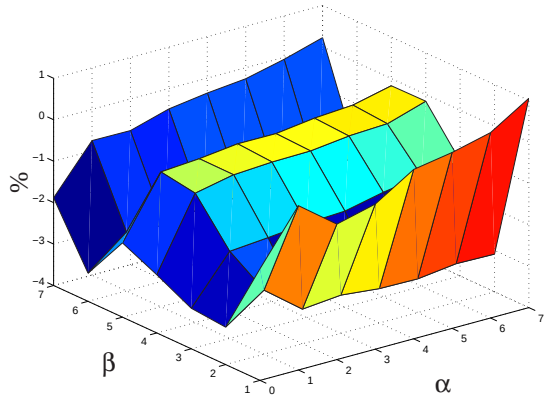


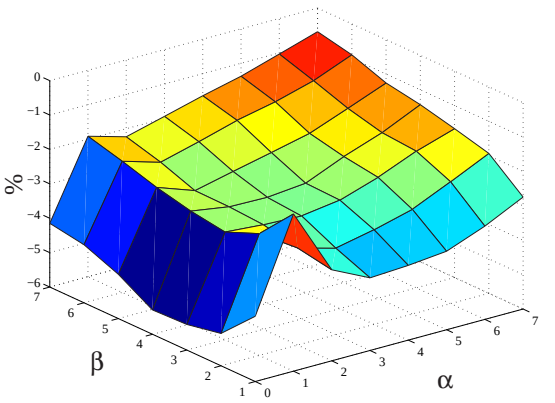
Left: Corridor - Right: Madroom





Left: Map - Right: Sawtooth

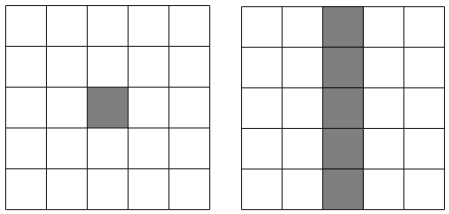




Left: Tsukuba - Right:Venus

FIGURE 6.1: Distance weighted cost function: Differences in percentages of good matches compared to the original Census algorithm *vs.* both inner and outer window radii.

The reference pixel is in the same column as O, the transform centre, and the same line as P' - see figure 6.2(b) - as opposed to the original algorithm where the reference pixel is in the middle of the whole window - see figure 6.2(a).



(a) (b)

FIGURE 6.2: (a) Original algorithm - Window over which the rank transform is is performed in step 1: all pixels are compared to the central grey pixel.
(b) Line-based variant: pixels in each line are compared to the grey pixels in the centre of the same line.

Figure 6.3 shows the improvement in the number of correct matches as a function of the two window sizes, α and β. For small β, the number of matches is smaller or only marginally larger. This is to be expected because the number of significant bits in the original census vector is $4\beta^2 + 4\beta$ *vs* $4\beta^2 + 2\beta$ for this variant. The difference is more significant for small β, *e.g.* for $\beta = 1$, the number of significant bits reduces from 8 to 6 (25%), whereas for $\beta = 5$, the reduction is from 120 to 110 (8%).

Note that larger improvements are observed for small values of α which has a significant effect on computation time, particularly if α can be set to 0, eliminating entirely the outer summation (and a program loop with its overhead) from equation 3.5.

Figure 6.4 shows the improvements obtained for all the images tested for varying values of β when α was set to 0. Improvements are noted for all cases when $\beta \geq 3$ although the improvements are small for the Tsukuba images. The Tsukuba pair's ground truth has only four significant bits - matching its range of possible disparity values, but introducing the possibility that quantization errors are affecting matching.

Note that this line-based variant actually simplifies a hardware implementation in that the first transform is entirely line-based so that only a window width $(2\beta+1)$ of the original pixel values needs to be retained in memory compared to $2(\alpha+\beta)$ rows for the original variant. This results in a significant saving in hardware resources to implement a real-time matching circuit.

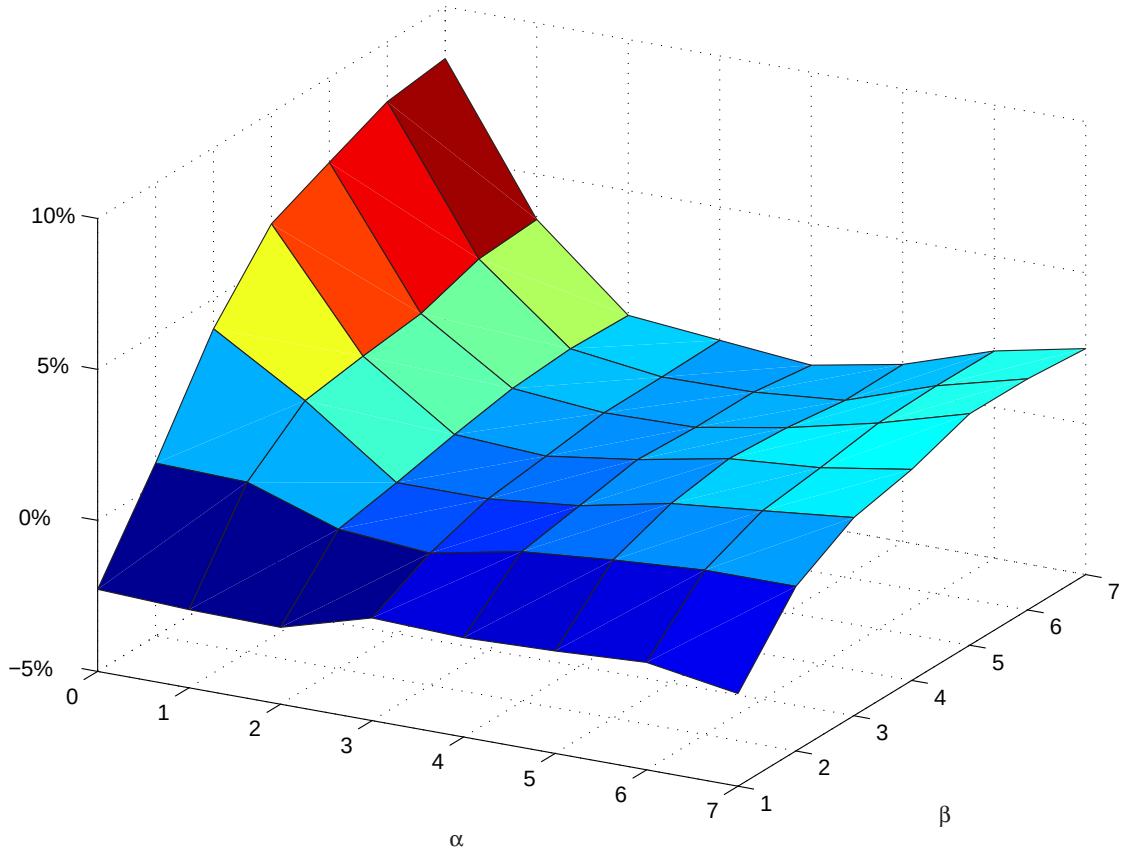


FIGURE 6.3: Improvement in correct matches compared to the original transform *vs.*the two window radius parameters, α and β.
Image: Corridor
Algorithm: Census - Line-based transform variant

6.4 Conclusion

The original Census algorithm may be improved by modifying the way the internal transform is performed so that each pixel is compared to the pixel in the middle of the same line in the window. However, this variant needs reasonable window sizes before it outperforms the standard algorithm as the window carries less information. It has the potential for much more efficient hardware implementations as considerably fewer original pixel values need to be buffered during the compuation. However, as soon as the outside (correlating) window is used, even though it remains generally better than the original, the improvement drops to a few percent.

The potential for very efficient hardware implementations suggests that this line-based variant may be worth additional study. However, in software, I have to point out that the improvements are not great enough to allow a Census algorithm to perform better or faster than several other algorithms examined in this work.

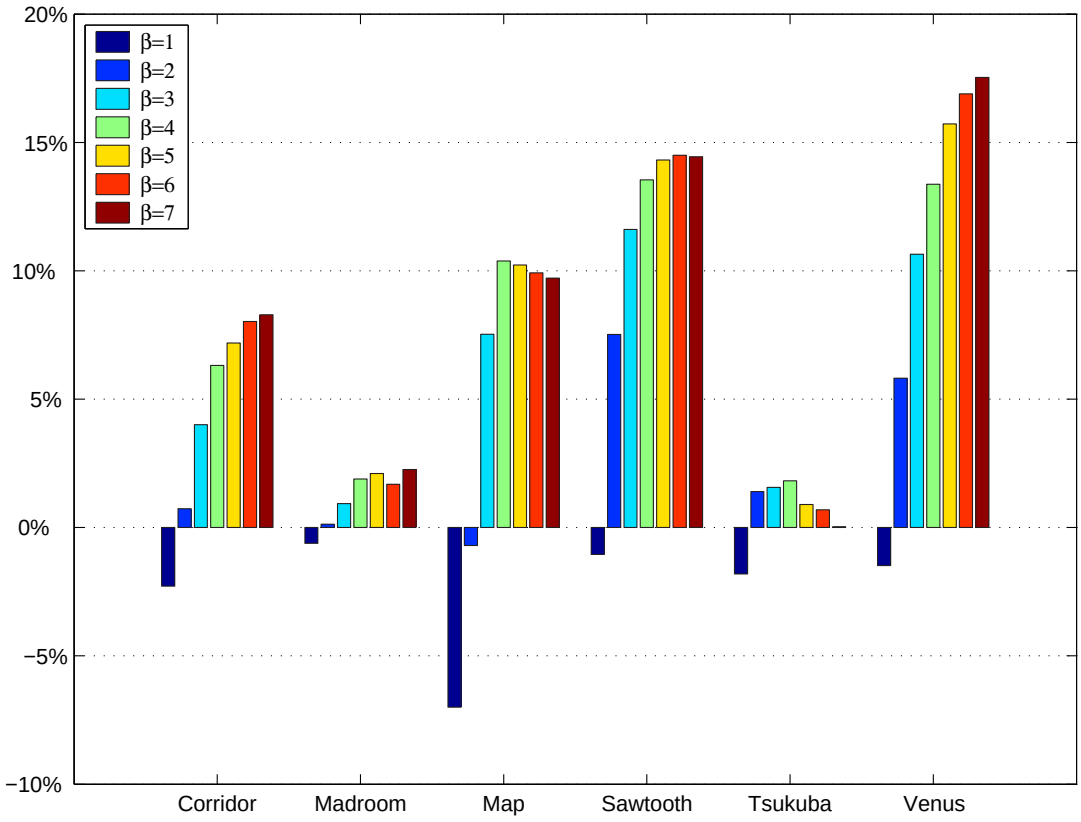


FIGURE 6.4: Improvement in correct matches using the Line-Based transform over the original transform for all the images used *vs.*β: α was set to zero.

Chapter 7

Robustness to Noise of Stereo Matching

7.1 Introduction

The goal of this experiment was to benchmark several algorithms against increasing levels of noise. The 'Corridor' stereo pair was used because it was generated by a ray tracer and does not contain any noise - as would be generated by a camera's sensor and electronics - see section 3.5.3 for more details. This allowed to precisely compute the level of noise - of gaussian distribution - to be added independently to each of the red, green and blue components of the image.

A full discussion of the robustness to noise was presented at the International Conference on Image Analysis and Processing - ICIAP'03 [98] - and has been bound in at appendix D. Therefore this chapter simply summarises the work and highlights the major results.

Note that Koschan evaluated the advantage of using active colour illumination with his pyramidal block matching algorithm against a single value of Gaussian noise, see table 3.14, a change in contrast 3.15 and a change in luminosity 3.16 between the left and right image [4].

The noise level of Additive White Gaussian Noise, AWGN, is given by a signal to noise ratio, SNR, in dB defined as:

$$SNR_{dB} = 10 \cdot \log_{10} \frac{P_{signal}}{P_{noise}}$$

where P_{signal} and P_{noise} are the powers of the user and noise signals respectively. Using ray-traced generated image one can compute the power of the signal and define the needed power of the noise for a given SNR.

Section 3.5.4 shows several examples of noise corrupted images with different noise levels. As a reminder, figure 7.1 illustrates the 'Corridor' with no added noise and corrupted with two different levels of noise:

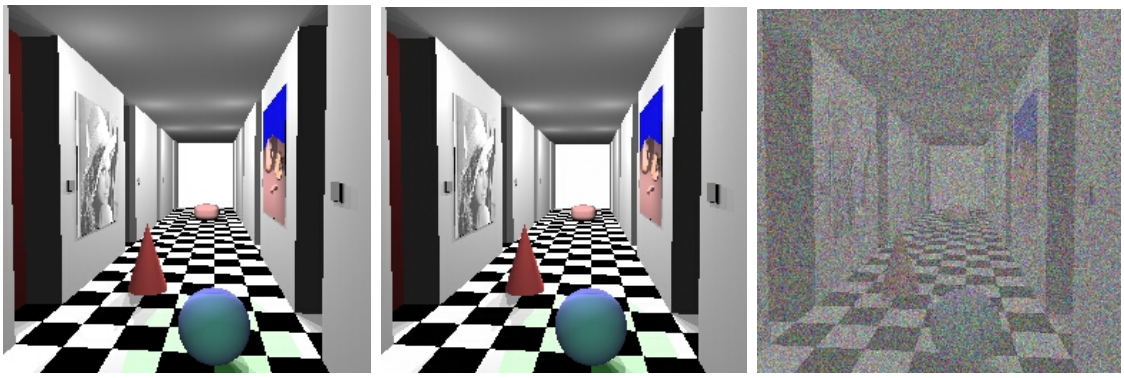


FIGURE 7.1: Left: original Corridor left image,
Middle: Corridor left image with a high SNR (+57dB),
Right: Corridor with an equal part of noise and original signal (SNR=0dB).

7.2 Choice of the Algorithms Parameters

Firstly, from the study in the previous chapter on the behaviour of algorithms - see chapter 5 - on all the stereopairs, two parameters sets were chosen for each algorithm:

- for the correlation algorithms - Corr1, Corr2 and SAD - the sets were:
  - the window radius producing the highest number of good matches: 4,
  - the window radius producing the smallest standard deviation: 10.
- for the Census algorithm:
  - the window radii producing the highest number of good matches: $(4, 3)$
  - the window radii producing the smallest standard deviation: $(7, 4)$.
- for the Pixel-to-Pixel algorithm:
  - the parameters producing the best results for the noise free set: $(\kappa_x = 5, \kappa_y = 6)$,
  - $(\kappa_x = 5, \kappa_y = 40)$, see following note for description.

Note that for Pixel-to-Pixel, the second set of parameters was chosen after noticing that the first set - even though being the best for noise free images - behaved poorly with increasing levels of of noise. One of the significant results from this experiment came from a search for parameters which were more robust to noise - see section 7.4.

7.2.1 Correlation Algorithms Parameters Choice

Chapter 5 summarises the results for the three correlation algorithms on the initial stereo pairs with no added noise. All three algorithms perform similarly. However, SAD is clearly the fastest and is not outperformed. A window radius of 4 always gives the optimum number of good matches percentages. Even for the Madroom pair, which remains an outlier in terms of absolute match performance, SAD outperforms the other algorithms.

7.3 Census Algorithm Parameters

For the Census algorithm, from chapter 5: the noise free results for the corridor stereo pair are shown in figure 5.2. From these results, an inner window radius, $\beta = 3$, and an outer window radius, $\alpha = 4$, were chosen as producing the best numbers of good matches. The smallest standard deviations were found for $\beta = 5$ and $\alpha = 7$.

7.4 Pixel-to-Pixel Algorithm Parameters

Except at the boundaries, Pixel-to-Pixel results show a large plateau: values for (κ_{occ}, κ_r) over a wide range produce similar results, see chapter 5. Initially $(\kappa_{occ} = 5, \kappa_r = 6)$ was chosen as producing slightly better results for noise free corridor images.

Further experiments with other values within the plateau region led to figure 7.2, which plots the percentage of good matches against increasing levels of noise for several pairs (κ_r, κ_{occ}). On this figure, the curve labelled 'Best' corresponds to the *overall* optimum results for all pairs (κ_r, κ_{occ}) at a specified noise level. Even though $(\kappa_{occ} = 5, \kappa_r = 6)$ provides the best results for lower levels of noise, $(\kappa_{occ} = 5, \kappa_r = 40)$ always gave results close to the best obtained for any (κ_{occ}, κ_r) pair ('Best' curve).

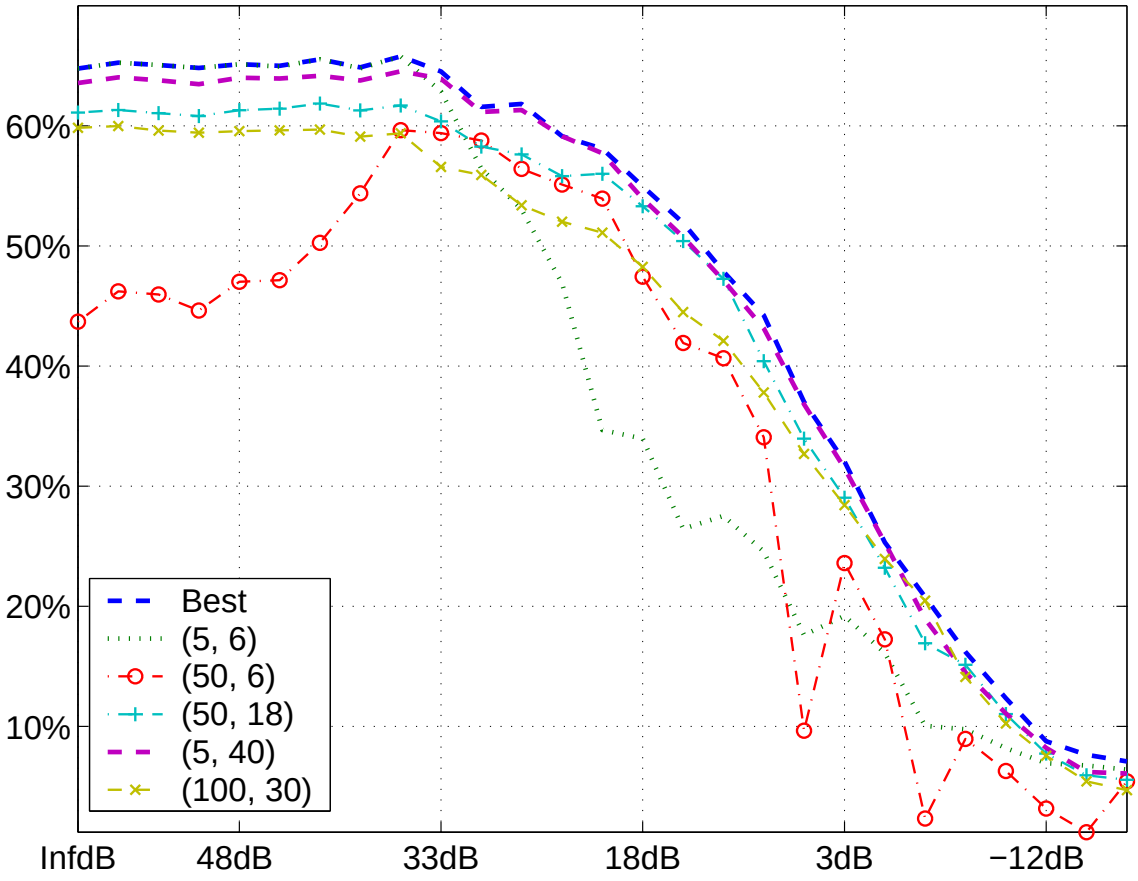


FIGURE 7.2: Percentage of good matches *vs.* noise for several pairs (κ_{occ}, κ_r).
Images: Corridor
Algorithm: Pixel-to-Pixel

7.5 Discussion

Using two sets of parameters chosen by different criteria, the performance of each algorithm against increasing levels of noise is plotted in figure 7.3 (percentage of good matches) and figure 7.4 (standard deviation). It can be seen that:

- Pixel-to-Pixel with ($\kappa_{occ} = 5, \kappa_r = 40$) clearly outperform any of the other algorithms both in terms of good match percentages and standard deviation.
- SAD performs as well as the other correlation algorithms and is computationally cheaper.
- Census clearly gives the worst results of the tested algorithms.

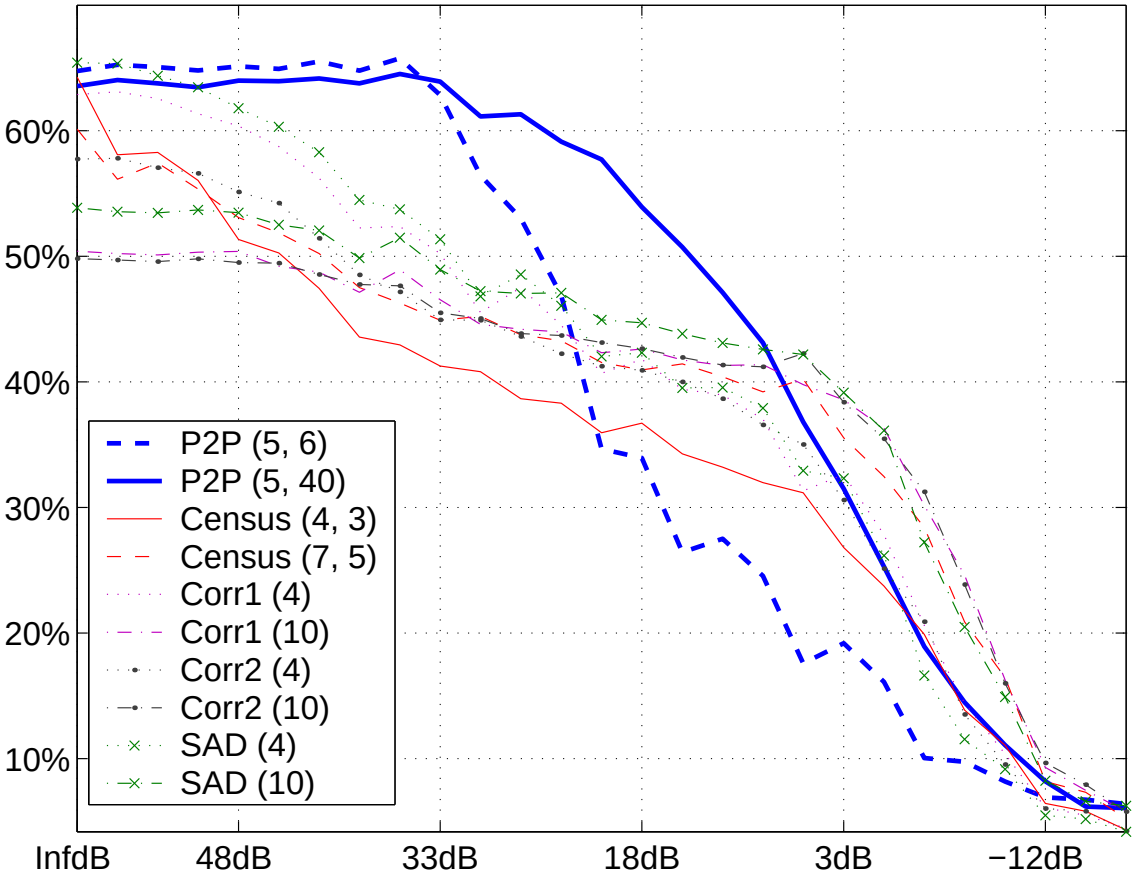


FIGURE 7.3: Percentage of good matches: effect of noise on different algorithms. Images: Corridor with varying degrees of white Gaussian noise added. Algorithms: all.

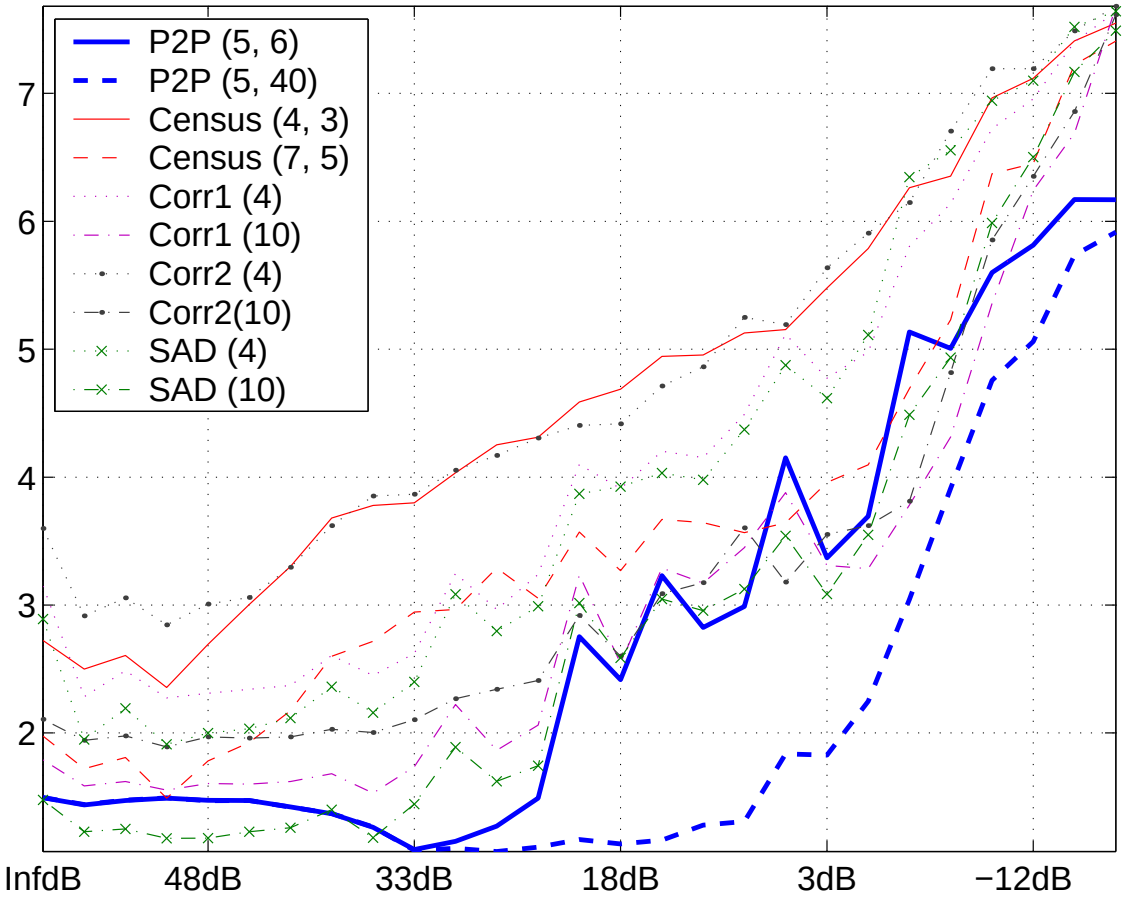


FIGURE 7.4: Standard deviation: effect of noise on the Corridor stereo pair for the indicated algorithms.
Images: Corridor with varying degrees of white Gaussian noise added.
Algorithms: all.

7.6 Speed

In chapter 5, it was established that Pixel-to-Pixel is the fastest of the algorithms, although SAD with a window radius of 4 has similar processing times. The results of this chapter show that Pixel-to-Pixel clearly outperforms SAD in terms of noise robustness.

Note that Pixel-to-Pixel computation times do not depend on the values (κ_r, κ_{occ}).

7.7 Summary

This experiment showed that Pixel-to-Pixel can outperform the other algorithms, *a*s long as its parameters have been chosen carefully - a 'careless' choice based on the noise-free images (where almost any reasonable pair of values produces good results) will perform badly when noise is added. When adding noise, the grey levels get more and more corrupted. Pixel-to-Pixel uses these levels as a first step to occlusions detection, hence the need for an adequate choice of occlusion cost and match reward.

It is worth noting also that Pixel-to-Pixel only processes a line at a time, *i.e.* it only needs one line to be buffered at any time. Other window-based algorithms need to buffer a number of full scan lines determined by the window height (radius for a square window). This means that the number of memory cells needed by a hardware implementation is considerably lower.

Chapter 8

Colour Experiments

8.1 Introduction

This chapter describes experiments where colour information has been used for matching: the correlation cost functions have been modified in two ways to allow the use of colour. Following the experiment description, the results are compared with other approaches found in the literature also using colour for matching. The two approaches used in this chapter are:

- the **combined** use of colour: all three colour bands (red, green and blue) were used in a combination inside the algorithms, *i.e.* the algorithm operated on a combination of the three colours:

$$Cost_{combined} = \overset{\cup}{_{c\in(R,G,B)}} Cost(c) \tag{8.1}$$

- the **separated** use of colour: calculations were performed separately on each colour band and the one providing the best match was chosen:

$$Cost_{separated} = \overset{Best}{_{c\in(R,G,B)}} Cost(c) \tag{8.2}$$

Results are presented for the correlation algorithms, comparing the usual greyscale results versus the combined and separated versions. It would be expected that using colour should improve the quality of matching because a blue and a red pixel should not match even though they have the same luminosity. However, one can note the strong correlation between colour values and greyscale:

$$Greyscale = \frac{R+G+B}{3}$$

8.2 Results

Figures 8.1, 8.2, 8.3, 8.4 and 8.5 show the results for:

- combined, separated and greyscale (*i.e.* standard) versions of the:
- Corr1, Corr2 and SAD correlation algorithms, on the
- Corridor, Madroom, Sawtooth, Tsukuba and Venus stereopairs.

Note that the map stereopair has not been used because it contains no colour. The Corridor and Madroom sets contain very little colour, therefore the difference between colour and greyscale processing is not expected to show big differences.

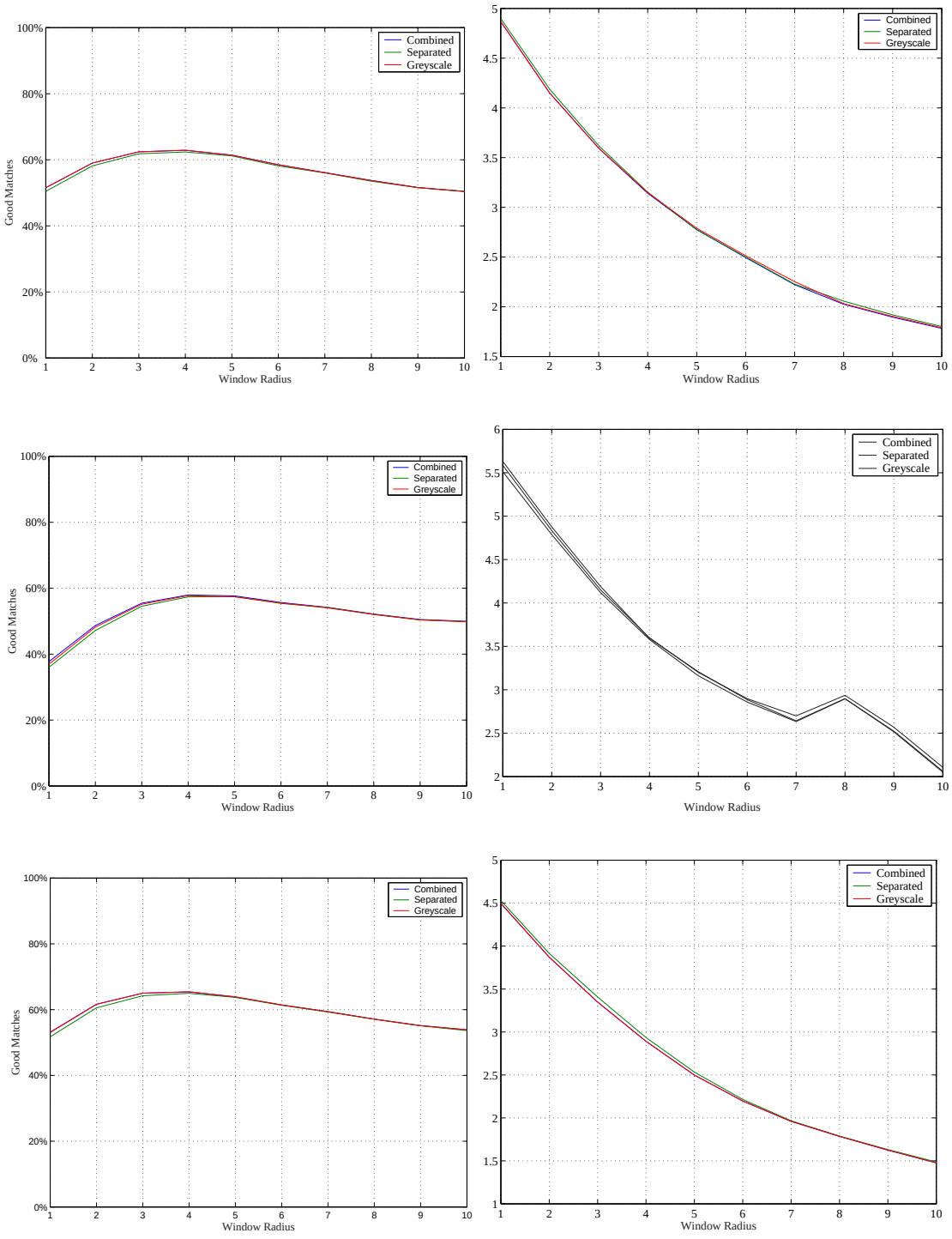


FIGURE 8.1: Corridor stereopair: Corr1 (top), Corr2 (middle) and SAD (bottom) - Good matches (left) and Standard deviation (right).

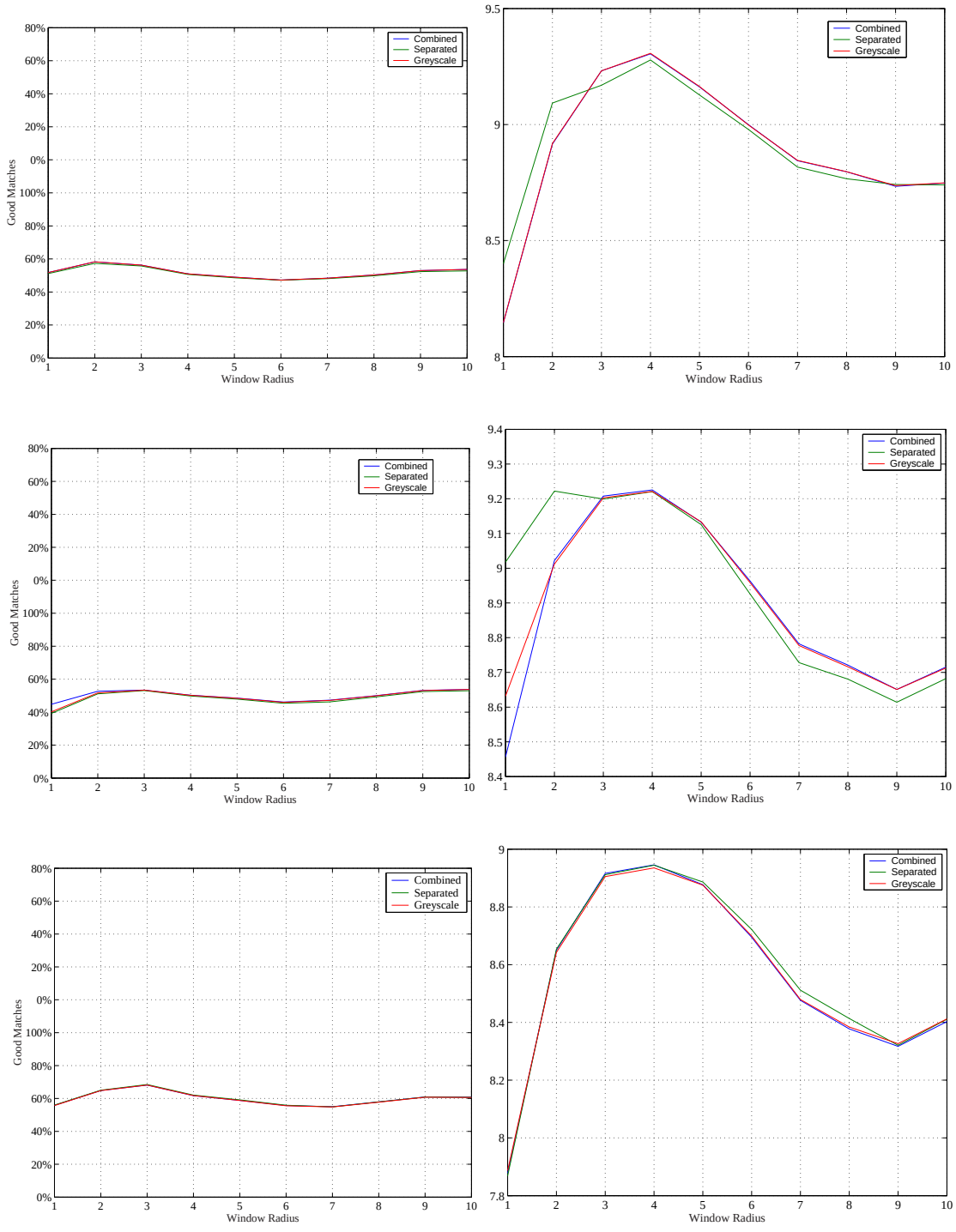


FIGURE 8.2: Madroom stereopair: Corr1 (top), Corr2 (middle) and SAD (bottom) - Good matches (left) and Standard deviation (right).

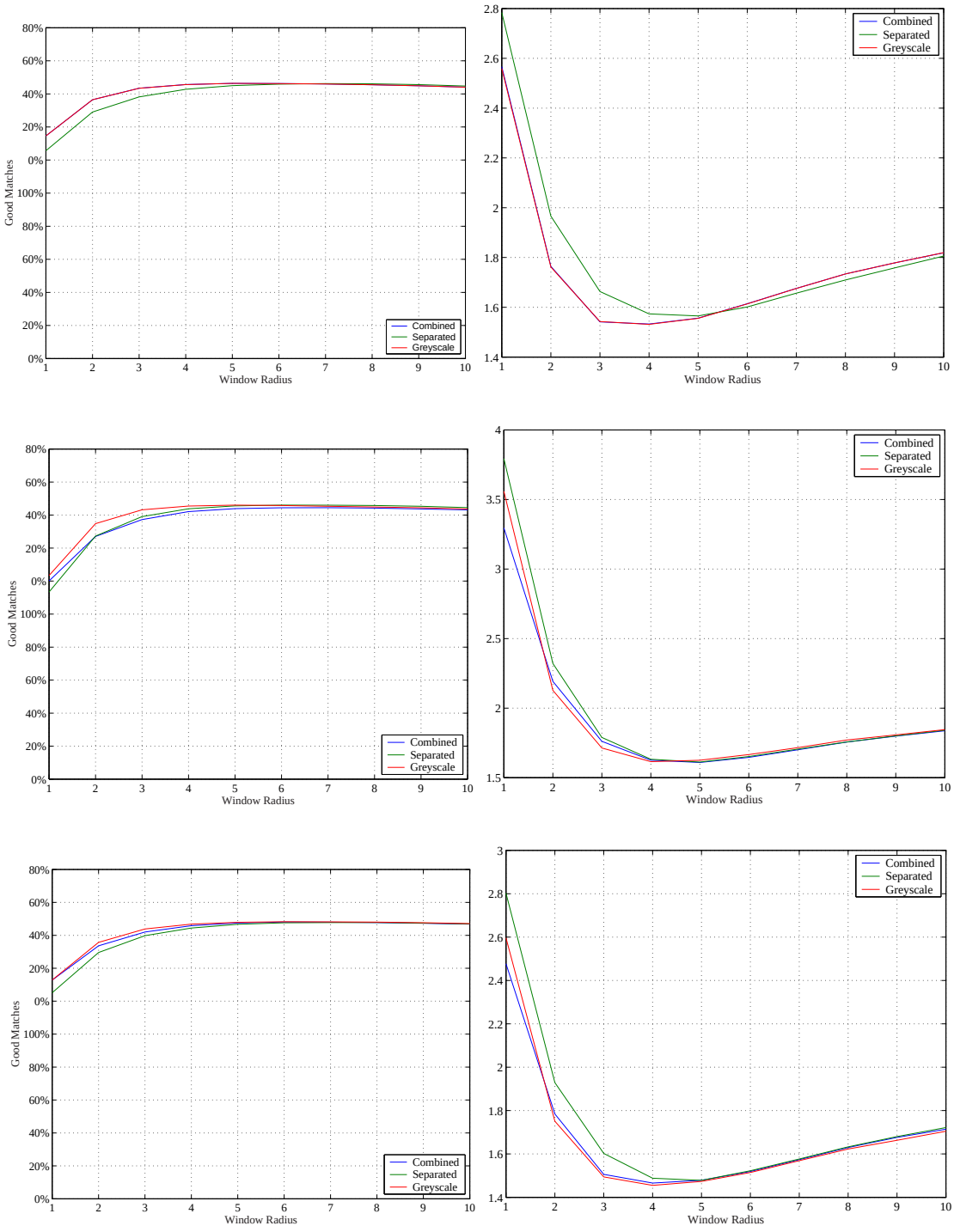


FIGURE 8.3: Sawtooth stereopair: Corr1 (top), Corr2 (middle) and SAD (bottom) - Good matches (left) and Standard deviation (right).

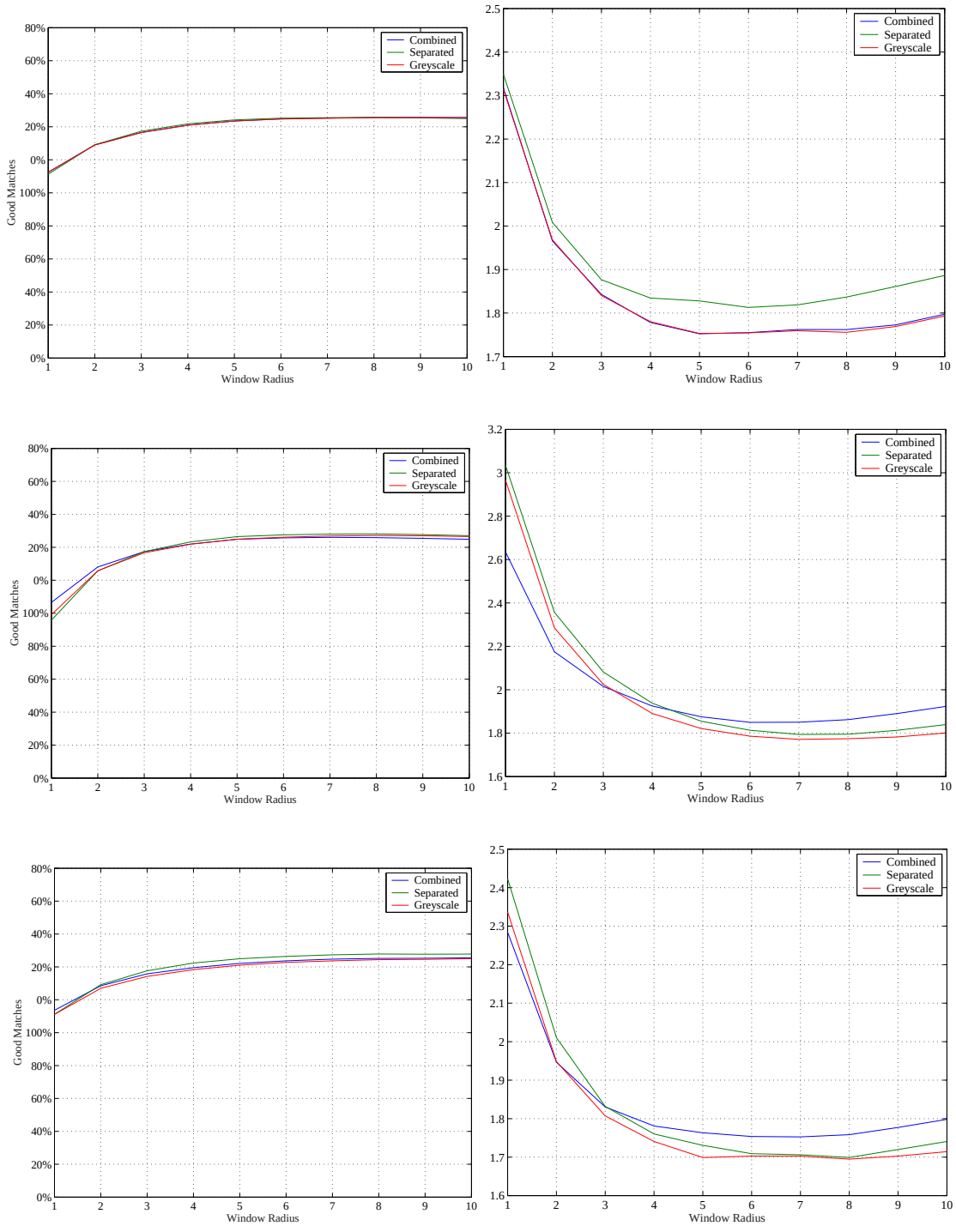


FIGURE 8.4: Tsukuba stereopair: Corr1 (top), Corr2 (middle) and SAD (bottom) - Good matches (left) and Standard deviation (right).

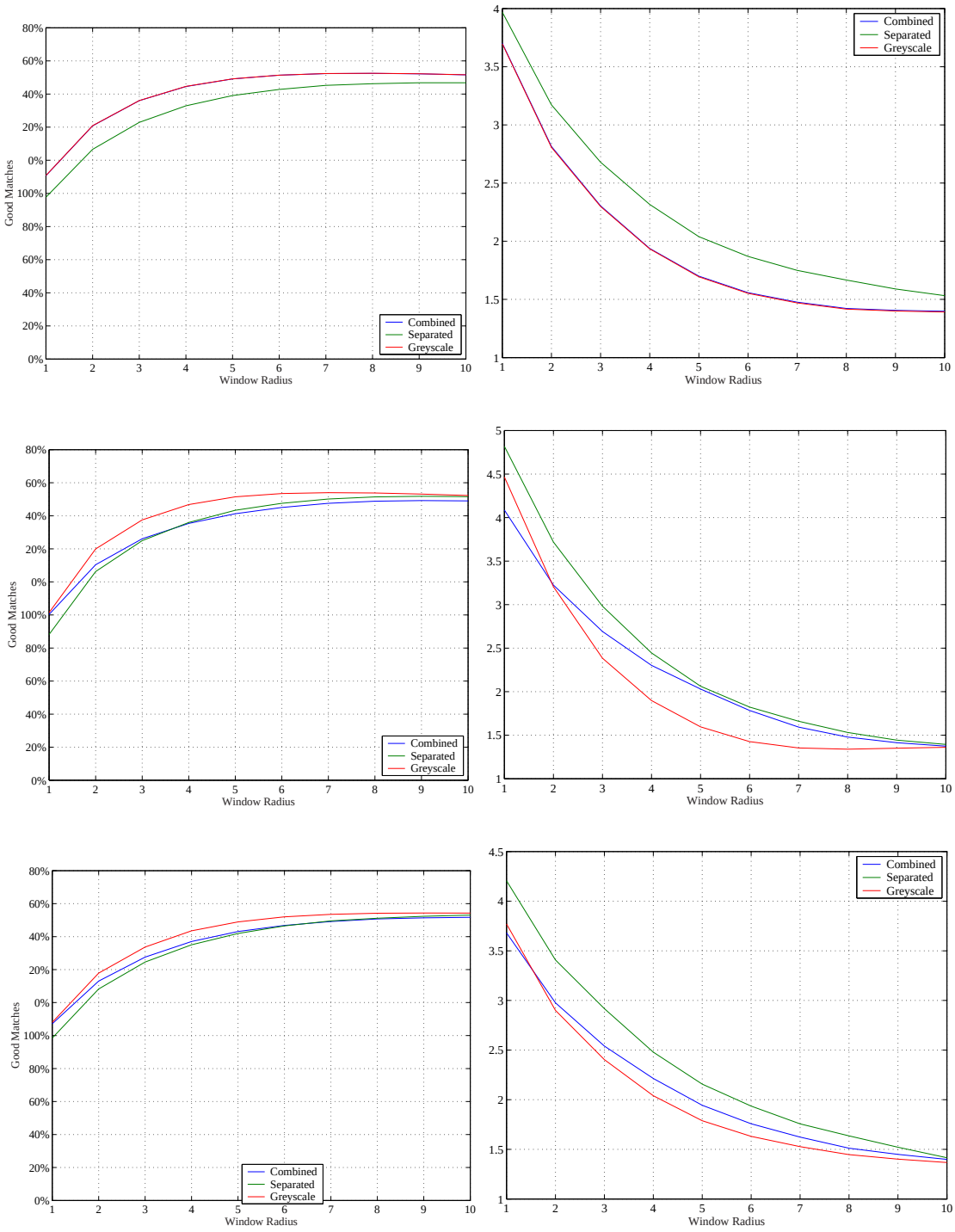


FIGURE 8.5: Venus stereopair: Corr1 (top), Corr2 (middle) and SAD (bottom) - Good matches (left) and Standard deviation (right).

In general, the combined versions present similar results to the greyscale and the separated versions present lower quality results. However, contrary to expectation, the differences were very small in all cases. It appears that:

- combining the red green and blue values is not much different from the greyscale - *i.e.* another combination of red, green and blue - values and that

- the separated versions provide worse results because the information in the *single* chosen colour band is not as useful as in any way of combining the *three* colour bands, for instance the greyscale combination.

8.3 Discussion

Different uses of colour for stereo matching have been proposed in the literature and are discussed here for comparison: notably, Jordan and Bovik and Koschan *et al.* - see section 3.2.5 - both claim improvements of the matching quality with the use of colour. This is contrary to the results presented here, illustrating the strong correlation between colour and greyscale values. Thus comparing this chapter and other's work show that the use of colour is not trivial and needs further careful study.

8.3.1 Jordan and Bovik

Jordan and Bovik - see section 3.2.5 - used a colour based gradient to qualify a possible match and did not use colour directly in a matching function. They chose this method instead of computing zero-crossings on each of the three colour bands as they contend that each colour band zero-crossing is strongly correlated to the greyscale ones.

Note that in this kind of approach the colour is used to reinforce the existing greyscale matches and increase the percentage of unique matches and also the quality of the matching. However on two different image sets this unique matches increase has a very large range: from +1% to +163%. This illustrates that colour contribution to matching is difficult to clearly define, and is strongly dependant on the situation.

Note that this increase's range - +1% to +163% - joins this chapter's results where the use of colour does not necessarily improve the matching results of an algorithm.

8.3.2 Koschan *et al.*

Koschan - see section 3.2.5 - works on blocks and segments the image into chromatic and achromatic blocks[1]: a block is declared as chromatic or achromatic depending on a threshold percentage of chromatic pixels within this block.

The block matching algorithm modified for chromatic/achromatic blocks was claimed to perform always better - by 25 30% - than the the standard approach using greyscale values.

8.4 Conclusion

This chapter described an experiment on the role of colour in stereo matching. Two different approaches were applied to three correlation algorithms but they lead to insignificant improvements despite expectations.

¿From other experiments reported in the literature, it appears that the way colour is used as well as the algorithm itself has a strong impact:

- Koschan *et al.* reported a 25 30% increase using colour inside a block matching algorithm, showing that different types of algorithms have different behaviours with or without colour. The way colour was used - defining chromatic/achromatic blocks *vs.* combined or separated bands - might be the cause of this improvement.

- Jordan and Bovik use the colour information, *i.e.* gradients, to increase the number of unique matches, *i.e.* select the best match from a number of candidates found by the greyscale matching alone, thus increasing the matching quality. However the impact of this method has a wide range from +1% to +163%.

The poor improvements made by colour in the combined and separated experiments might result from the combination of correlation algorithms with strong correlated values between colour bands and greyscale values.

It would appear obvious that failing to use the colour information is ignoring information which could be used to improve matching. It is possible to propose (quite realistic) synthetic scenes (an object of one colour on a background of different colour where the two colours have the similar saturation levels) in which colour

[1]Intensity and saturation values only were used in the HSI colour space to make this segmentation: the hue value is complex to compute from the RGB values usually obtained from a digital camera.

information would be vital to obtaining any matching at all, yet on three sets of 'real' colour images, greyscale information produces similar results to that obtained from colour images. Thus it would appear that further independent experiments are needed to resolve this issue.

Chapter 9

Baseline Modification Effects

9.1 Introduction

The baseline is an important, easily adjustable, parameter in a stereo vision configuration. Chapter 2 discusses the geometry and describes the best baseline choice for a given object at a given depth. This experiment uses MRTStereo's capability to generate ray-traced stereo pairs with increasing baselines to study the influence of the baseline length on matching and to provide experimental confirmation of the issues discussed in chapter 2.

9.2 Matching *vs.* Baseline Length

The two algorithms giving the best and reliable results through the whole study were used: SAD with a window radius of 4 and Pixel-to-Pixel with ($\kappa_{occ} = 5, \kappa_r = 40$). Stereo pairs with baselines from 10 cm to 90 cm in steps of 10 cm[1] were generated using the MRTStereo software.

As the baseline increases, the range of possible disparities also increases. This implies that, if edge effects are to be eliminated from the experiments (see section 3.5.2), the central matching window has to be reduced in size. Table 9.1 lists the number of pixels used in the matching experiments and figure 9.1 shows the percentage of good matches versus the baseline length for two algorithms, SAD($w = 4$) and Pixel-to-Pixel($\kappa_{occ} = 5, \kappa_r = 40$).

From figure 9.1, it is seen that increasing the baseline clearly handicaps the matching algorithms. As the baseline increases, objects are seen from increasing angles, this has several effects:

[1] In the MRT scene description files - see appendix B - the distance unit is cm. Note that the Corridor set used in the rest of the study was generated with a baseline of 20 cm.

Baseline	Number of pixels
10	44280
20	41904
30	39528
40	37152
50	34776
60	32400
70	30240
80	27864
90	25488

TABLE 9.1: Number of pixels able to be used for each baseline value.

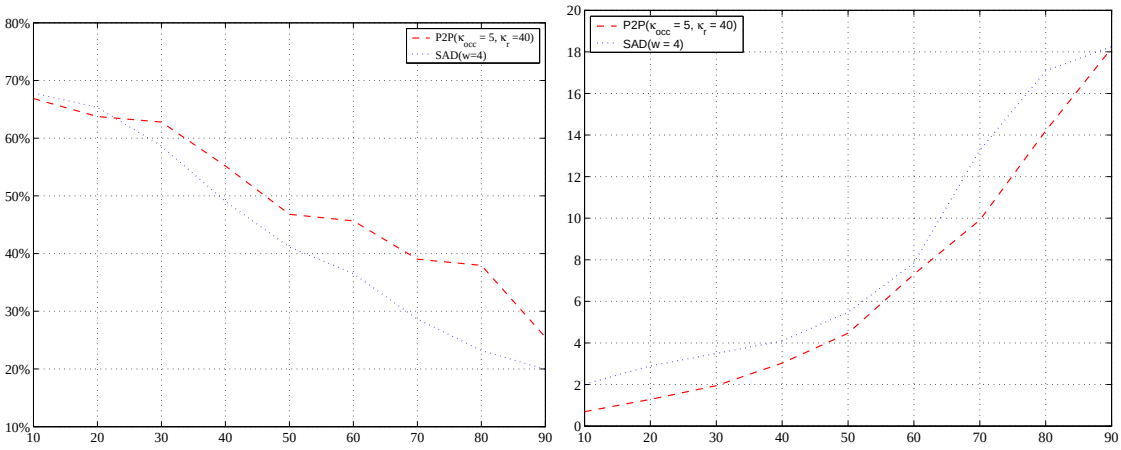


FIGURE 9.1: Percentage of good matches (left) and standard deviation (right) for Pixel-to-Pixel($\kappa_{occ} = 5, \kappa_r = 40$) and SAD(w = 4) *vs.* baseline.

- firstly, it increases the number of occlusions, *i.e.* the number of pixels seen in one image and invisible in the other one (see figure 3.21 in section 3.5.4 for an illustration of the occlusions),

- secondly, simple matching algorithms assume that an object reflects rays with uniform intensity over all angles, *i.e.* they are Lambertian or perfect scatterers. For typical scattering (rough) surfaces, the Lambertian assumption holds well enough over small angles allowing simple matching algorithms to perform well with small baselines, but as the baseline increases this assumption will become harder to justify with real objects, and

- thirdly, the number of pixels subtended by a surface on an image plane is a function of the angle between the surface and the rays from the image through

the camera's optical centre. Again, for shorter baselines and thus smaller differences in the directions of rays through the optical centres of the two cameras, this factor only presents a problem for surfaces at high angles to the image planes. However, as the baseline increases, the 'size' of a plane projected onto an image may start to differ significantly even for planes at quite small angles to the image planes.

The influence of the baseline described by the graphs on figure 9.1 shows that:

- for small values of the baseline, SAD has a slight advantage ($1 \sim 2\%$) in terms of good matches percentages against Pixel-to-Pixel. However, comparing the standard deviation for the two algorithms shows that Pixel-to-Pixel always has a better distribution than SAD, and
- for increasing values of the baseline, both algorithm have worse and worse results both in terms of a smaller good matches percentage and bigger values of the standard deviation.

Note that Pixel-to-Pixel explicitly considers occlusions which explains its better behaviour with increasing baselines. For small baselines, SAD's small good match percentage advantage is because it matches pixels one by one: the view angles are similar, so correlation works fine. On the other hand, Pixel-to-Pixel optimises a path on a full line and has a better error distribution, with smaller numbers of large errors.

9.3 Depth Accuracy *vs.* Baseline Length

9.3.1 Determination of D_1/p

Figure 9.3 shows the value of D_1/p determined for 17 points shown in figure 9.2. These values are also reported in table 9.2) with their corresponding disparity from the ground truth file. Using distances derived from information in the 3D scene description file, the D_1/p value and the mean computed, leading to:

$$\boxed{D_1/p = 223.9 \pm 1.6(1\sigma)}$$

Note that the uncertainty in figure 9.3 comes from the image discretisation or pixelisation. The points chosen to create figure 9.3 were taken at the intersections of tiles - see figure 9.2 - for a easy localisation in the 3D scene. These pixels end up being black or white depending on the predominant colour over the area covered

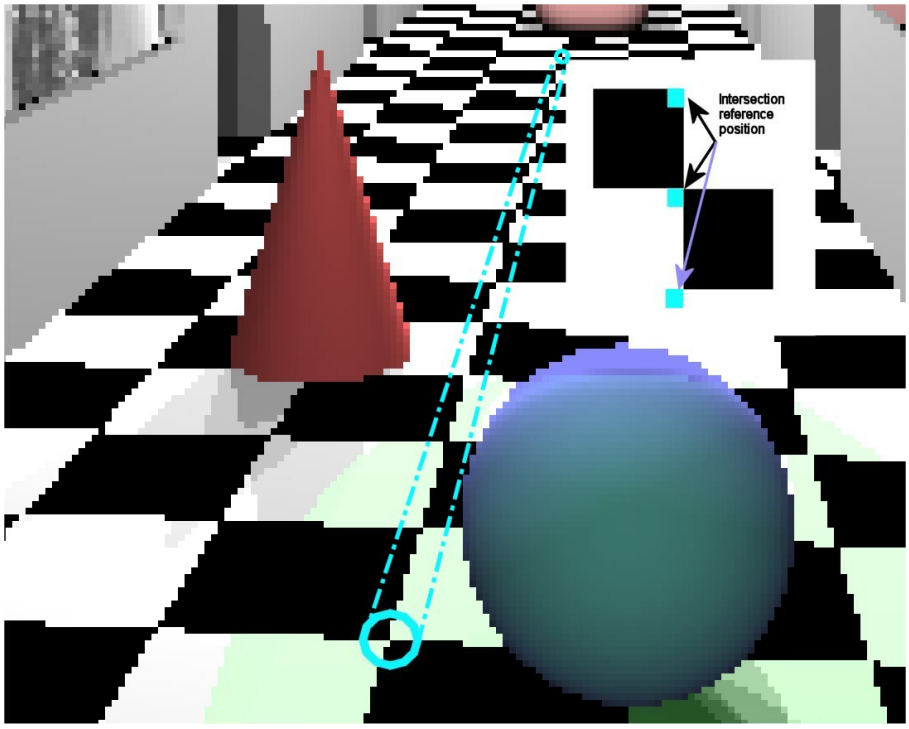


FIGURE 9.2: The blue circle shows the position of the first of the 17 selected pixels: the remainder are at the intersections enclosed by the dotted line. The inset shows which pixel of each intersection has been consistently chosen: the lower left pixel.

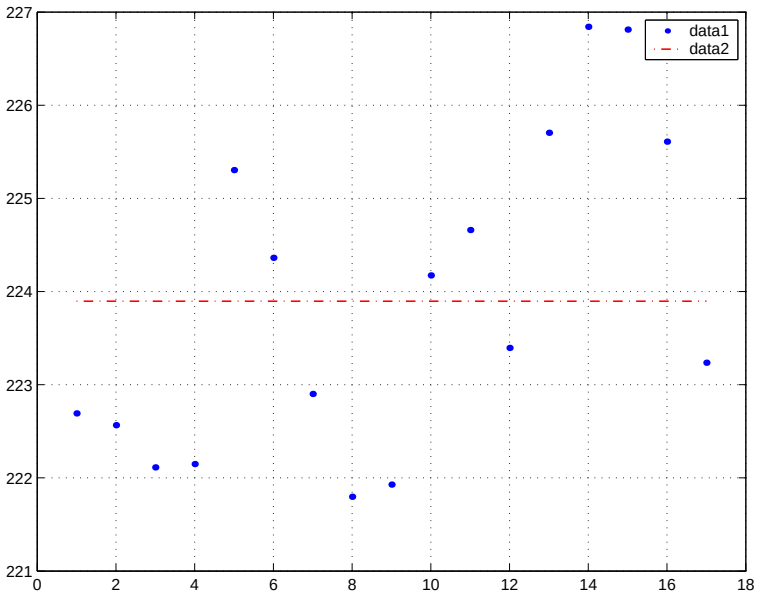


FIGURE 9.3: Mean of the 17 D_1/p values: 223.9; the range is: 221.8 to 226.8 and the standard deviation 1.6.

by the pixel. This pixel quantisation introduces an error, because the position used is an *integer* coordinate whereas the exact position of this intersection could be a fraction of a pixel away[2]. A few obvious irregularities due to pixel quantisation are illustrated in figure 9.4.

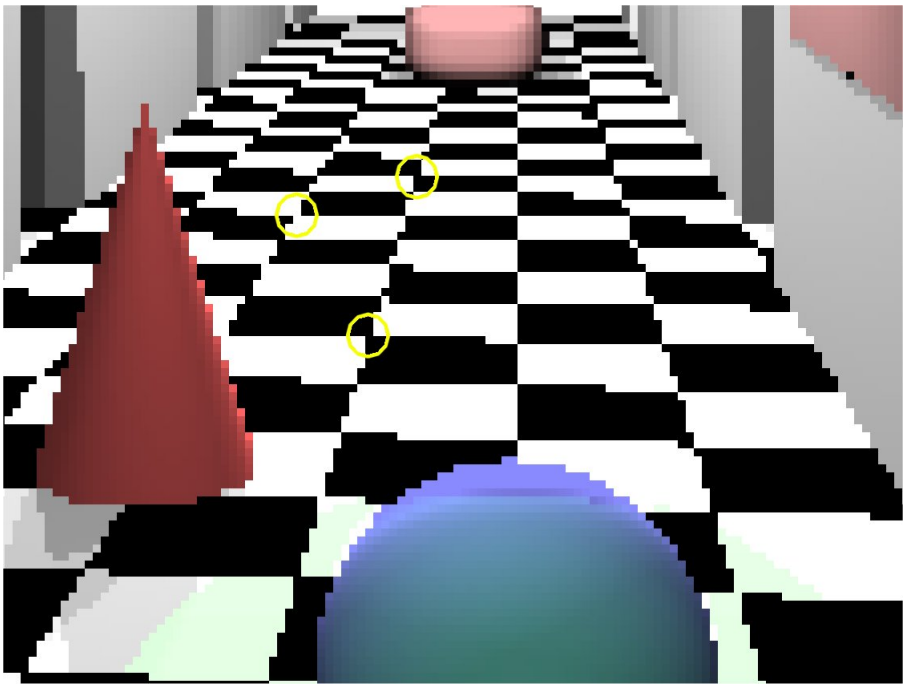


FIGURE 9.4: Examples of pixel quantisation problems seen after magnifying the floor area of the Corridor scene.

Table 9.2 gives the pixel world coordinates (from the scene description file in appendix B), left image pixel coordinates, calculated ground truth disparity and calculated D_1/p for the 17 points.

$$\frac{D_1}{p} = \frac{D_2 \cdot d}{b}$$

D_2 was obtained by projecting in the real world coordinates:

- x: across the corridor,
- y: depth and,
- z: upwards.

the vector:

[2]For all 17 pixels a consistent position - illustrated in the inset to figure 9.2 - was chosen: the lower left pixel of the intersection for 17 intersections at different depths.

- starting from the camera optical centre, $Cam(250, 150, 170)$ and
- pointing at the point at the tile's intersection, $P(x, y, z)$.

onto the camera optical axis:

- starting from the camera optical centre, $Cam(250, 150, 170)$ and
- pointing at the look point: $LP(100, 1600, 100)$.

The dot product of these two 3D vectors gives the cosine of the angle between them, this cosine is used to obtain the norm of the projected ray between the camera and the considered scene point onto the camera optical's axis:

$$D_2 = \|\overrightarrow{P(x,y,z) - Cam}\| \times \overrightarrow{P(x,y,z) - Cam} \cdot \overrightarrow{LP - Cam} \tag{9.1}$$

Point World Coordinates	Point Image Coordinates	Disparity			Calculated D_1/p		
		b=10	b=50	b=90	b=10	b=50	b=90
200 150 0	241 110	7.19	35.95	64.71	222.69	222.69	222.69
200 200 0	224 115	6.19	30.95	55.72	222.56	222.56	222.56
200 250 0	211 118	5.42	27.13	48.84	222.11	222.11	222.11
200 300 0	201 121	4.83	24.19	43.55	222.14	222.14	222.14
200 350 0	193 124	4.42	22.14	39.85	225.30	225.30	225.30
200 400 0	187 126	4.01	20.08	36.14	224.36	224.36	224.36
200 450 0	181 127	3.66	18.31	32.97	222.90	222.90	222.90
200 500 0	176 129	3.37	16.85	30.33	221.79	221.79	221.79
200 550 0	172 130	3.13	15.67	28.21	221.92	221.92	221.92
200 600 0	169 131	2.95	14.79	26.62	224.17	224.17	224.17
200 650 0	166 132	2.78	13.91	25.04	224.65	224.66	224.66
200 700 0	163 133	2.60	13.02	23.45	223.39	223.39	223.39
200 750 0	161 133	2.48	12.44	22.39	225.70	225.70	225.70
200 800 0	159 134	2.37	11.85	21.33	226.84	226.84	226.84
200 850 0	157 135	2.25	11.26	20.27	226.81	226.81	226.81
200 900 0	155 135	2.13	10.67	19.22	225.60	225.60	225.60
200 950 0	153 136	2.01	10.09	18.16	223.23	223.23	223.23

TABLE 9.2: Calculated $D1/p$ values for the 17 chosen points: world coordinates, pixel position on the left image, disparity and computed D_1/p for baselines (b =10, 50, and 90). Values vary slightly due the pixel quantisation problem but are constant for different baselines. An average of all these D_1/p values was used.

9.3.2 Experimental Accuracy Check

Rewriting equation 2.7 by dividing both the numerator and the denominator by p to use the D_1/p ratio leads to:

$$\delta D_2 = \frac{D_2^2}{b \cdot \dfrac{D_1}{p} - D_2} \tag{9.2}$$

In the Corridor scene description file, the sphere as a radius of 33 cm and is positionned 290 cm away from the camera's optical centre.

From the Corridor ground disparity map for a baseline of 50 cm (see figure 9.5 for the pixel positions):

- the closest point of the centre of the sphere has a disparity $d_{max} = 42.4\ pixels$ and
- the furthest point on the top has a disparity $d_{min} = 36.6\ pixels$,

i.e. 5.8 $pixels$ to describe a depth range corresponding to the radius of the ball (33 cm). Thus the accuracy at the point halfway between these two is approximately[3]:

$$\boxed{31.5 \div 5.8 = 5.4cm \cdot pixels^{-1}}$$

On the other hand, using equation 9.2 in this situation, with:

- $D_2 = 261.3 \pm 33cm$,
- $b = 50cm$,
- $D_1/p = 223.9$ from figure 9.3

gives:

$$\boxed{\delta D_2 = 6.2cm \cdot pixels^{-1}}$$

The three points chosen for checking are:

- the centre of the sphere and the top of the sphere,
- the middle of the base of the cone and the top of the cone and
- the two first tile intersections used in section 9.3.1.

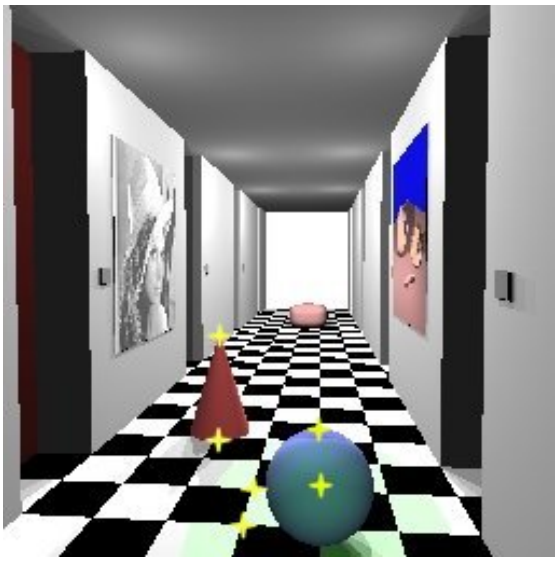


FIGURE 9.5: Positions on the sphere used for the accuracy check.

Point World Coordinates	Point Image Coordinates	b=10 Comp. δD_2	b=10 Th. δD_2	b=50 Comp. δD_2	b=50 Th. δD_2	b=90 Comp. δD_2	b=90 Th. δD_2
Sphere centre and sphere top							
250 107 33	196 145						
250 173 66	234 145	27.3	34.5	5.4	6.2	3.0	3.4
Cone base and cone top							
150 290 0	152 100						
150 320 100	201 100	103.9	115.0	20.7	19.1	11.5	10.4
First tile							
200 150 0	241 110						
200 200 0	224 115	49.8	49.7	13.0	8.8	7.2	4.8

TABLE 9.3: List of the 3 chosen accuracy checks in the Corridor scene.

Each of these was checked for three different values of the baseline: 10 *cm*, 50 *cm*, and 90 *cm*. Table 9.3 shows that computed depth accuracies are close to the theoretical ones. Discrepancies are found for some of the values, here again quantisation problem are an explanation. Also, this test is based on easy to spot - but quite far apart - points in the image: the discrepancies get bigger with the depth difference inherent to the object:

- cone: 30 *cm*,

[3]This is an approximation because the depth accuracy is a non-linear function of D_2.

- sphere: 33 cm and
- tile: 50 cm.

This is designed to get an easy comparison with the theoretical law. It uses a range to compare to a local, non-linear, law; also explaining existing discrepancies.

9.4 Summary

The results presented demonstrate that, even though increasing the baseline increases the disparity range and thus the accuracy obtainable through the whole scene, it also degrades drastically the matching quality of the algorithms so it cannot be simply used in order to improve the accuracy of a given scene.

The experiment also enabled a check of the validity of theoretical accuracy results given in chapter 2.

Chapter 10

Fossil Shell Measurement

10.1 Introduction

This experiment was motivated by a need to measure several dimensions on some plaster casts of fossil shells: it was a practical application of theory described in chapter 2. The position of the shell which would provide the best possible accuracy was determined following chapter 4. These casts were fragile and not easily measured by conventional means - which prompted the request to use photogrammetry to obtain the dimensions of interest - but this meant that a ground truth disparity map was not available.

Projecting patterned light onto the scene adds extra information and thus helps matching. Active illumination - described in section 3.2.6 - was assessed: using a colour pattern enabled comparisons of standard algorithms on greyscaled images as well as *combined* and *separated* versions of SAD and Corr2 - see chapter 8 - for colour processing.

10.2 Experiment Setup

The shell is about $20 \times 20\ mm^2$, $10mm$ high and the ridges on the surface of the shell are about $1mm$ deep, see figure 10.1.

Chapter 4 describes methods to adjust the camera parameters once the desired configuration has been determined. Knowing from section 4.2.2 that a reasonable D_1/p for Camera A is $\sim 2.0 \cdot 10^3$ - or $D_1 = 8mm$ for a pixel size $p = 4\mu m$ (see section 4.2.2) - and using chapter 2, it can be inferred:

- $b \simeq 20\ mm$ from equation 2.13,
- $D_2 \simeq 110\ mm$ from equation 2.11

FIGURE 10.1: Stereopair of the shell with a ruler for measuring. Original image size is 720×540 pixels.

- in this configuration the disparity range is about 70 pixels to describe a shell depth of almost 10mm, *i.e.* $\delta D_2 \simeq 0.12mm$.

Two sets of images were used, differing in the illumination scheme used:

- Set A: using ambient light (sunlight and fluorescent room lighting) - see figure 10.3 top - and
- Set B: using a colour mask - see figure 10.2 - projected onto the scene - see figure 10.3 bottom.

Note that the second scheme was designed to overcome the lack of texture in the white plaster cast - see the active illumination discussion in section 3.2.6 - using the repetitive colour pattern illustrated in figure 10.2, following Kanade's proposition that using colour would further improve active illumination [92]. The pattern was printed on a transparency and projected onto the shell.

For this chapter:

- $SAD(r = 4)$, $SAD(r = 10)$, and
- Pixel-to-Pixel($\kappa_{occ} = 5, \kappa_r = 40$)

were used as previous trials had shown them to be reliable and producing the best results of this whole study. Also

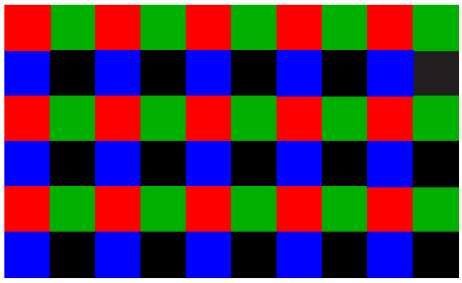


FIGURE 10.2: Colour pattern used for the active colour illumination experiment.

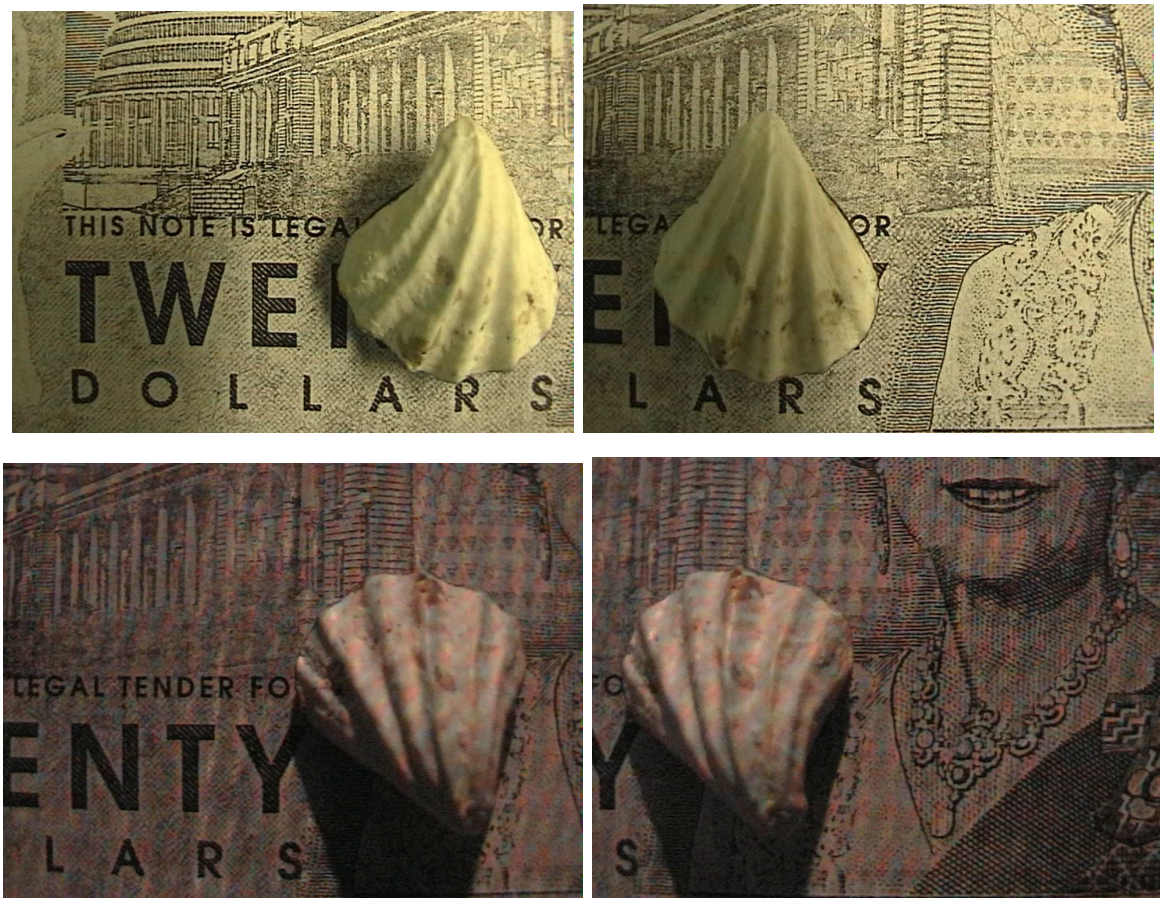


FIGURE 10.3:
Top: Set A, left and right images using ambient light,
Bottom: Set B, left and right images using a colour pattern projected onto the scene.

- $Corr2(r = 4)$ and $Corr2(r = 10)$

were assessed because of Corr2's better behaviour with luminosity changes - compared to Corr1 and SAD - see figure 3.3 in section 3.2.1.

To eliminate irrelevant background regions, all results were filtered using the binary masks in figure 10.4.

FIGURE 10.4:
Left: binary mask for set A (ambient light),
Right: binary mask for set B (active colour illumination).

Note 1: No sub-pixel estimation technique was used so that computed disparities were constrained to integers which adds to the apparent irregularities of the result.

Note 2: Birchfield and Tomasi use post processing steps with Pixel-to-Pixel to reduce errors. No post processing was used here as it can be used to improve **any** processed disparity map and thus could bias the evaluation of core algorithms.

10.3 Experiment Results

Results are presented:

- firstly for set A using the standard versions - *i.e.* converting the images to greyscale before processing them - of the algorithms, denoted as *'greyscale processing'* in the following of this chapter,
- secondly for set B as an illustration of active colour illumination used to assist greyscale processing algorithms, and
- finally for set B using the modified correlation algorithms (SAD and Corr2) presented in chapter 8, denoted *'colour processing'* in the remainder of this chapter.

Note 1: The colour experiments in chapter 8 showed no significant improvement with either *combined* or *separated* use of the colour bands. However, the hypothesis for the current experiments was that these techniques might assist matching with active colour illumination of an otherwise weakly textured scene.

Note 2: A standard layout was used for all the the following experimental figures:

- on the left: the disparity map in *pixels*,
- on the right: the reconstructed model - see section 4.5 - for which the depth is given in m,
  - for the ambient light image set - section 10.3.1 - the reconstructed model is directly derived from its corresponding disparity map, because the matching quality is too low and filtering did not improve the view,
  - for the active illuminated image set - section **??** - the reconstructed model used a filtered disparity map which eliminates physically impossible disparities from the reconstructed view.

10.3.1 Set A: Ambient light using Greyscale Processing

In this section, all images were converted to greyscale, *i.e.* luminosity, before computing the disparity maps. Figures 10.5, 10.6, 10.7, 10.8, and 10.9 show the computed disparity map and the 3D reconstructed object using synthetic colours - see section 4.5. Parameters used were:

- $SAD(r = 4)$, $SAD(r = 10)$,
- $Corr2(r = 4)$, $Corr2(r = 10)$ and
- Pixel-to-Pixel($\kappa_{occ} = 5, \kappa_r = 40$).

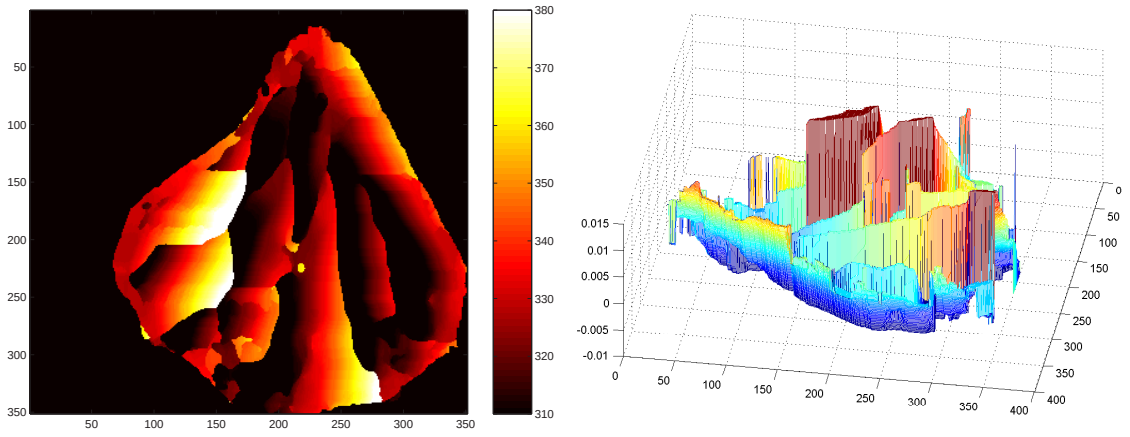


FIGURE 10.5: Left: disparity map, Right: 3D reconstruction using synthetic colours.
Algorithm: $SAD(r = 4)$
Images: ambient light shell set.

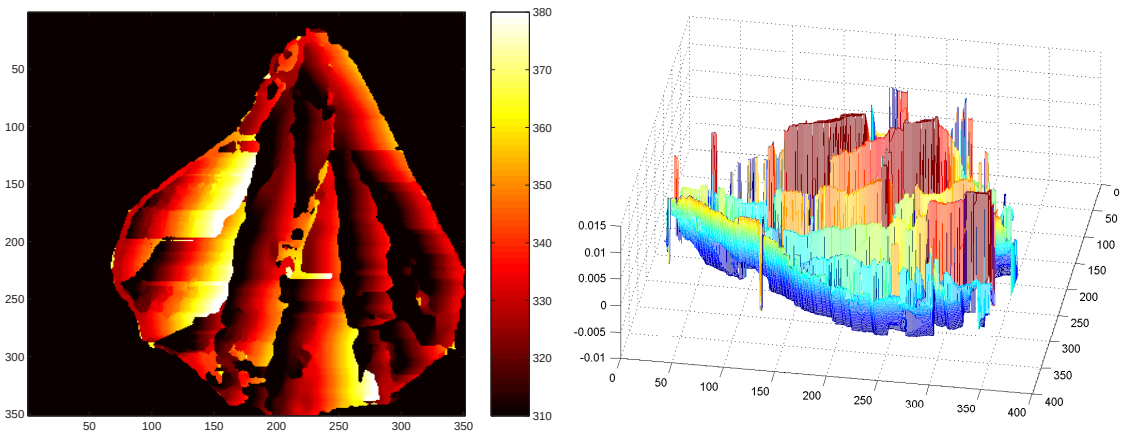


FIGURE 10.6: Left: disparity map, Right: 3D reconstruction using synthetic colours.
Algorithm: $SAD(r = 10)$
Images: ambient light shell set.

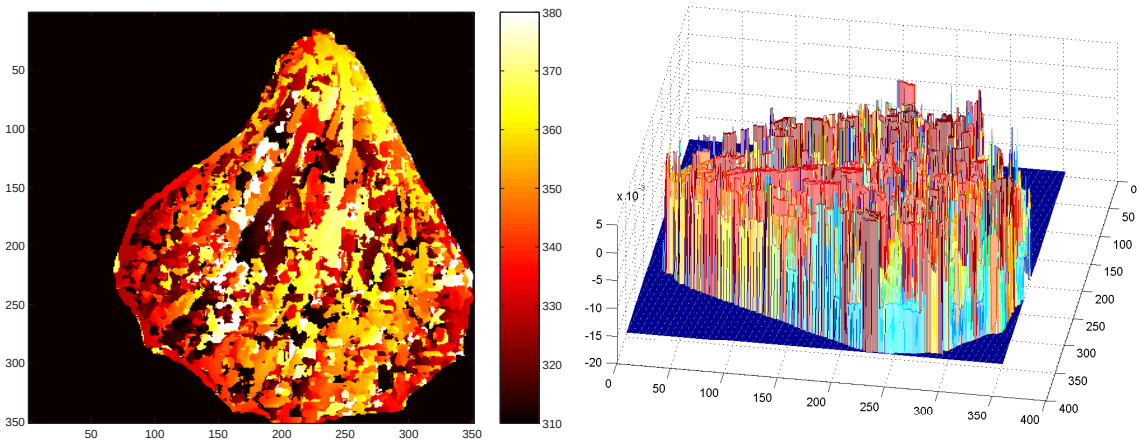


FIGURE 10.7: Left: disparity map, Right: 3D reconstruction using synthetic colours.
Algorithm: $Corr2(r = 4)$
Images: ambient light shell set.

All these results show a very poor reconstruction with no post-processing applied to them: the ridges are wrongly matched because of their repetitive nature leading to a completely false 3D map of the shell for all algorithms. As illustrated by figure 9.1,in section 9.2 of the experiment dealing with the baseline modification impacts: increasing the baseline does significantly handicap matching algorithms. Here the disparity range is about 70 pixels and contributes to the poor matching results.

In the light of these results, active colour illumination of the scene was tried. In the following section - see 10.3.2 - both greyscale and colour processing algorithms have been compared on set B stereopair.

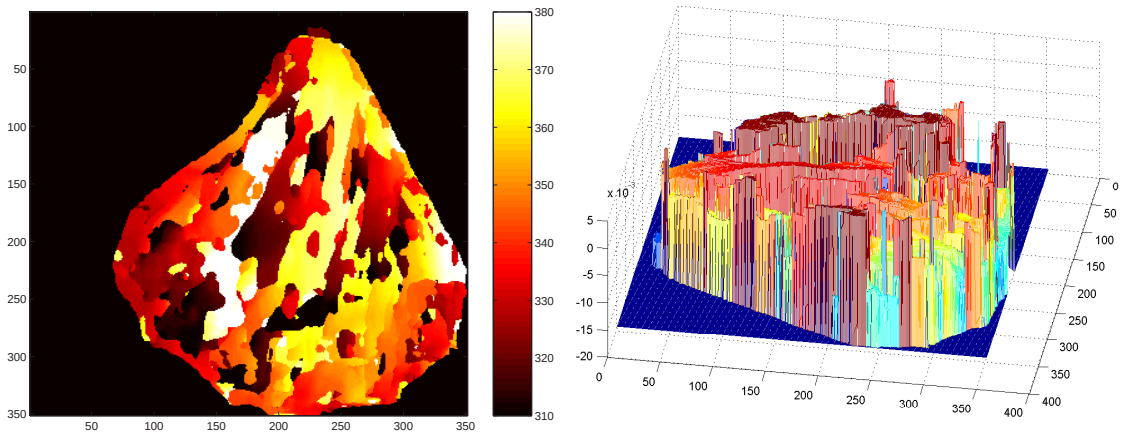


FIGURE 10.8: Left: disparity map, Right: 3D reconstruction using synthetic colours.
Algorithm: $Corr2(r = 10)$
Images: ambient light shell set.

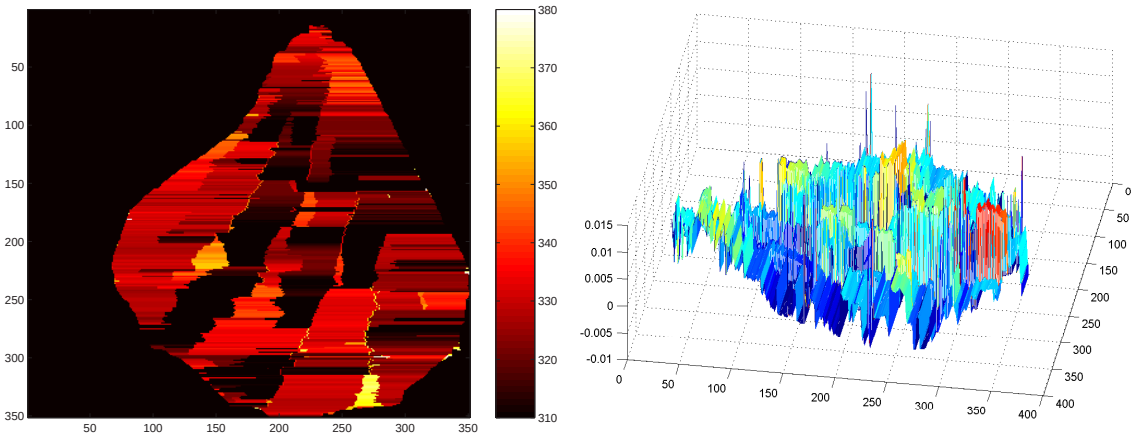


FIGURE 10.9: Left: disparity map, Right: 3D reconstruction using synthetic colours.
Algorithm: Pixel-to-Pixel($\kappa_{occ} = 5, \kappa_r = 40$)
Images: ambient light shell set.

10.3.2 Set B: Active Colour Illuminated Scene

Greyscale Processing

Figures 10.10, 10.11, 10.12, 10.13 and 10.14 show computed disparity maps and reconstructed objects for:

- $SAD(r = 4)$, $SAD(r = 10)$,
- $Corr2(r = 4)$, $Corr2(r = 10)$ and
- Pixel-to-Pixel($\kappa_{occ} = 5, \kappa_r = 40$).

Note that the disparity maps show all computed disparities, but, for the reconstructions, disparity values used have been filtered using bounds derived from approximate measurements of the shell depth.

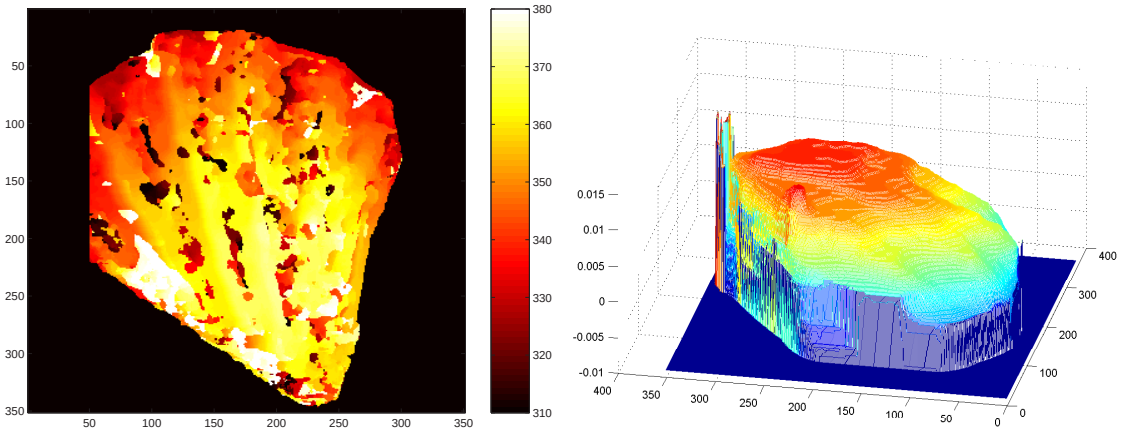


FIGURE 10.10: Left: disparity map, Right: 3D reconstruction using synthetic colours.
Algorithm: $SAD(r = 4)$
Images: active colour illumination shell set.

Using the greyscale processing algorithms on the active colour illuminated scene does significantly improve matching with the SAD and Corr2 algorithms - especially with a window radius of 10 - but Pixel-to-Pixel still fails on all the ridges and does not give any intelligible results.

With a window radius of 4, SAD and Corr2 provide recognisable shell shapes, and only small parts of the shell are not matched properly with a window radius of 10. The two main areas of incorrect matching with SAD and Corr2 are on the bottom left side of the shell which is mainly shaded in the original images.

Even with active colour illumination, Pixel-to-Pixel failed to provide any meaningful matching. The disparity range for these experiments is 70 pixels for both set

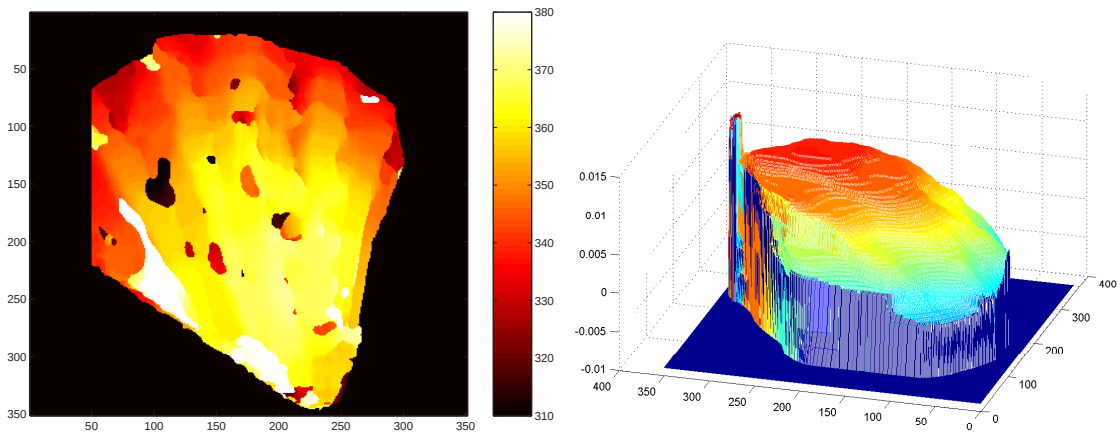


FIGURE 10.11: Left: disparity map, Right: 3D reconstruction using synthetic colours.
Algorithm: $SAD(r = 10)$
Images: active colour illumination shell set.

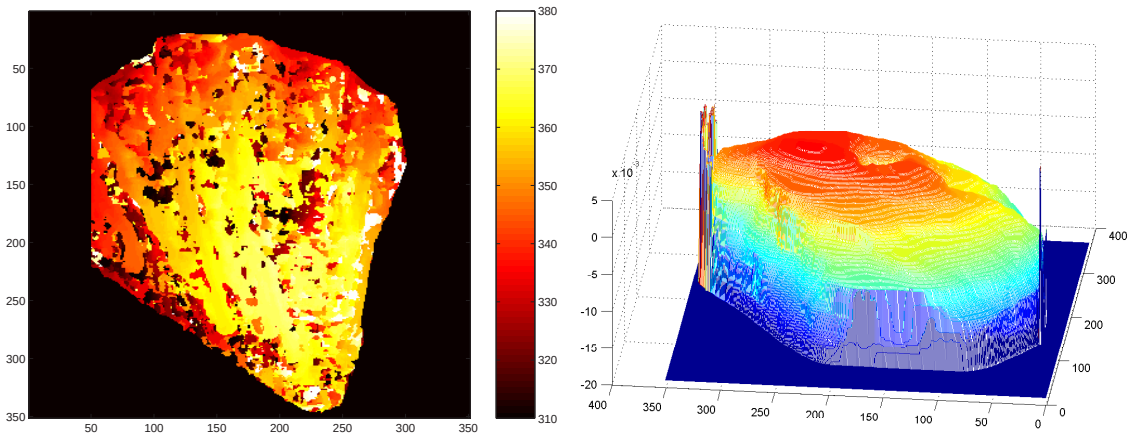


FIGURE 10.12: Left: disparity map, Right: 3D reconstruction using synthetic colours.
Algorithm: $Corr2(r = 4)$
Images: active colour illumination shell set.

A and B and is much larger than those in the test images (from 14 pixels for the Tsukuba pair to 29 for the Map pair). This effect is illustrated in figure 9.1 and further discussed in chapter 9.

In addition, the repeated patterns of the ridges on the shell have some of the characteristics of the Madroom set - **all** algorithms had difficulties matching repetitive patterns properly.

Internally Pixel-to-Pixel uses a threshold of 5 intensity levels - on 8 bit images, *i.e.* a range of 255 intensity levels - over 3 consecutive pixels to declare a gradient. The

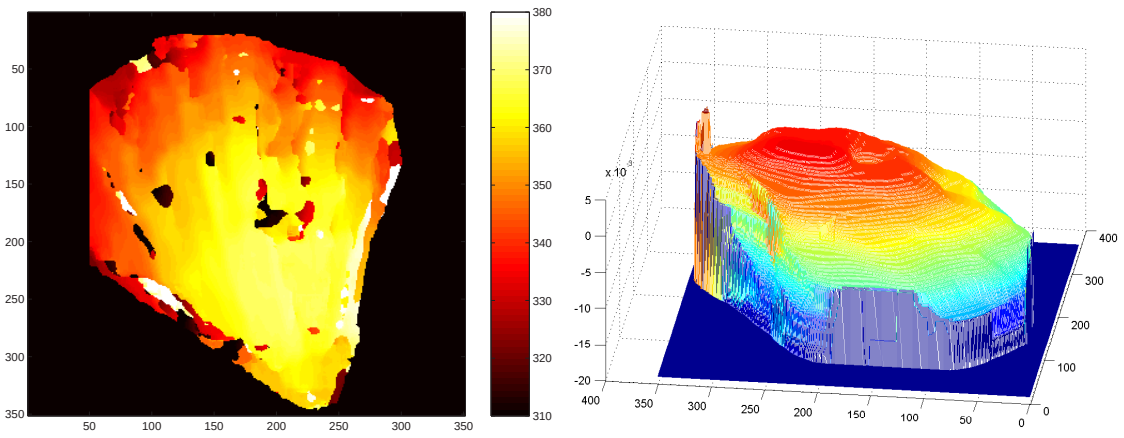


FIGURE 10.13: Left: disparity map, Right: 3D reconstruction using synthetic colours.
Algorithm: $Corr2(r = 10)$
Images: active colour illumination shell set.

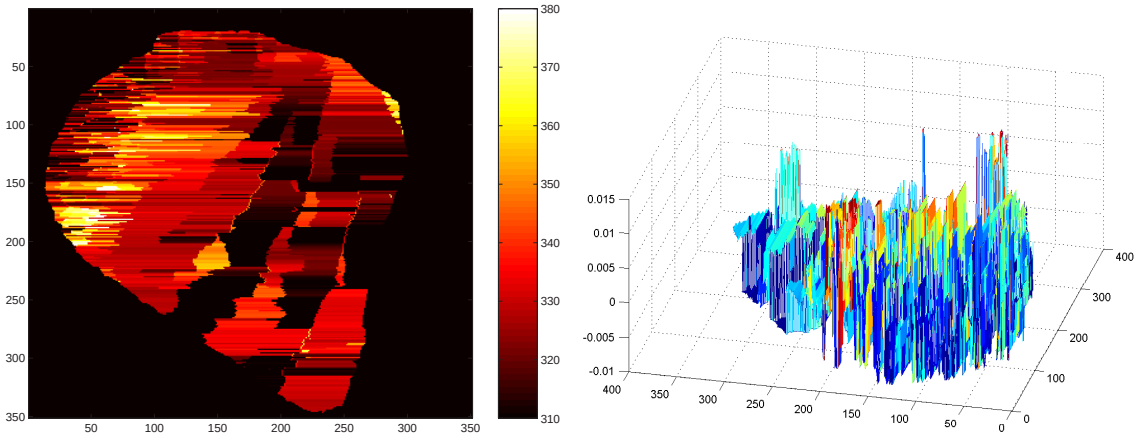


FIGURE 10.14: Left: disparity map, Right: 3D reconstruction (non-filtered because of the low-quality of the matching) using synthetic colours.
Algorithm: Pixel-to-Pixel($\kappa_{occ} = 5, \kappa_r = 40$)
Images: active colour illumination set.

gradients are part of Pixel-to-Pixelś occlusion finding strategy - see section 3.2.3. This scene of a shell seen from the top is expected to have very few occlusions. However, figure 10.15 illustrates Pixel-to-Pixelś computed occlusion maps both for set A and B.

To an extent, active colour illumination provides better results even though Pixel-to-Pixel still does not really perform matching properly:

- for set A: the background - being black and white - is almost entirely interpreted as occlusions whereas it is just a flat back ground, and

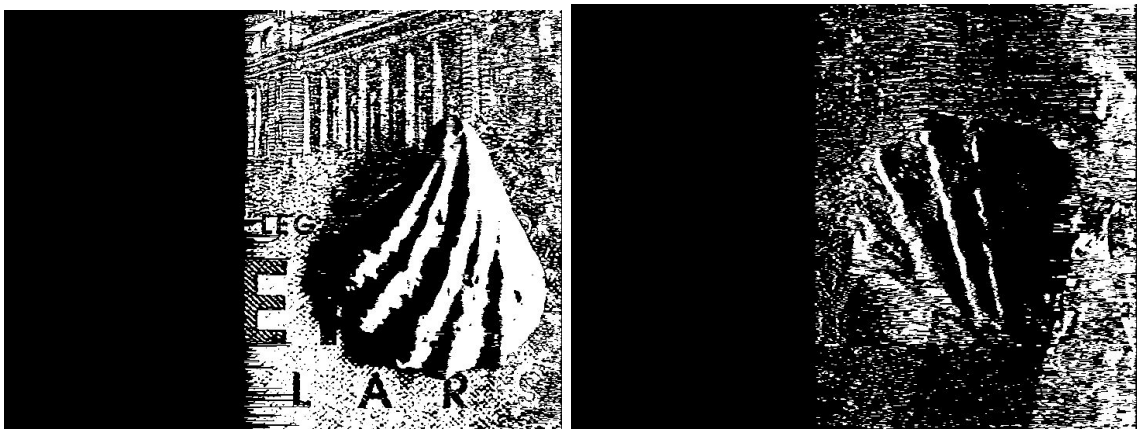


FIGURE 10.15: Occlusion maps using Pixel-to-Pixel($\kappa_{occ} = 5, \kappa_r = 40$) with its default threshold for the greyscale gradient.
Left: set A, Right: set B.

- for set B: fewer occlusions are found and are much closer to reasonable ones.

The two other parameters for Pixel-to-Pixel guide the penalty for an occlusion and the reward for a match. Figure 10.16 shows the disparity map and reconstructed 3D model for set B, where:

- $\kappa_{occ} = 150$: the penalty was increased as this scene should have is not contain any significant occlusions,
- $\kappa_r = 40$: following the same strategy, the reward for a match was increased, and
- $Grey_{threshold} = 10$: the threshold for the greyscale gradient was also increased.

This choice of Pixel-to-Pixel parameters produces a much more realistic reconstruction. This shows that Pixel-to-Pixel can perform accurately in different circumstances but that it is very sensitive to its parameters: without appropriate parameters it can fail completely. So, for general use, it appears that Pixel-to-Pixel would need an a priori knowledge of the type of scene to dictate the choice of the three parameters: the penalty for an occlusion, the reward for a match and the threshold for greyscale gradients. Here, the best parameters - ($\kappa_{occ} = 5, \kappa_r = 40$) - deduced from the robustness to noise experiment - see chapter 7 using a precise ground truth - lead to a complete failure and other values have to be derived to suit the new conditions.

Note that it is probably possible to further increase the quality of matching for Pixel-to-Pixel on this specific stereo pair. However, without a ground truth, it is not possible to assess *objectively* which parameters lead to the best results.

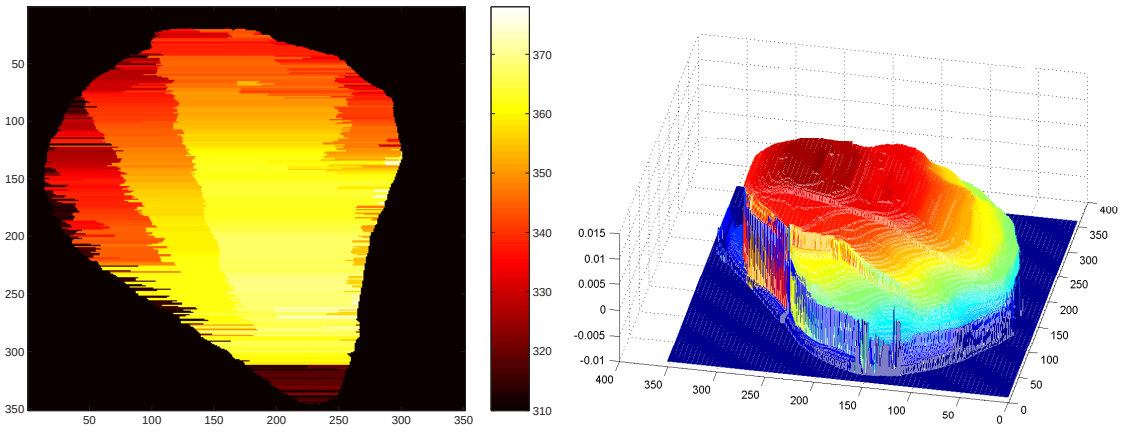


FIGURE 10.16: Left: Disparity map, Right: Filtered 3D reconstruction. Algorithm: Pixel-to-Pixel($\kappa_{occ} = 150, \kappa_r = 40$) threshold for the greyscale gradient = 10. Images: set B.

Colour Processing,* Combined *SAD and Corr2

Figures 10.17, 10.18, 10.19 and 10.20 show computed disparity maps and reconstructed objects for:

- $SAD_{Combined}(r = 4)$, $SAD_{Combined}(r = 10)$,
- $Corr2_{Combined}(r = 4)$ and $Corr2_{Combined}(r = 10)$ respectively.

As before, the disparity maps show all computed points but the reconstructed objects have been filtered.

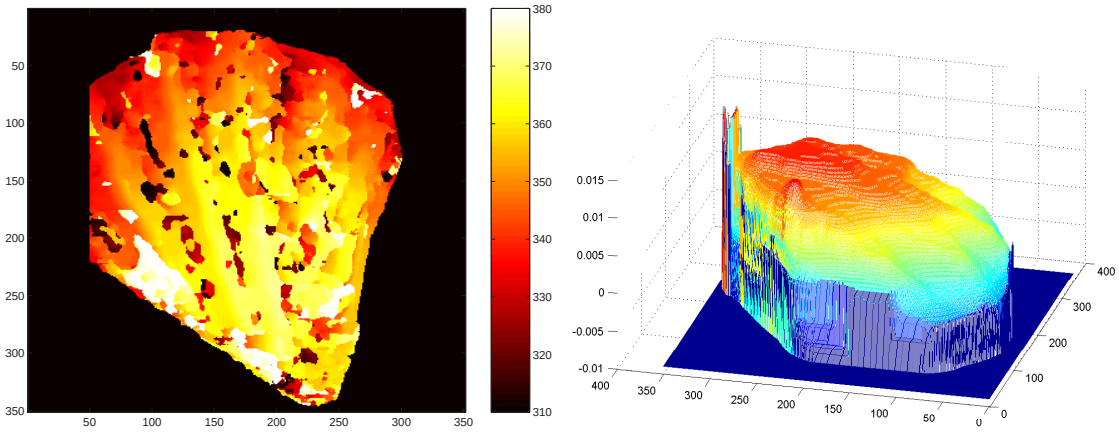


FIGURE 10.17: Left: disparity map, Right: 3D reconstruction using synthetic colours.
Algorithm: $SAD_{Combined}(r = 4)$
Images: active colour illumination shell set.

As already shown in chapter 8, the *combined* variants of SAD and Corr2 do not produce any significant improvement in matching. The lack of a ground truth makes this assessment difficult. However, carefully comparing the disparity maps (and 3D reconstruction) on figures 10.10, 10.11 on the one hand and figures 10.17, 10.18 - or 10.19, 10.20 for the Corr2 algorithm - on the other shows very similar error patches.

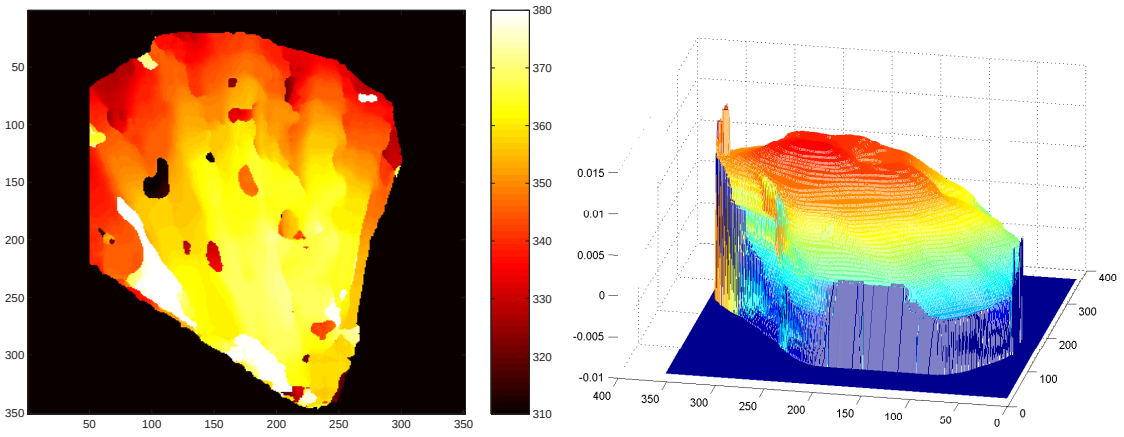


FIGURE 10.18: Left: disparity map, Right: 3D reconstruction using synthetic colours.
Algorithm: $SAD_{Combined}(r = 10)$
Images: active colour illumination shell set.

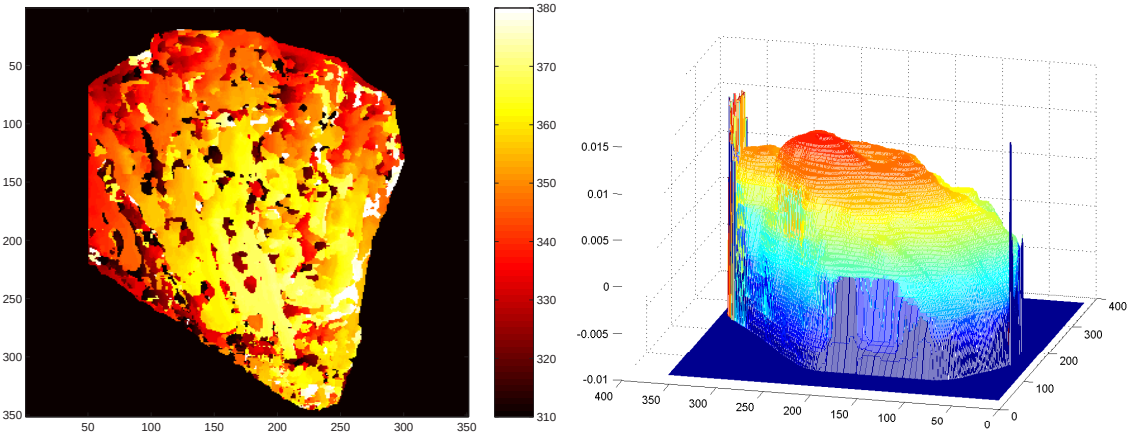


FIGURE 10.19: Left: disparity map, Right: 3D reconstruction using synthetic colours.
Algorithm: $Corr2_{Combined}(r = 4)$
Images: active colour illumination shell set.

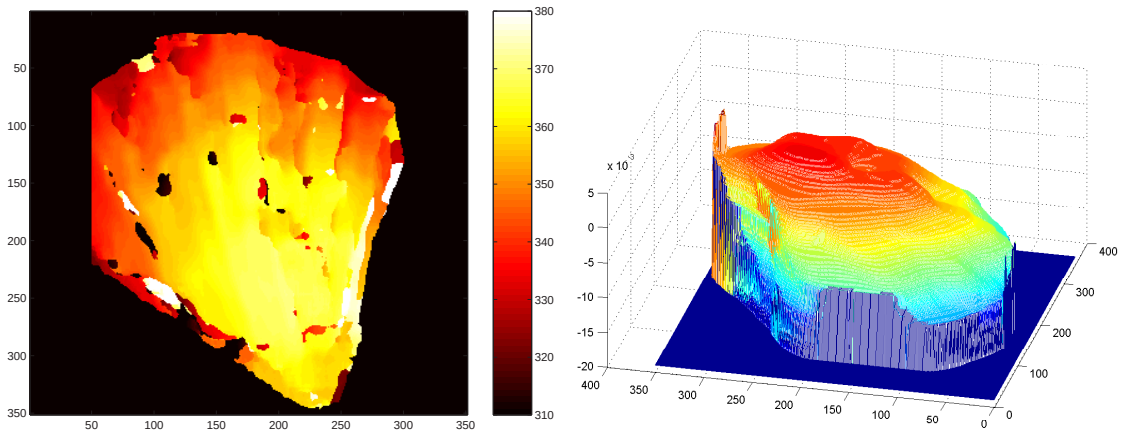


FIGURE 10.20: Left: disparity map, Right: 3D reconstruction using synthetic colours.
Algorithm: $Corr2_{Combined}(r = 10)$
Images: active colour illumination shell set.

Colour Processing,* Separated *SAD

Figures 10.21, 10.22, 10.23 and 10.24 computed disparity maps and reconstructed objects for:

- $SAD_{Separated}(r = 4)$, $SAD_{Separated}(r = 10)$,
- $Corr2_{Separated}(r = 4)$ and $Corr2_{Separated}(r = 10)$.

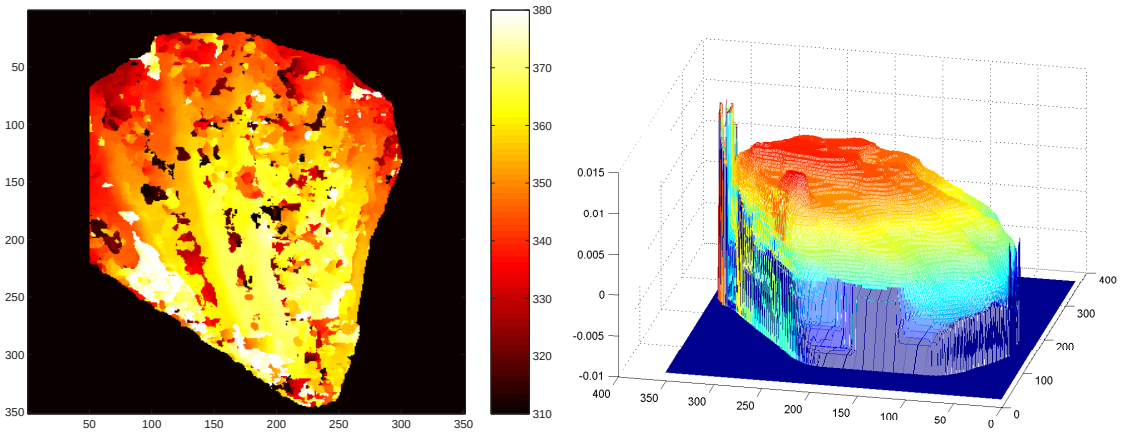


FIGURE 10.21: Left: disparity map, Right: 3D reconstruction using synthetic colours.
Algorithm: $SAD_{Separated}(r = 4)$
Images: active colour illumination shell set.

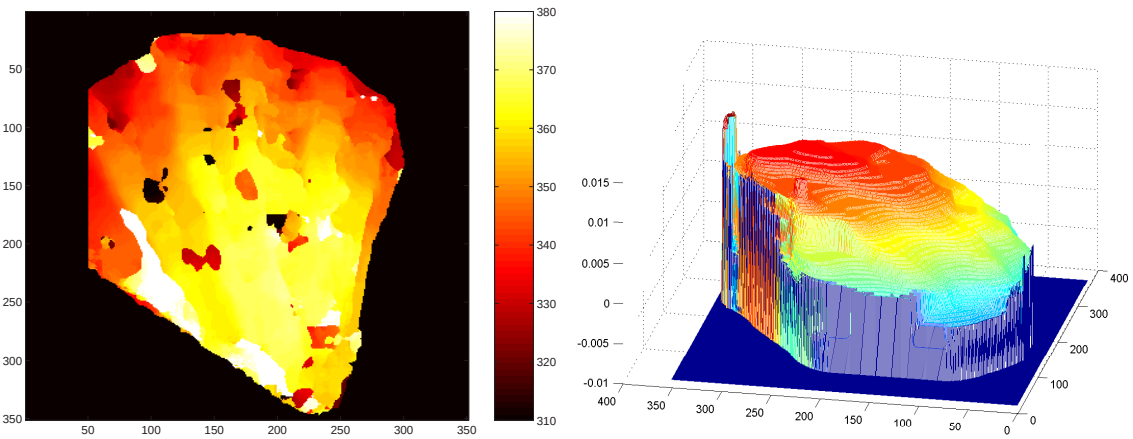


FIGURE 10.22: Left: disparity map, Right: 3D reconstruction using synthetic colours.
Algorithm: $SAD_{Separated}(r = 10)$
Images: active colour illumination shell set.

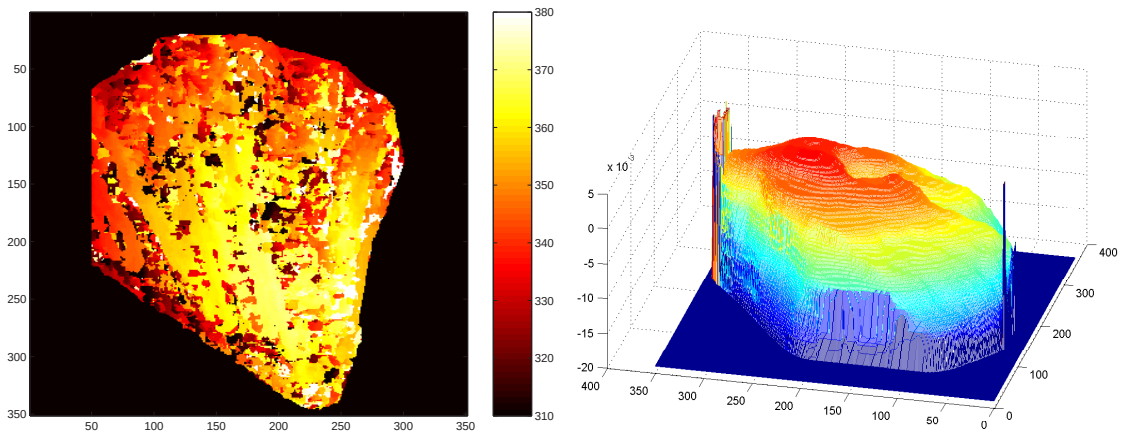


FIGURE 10.23: Left: disparity map, Right: 3D reconstruction using synthetic colours.
Algorithm: $Corr2_{Separated}(r = 4)$
Images: active colour illumination shell set.

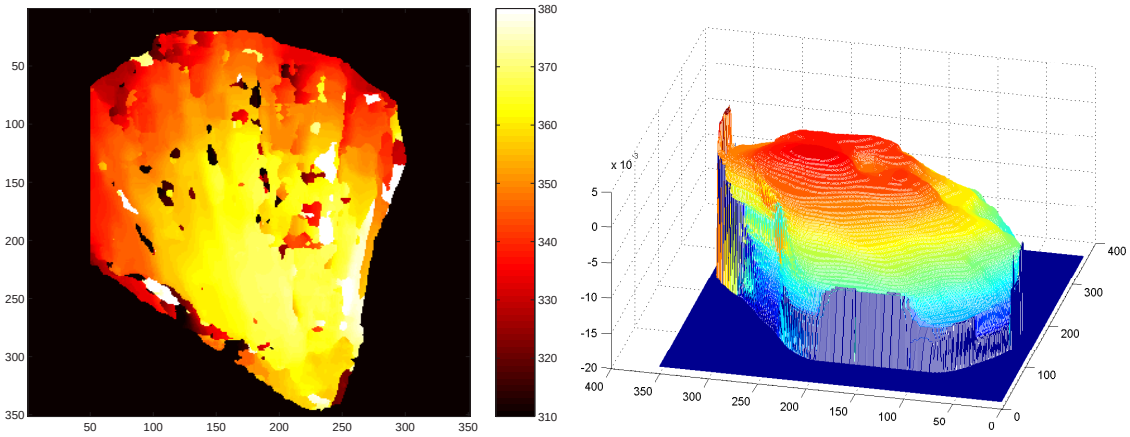


FIGURE 10.24: Left: disparity map, Right: 3D reconstruction using synthetic colours.
Algorithm: $Corr2_{Separated}(r = 10)$
Images: active colour illumination shell set.

The *separated* variant shows the similar behaviour to the *combined* one, *i.e.* no clearly better results are seen by comparing the disparity maps and 3D reconstruction on figures 10.10, 10.11 on the one hand and figures 10.21, 10.22 - or 10.23, 10.24 for the Corr2 algorithm - on the other.

10.4 Summary

This chapter describes a practical photogrammetric experiment based on both the geometry described in chapter 2 and the experiment setup procedures set out in chapter 4.

The shell was measured to have a height of about $10mm$. From the experiments giving useful results, *i.e.*figures 10.10 to 10.13 and 10.16 to 10.24, the height in the reconstructed model is also about $10mm$.

This experiment does not have a ground truth, so the comparisons are based on both the processed disparity maps as well as reconstructed 3D models and rely on one's ability to compare these with an expected sea shell shape: it further emphasizes the need for ground truth, *cf.* section 3.4, as comparing different results is necessarily somewhat subjective when relying on the appearance of the reconstructed models.

However, the greyscale processing algorithms on an ambient light illuminated object clearly do not produce the expected shape of the shell. In contrast, active colour illumination provides significantly better results. The colour patterns provide added matching information - distinguishing many regions that would otherwise be confused.

As in the experiments of chapter 8, the *combined* and *separated* colour processing variants of the correlation algorithms - SAD and Corr2 - did not lead to significantly better results than their greyscale versions.

Chapter 7 already illustrated the sensitivity of Pixel-to-Pixel to the choice of its parameters: the best parameters for the tested sets were $(\kappa_{occ} = 5, \kappa_r = 6)$, but $(\kappa_{occ} = 5, \kappa_r = 40)$ was significantly more stable against increasing levels of noise. Here again, in another different situation, $(\kappa_{occ} = 5, \kappa_r = 40)$ had to be changed to $(\kappa_{occ} = 150, \kappa_r = 40)$ to obtain useful results. It is thus clear that effective use of Pixel-to-Pixel requires a priori knowledge of the scene to guide a choice of reasonable values for its parameters.

Chapter 11

IGN Aerial Stereo Pair

11.1 Introduction

This chapter presents the results of experiments on the aerial image set described in section 3.5.11. The images were taken from the air at two different times: the lower right shows two cars and pedestrians around the church which have moved. The photos were provided by the MATIS laboratory of the French National Geographic Institute[1] and are rectified *i.e.* epipolar. Taken from a plane, this set does not have the parallel camera axes configuration of the other sets used in this study. Its crossing axis geometry is discussed in section 2.3, where the presence of both positive and negative disparities is explained.

Note that two disparity maps were supplied with these images: a manual entry one used as ground truth because of its accuracy - see figure 11.7 bottom - and one obtained by laser.

Laser-range scanning has been performed by TopoSys GmbH [99]. The sensor [100] is linear and made of 127 optic fibers scanning in different directions every $\pm 14.3^o$ orthogonally to the flight axis. The number of measured points per square meter changes with the plane's speed and altitude. The laser generated disparity map provided by the MATIS laboratory has about one point every 100mm in the flight direction and one point every 1.20m in the orthogonal direction. Thus this device measures irregular clouds of points as a function of the depth of the observed scene. To obtain a dense model, these measures are interpolated, which introduces some defaults especially close to the buildings. The point measure accuracy is about 100mm, but because of their irregular distribution as well as the interpolation step it is difficult to give a precision for the dense model.

[1] Institut Géographique National: IGN

The manual entry disparity map was produced end of the 70's, early 80's by the IGN and is much older than the laser generated one. Buildings might have been modified since then, however choosing a subset close to the church makes it easier to check and the main part of the image has not been modified. Being matched by hand using stereo-restitution this disparity map has been chosen as our groundtruth, it's precision is about 100mm.

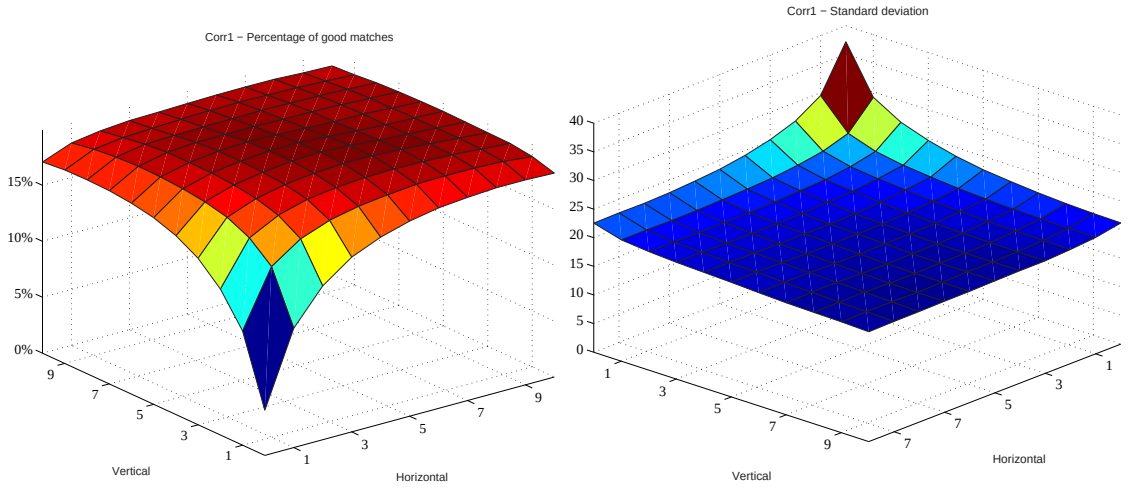
Having these two disparity maps enabled to compare automated binocular stereovision with a completely different method - *i.e.* laser range scanning - for obtaining depth information.

11.2 Results

Figures 11.1 to 11.5 show results using:

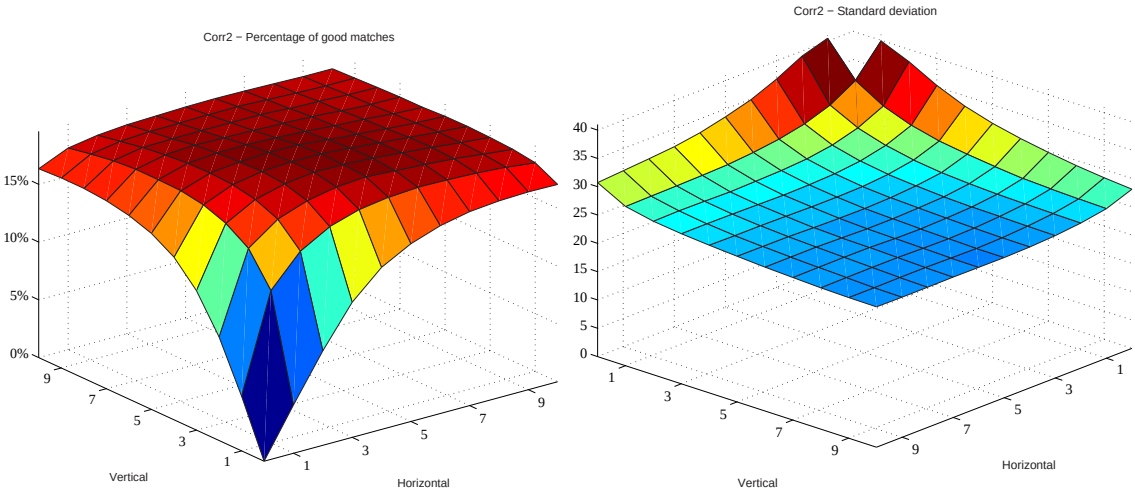
- Corr1, parameters: correlating window height and width,
- Corr2, parameters: correlating window height and width,
- SAD, parameters: correlating window height and width,
- Census, parameters: α and β, and
- Pixel-to-Pixel, parameters: κ_{occ} and κ_r.

on an 800×800 pixel region of the full images: and the working window enables a comparison on 274360 pixels.



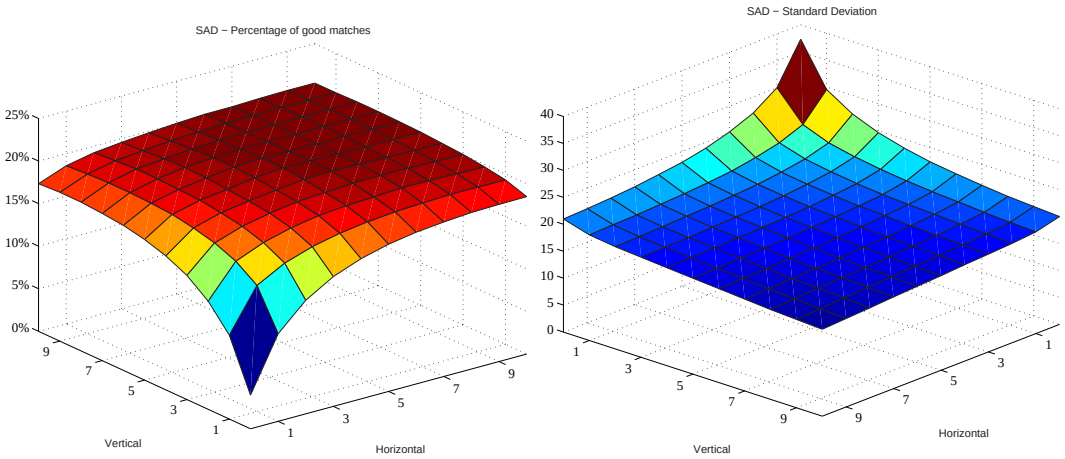
Percentage of good matches - Standard deviation

FIGURE 11.1: Algorithm: Corr1



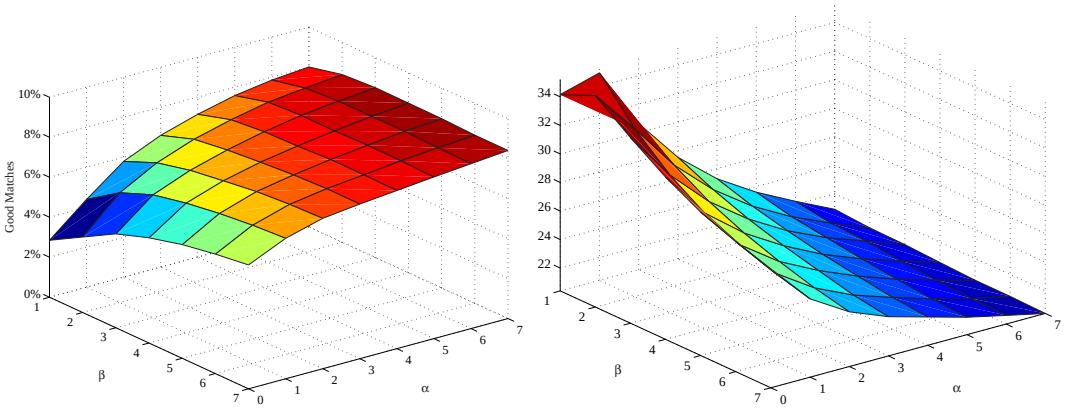
Percentage of good matches - Standard deviation

FIGURE 11.2: Algorithm: Corr2



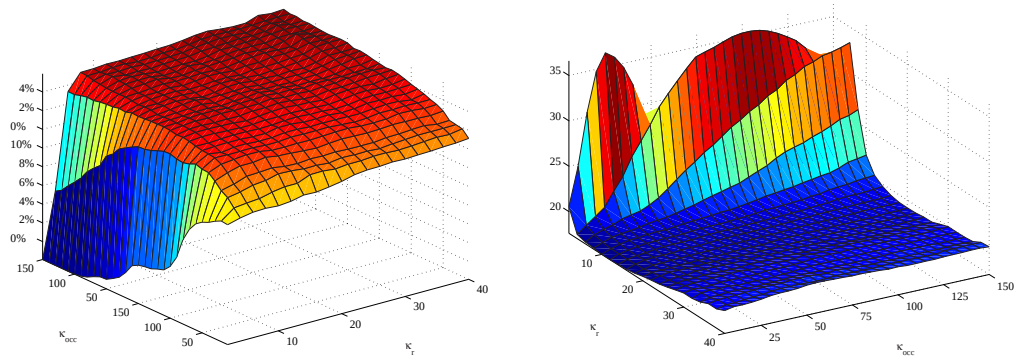
Percentage of good matches - Standard deviation

FIGURE 11.3: Algorithm: SAD



Percentage of good matches - Standard deviation

FIGURE 11.4: Algorithm: Census



Percentage of good matches - Standard deviation

FIGURE 11.5: Algorithm: Pixel-to-Pixel

11.2.1 Percentage of Good Matches

This aerial stereopair has the largest disparity range of this study:

- $-42 \sim +87 = 130$ pixels *vs.*
- $14 \simeq 29$ for Corridor, Madroom, Map, Sawtooth, Tsukuba and Venus - see section 3.5 -
- and 70 for the fossil shell experiment - see chapter 10.

Chapter 9 - figure 9.1 - illustrated the increasing difficulty of matching as the baseline increased, *i.e.* increasing the disparity range. This was mainly attributed to:

- geometry: the larger the baseline the less alike the two images become - increasing number and size of occlusions - and
- statistics: the wider matching range increases the chances to chose an incorrect match.

This was also observed in the fossil shell experiment - see chapter 10 - where similar matching difficulties were seen, which were partially alleviated when extra information was added: see the description of active colour illumination in section 3.2.6 and results in section 10.3.2.

The results generally show a much lower percentage of good matches: table 11.1 compares the good match percentages for standard values of the algorithms parameters for the aerial *vs.* the Corridor stereopair.

	Good Matches (%)	
Algorithm	Aerial stereopair	Corridor
$Corr1(r = 4)$	19.8	62.9
$Corr2(r = 4)$	19.6	57.7
$SAD(r = 4)$	20.0	65.4
$Census(\alpha = 4, \beta = 3)$	7.8	64.2
$Pixel - to - Pixel(\kappa_{occ} = 5, \kappa_r = 40)$	7.6	63.6

TABLE 11.1: Good matches for the aerial *vs.* the Corridor stereopair for algorithms using the indicated parameter values.

11.2.2 Ordering Constraint: Church Tower

A typical problem of stereopairs taken with a large baseline is illustrated in figure 11.6. Aerial images are taken from about 1000m and need a large enough baseline to obtain a satisfying disparity range. The figure shows the the region around church tower on the left and right images and illustrates the two main problems of this situation:

- the images show different faces of the tower and
- neighbouring regions are completely different.

This illustrates an exception to the *ordering constraint* - *i.e.* a point on one side of an object in one image is on the same side of this object in the second image, see section 3.2 - assumed by all algorithms used in this thesis.

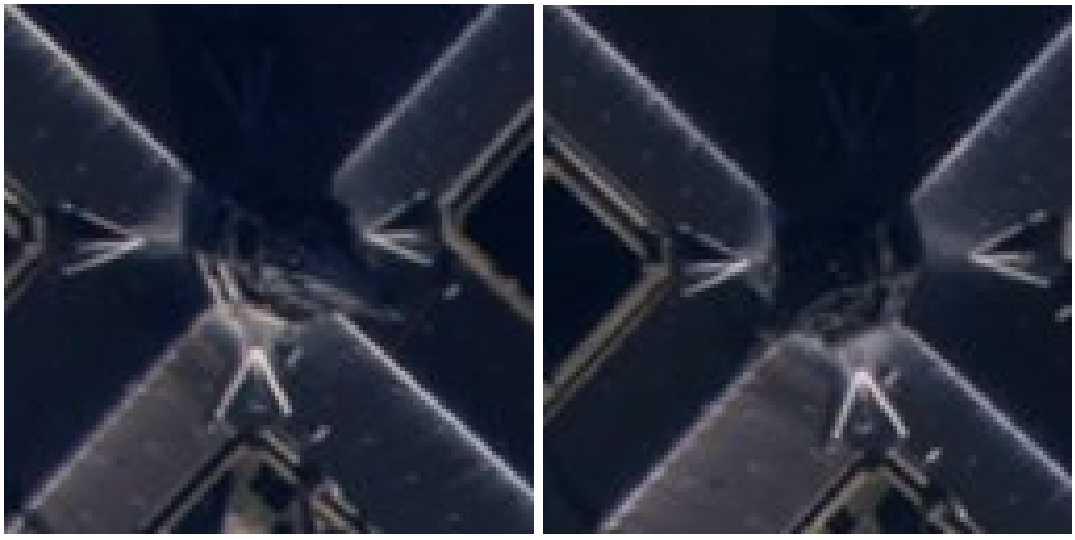


FIGURE 11.6: Magnified section of IGN images showing the region around the church tower.
Note that the tower shows different faces in the two images. Also a region on one side of the tower in one image is on the other side of it on the other image.

11.2.3 Laser Generated and Manual Entry Disparity Maps Comparison

Comparing the two disparity maps in figure 11.7, it is notable that the laser generated map gives blurred edges whereas the manual entry map gives accurate data to

use as ground truth. Figure 11.7 shows the error distribution for the laser generated disparity map compared to the manual entry one, it is characterized by:

- 33.2% of good matches,
- a standard deviation of 5.99 and
- an error range from -63 to +68 pixels.

Compared to automated stereo matching algorithms studied here, laser range scanner imaging gives a higher percentage of good matches, a narrower distribution and a smaller error range. However, this study's scope was biased towards algorithms with a potential for real-time and hardware implementation, thus other more recent, but much more complex algorithms, like Zabih's graph cut variants [34] should also be compared before generalizing this conclusion to all automated matching algorithms.

Algorithm	Good Matches (%)	Std. Dev.	Range
$Corr1(r=4)$	19.8	20.7	$-106 \sim +122$
$Corr1(r=10)$	18.7	19.2	$-73 \sim +122$
$Corr2(r=4)$	19.6	26.0	$-119 \sim +123$
$Corr2(r=10)$	18.1	24.8	$-89 \sim +122$
$SAD(r=4)$	20.0	19.3	$-101 \sim +122$
$SAD(r=10)$	20.4	16.1	$-74 \sim +120$
$Census(\alpha=4,\beta=3)$	7.8	23.9	$-98 \sim +122$
$Census(\alpha=7,\beta=7)$	8.4	20.4	$-91 \sim +122$
$Pixel-to-Pixel(\kappa_{occ}=5,\kappa_r=40)$	7.6	20.1	$-106 \sim +119$
Laser generated disparity map	33.2	6.0	$-63 \sim +68$

TABLE 11.2: Comparison of several binocular stereo algorithms *vs.* laser range scanning.

With this wide disparity range - 130 pixels - Pixel-to-Pixel does not outperforms the correlation algorithms. It has already been shown that it is very sensitive to the choice of its parameters (*cf.* Chapter 10), and once again, $\kappa_{occ}=5, \kappa_r=40$ does not perform well on this stereo pair.

Other parameter values: κ_{occ}, κ_r as well as the greyscale threshold for gradients have been tried outside the usual range of this study, table 11.3 reports these results.

The trials for other parameters for Pixel-to-Pixel - see table 11.3 - do not show significant improvement, so the laser generated disparity map has been compared

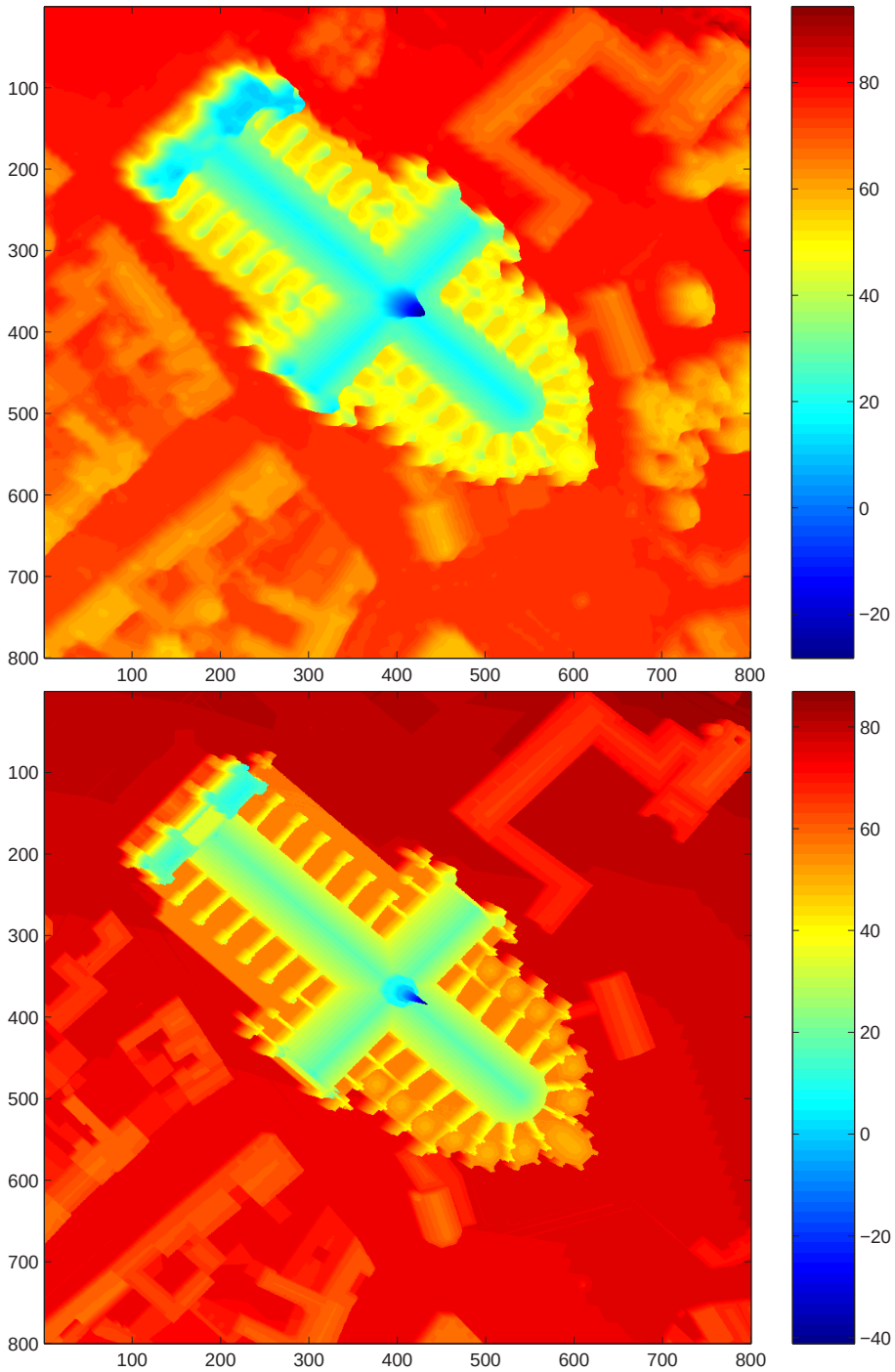


FIGURE 11.7: Disparity maps for the selected region of the IGN stereoset:
Top: laser generated map
Bottom: manually generated map.

to the $SAD(r = 10)$ computed one because it is the binocular stereo algorithm providing the highest number of good matches as well as the smallest standard deviation:

$P2P(\kappa_{occ}, \kappa_r, g_{thr}$	Good Matches (%)	Std. Dev.	Range
$P2P(5, 60, 5)$	7.9	23.5	$-109 \sim +120$
$P2P(5, 80, 5)$	8.3	26.7	$-107 \sim +120$
$P2P(225, 2, 5)$	N/A	N/A	N/A
$P2P(225, 60, 5)$	10.5	23.7	$-110 \sim +120$
$P2P(225, 80, 5)$	10.9	26.8	$-107 \sim +120$
$P2P(300, 2, 5)$	N/A	N/A	N/A
$P2P(300, 60, 5)$	11.4	23.6	$-110 \sim +120$
$P2P(300, 80, 5)$	11.7	27.1	$-107 \sim +120$
$P2P(5, 60, 10)$	7.8	23.8	$-109 \sim +120$
$P2P(5, 80, 10)$	8.1	27.0	$-107 \sim +120$
$P2P(225, 2, 10)$	N/A	N/A	N/A
$P2P(225, 60, 10)$	10.7	24.2	$-110 \sim +120$
$P2P(225, 80, 10)$	10.8	27.1	$-107 \sim +120$
$P2P(300, 2, 10)$	N/A	N/A	N/A
$P2P(300, 60, 10)$	11.6	24.0	$-110 \sim +120$
$P2P(300, 80, 10)$	11.8	27.1	$-107 \sim +120$
$P2P(5, 60, 20)$	8.1	24.4	$-109 \sim +120$
$P2P(5, 80, 20)$	8.2	27.5	$-107 \sim +120$
$P2P(225, 2, 20)$	N/A	N/A	N/A
$P2P(225, 60, 20)$	11.0	24.4	$-110 \sim +120$
$P2P(225, 80, 20)$	11.0	27.7	$-107 \sim +120$
$P2P(300, 2, 20)$	N/A	N/A	N/A
$P2P(300, 60, 20)$	11.6	24.2	$-110 \sim +120$
$P2P(300, 80, 20)$	12.0	27.4	$-107 \sim +120$
$P2P(5, 60, 50)$	8.7	24.6	$-110 \sim +120$
$P2P(5, 80, 50)$	8.9	27.8	$-110 \sim +120$
$P2P(225, 2, 50)$	N/A	N/A	N/A
$P2P(225, 60, 50)$	11.3	24.6	$-111 \sim +120$
$P2P(225, 80, 50)$	11.1	27.8	$-111 \sim +120$
$P2P(300, 2, 50)$	N/A	N/A	N/A
$P2P(300, 60, 50)$	12.0	24.7	$-111 \sim +120$
$P2P(300, 80, 50)$	12.0	27.6	$-111 \sim +120$

TABLE 11.3: Good matches percentage, standard deviation and error range for different values of Pixel-to-Pixel parameters.
'N/A' denotes parameters values for which Pixel-to-Pixel cannot match at all.

- firstly, in figure 11.8 showing the two different error distributions, and

- secondly, in figure 11.9 illustrating the disparity error values with their location: it shows the point difference between the groundtruth map on the one hand and the laser generated and $SAD(r = 10)$ computed maps on the other

hand. Darker values show small errors, the ranges of the two images have been normalised so that the colour used for an error value is the same.

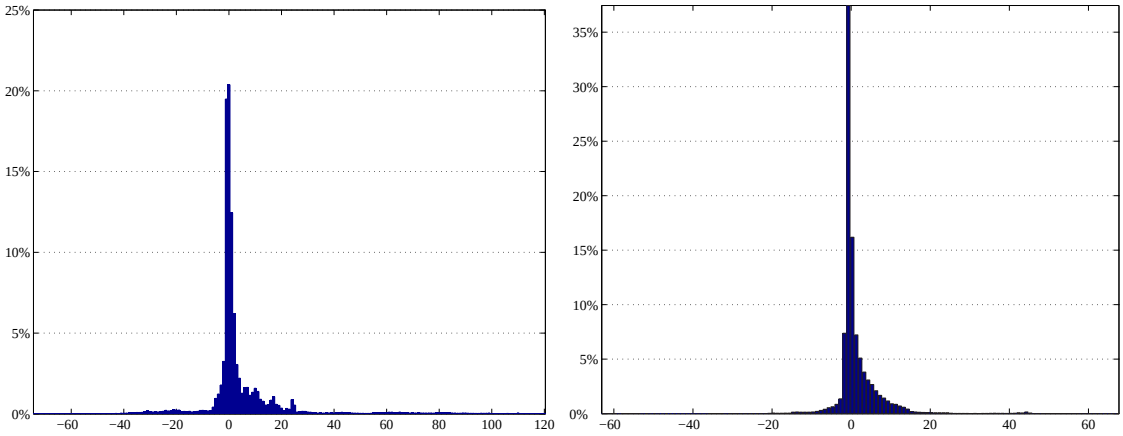


FIGURE 11.8: Aerial subset.
Left: $SAD(r = 10)$ error histogram
Right: Laser generated disparity error histogram.

From figure 11.9 one can verify that the laser generated map has less errors dealing with failure to meet the ordering constraint around the church tower, whereas SAD, in particular, and most of the other algorithms cannot deal with this situation because they are implicitly based on this constraint. Also, most of the differences between the laser generated disparity map and the $SAD(r = 10)$ computed one may be explained by the occlusions coming from the side walls of the church. These occlusions handicap algorithms like SAD whereas the laser method is not sensitive to such problems.

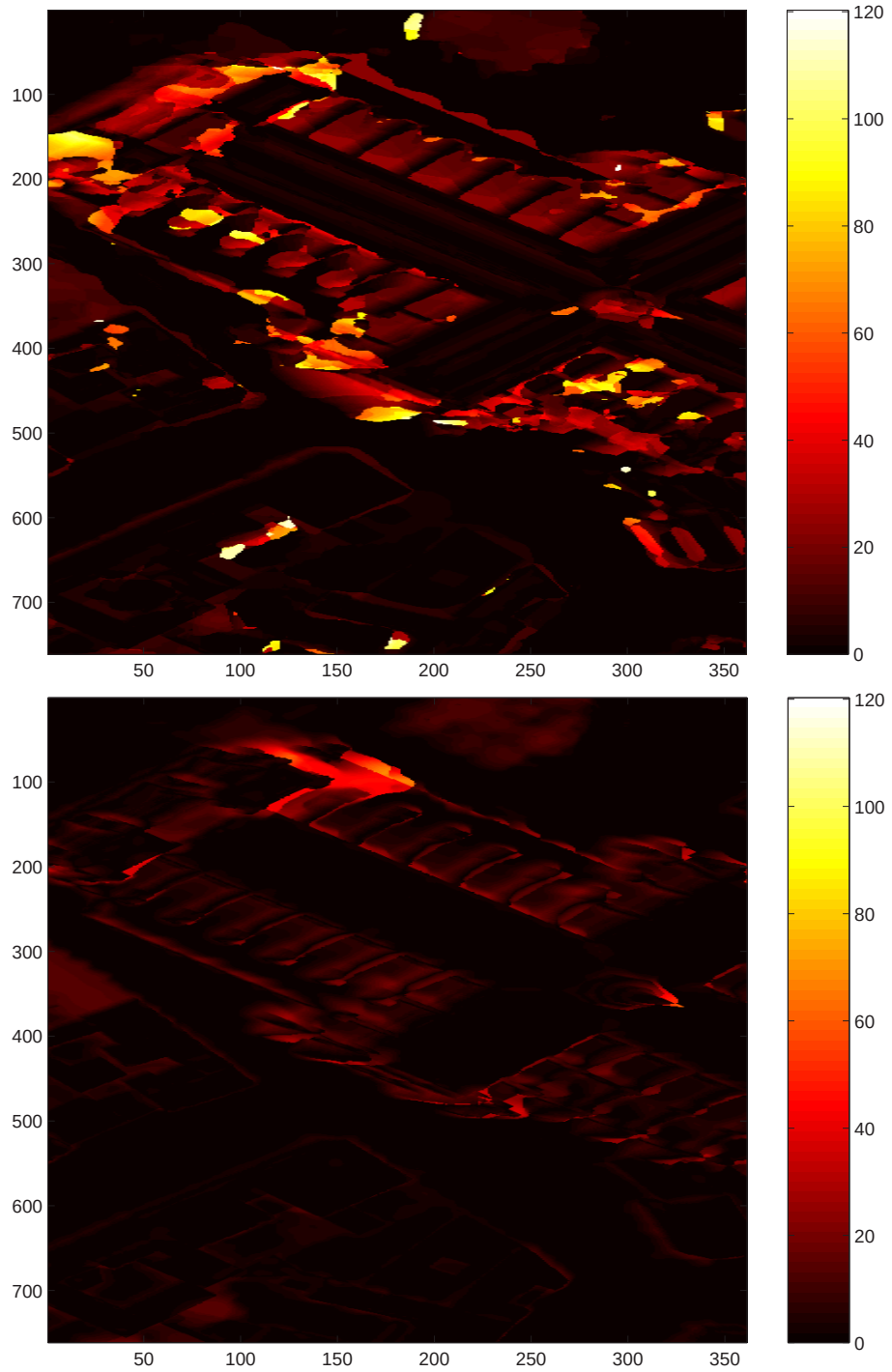


FIGURE 11.9: Aerial subset.
Top: $SAD(r = 10)$ error position
Bottom: Laser generated error position.

11.3 Summary

In all cases, the percentage of good matches was less than 20% using the ±0.5 pixel criterion for a good match, instead of 60% to 90% on other sets of images, and the standard deviation is ~ 20 against 10 or less for the other sets.

For the binocular stereo algorithms:

- in terms of good matches percentage:
  - the best is SAD with 20.4%,
  - the worst is Pixel-to-Pixel with 7.6%
- in terms of standard deviation:
  - the best is SAD with 16.12,
  - the worst is Corr2 with 25.97.

A clear example of a situation violating the ordering constraint has been illustrated. Stereovision algorithm are getting more and more accurate, Scharstein and Szeliski report more than 90% of good matches for Zabih *et al.* graph cut algorithm. However, all current algorithms use, at least implicitly this constraint. The next step for binocular stereo vision is to take into account situations where this constraint is not valid.

Having both a manual entry disparity map used as a ground truth and laser generated map also enabled a comparison of the studied binocular stereo algorithms with a different depth recovering method. It has been shown that the laser generated disparity map is of much better quality: 4 times more good matches, much smaller standard distribution of the error and a smaller error range. However, the because the algorithms chosen for study here remain in the scope of real-time **and** efficient hardware implementation, further comparisons with Zabih *et al.* Graph Cut - reported by Scharstein and Szeliski as the best current performing algorithm - would be needed to assess binocular stereo *vs.* laser. This also emphasizes the ordering constraint problem: the laser generated map gets most of the church tower right except for the sharp point at the apex, whereas stereo algorithms have very few chances of matching it properly as it shows two different faces and clearly breaks the ordering constraint.

Part III

Conclusion and Further Developments

Chapter 12

General Conclusion

Practical use of stereovision devices need an accuracy measure. The relationship between depth accuracy and camera configuration for the parallel camera axis configuration was studied leading to a relation between the absolute accuracy and the scene and cameras parameters. It was also extended to describe static measurement experiences setup and illustrated in the shell experience. A relative accuracy was also introduced for the case of dynamic situations like obstacle avoidance.

An initial assessment of the algorithms was performed as a reference. It was designed to qualify the algorithm's behaviour *vs.* their parameters, essential results are reproduced in table 12.1.

- Pixel-to-Pixel best performing parameters were found as $(\kappa_{occ} = 5, \kappa_r = 6)$.
- Correlation algorithms (Corr1, Corr2 and SAD) were found to have an optimal - good matches / processing time - window radius of 4. Increasing the window radius might slightly decrease the percentages of good matches but improves the error distribution. However, large correlation windows get very slow to process. The simplest correlation algorithm - SAD - was found to be almost as fast as Pixel-to-Pixel.
- Census optimal set of parameters was $(\alpha = 4, \beta = 3)$. It behaves like correlation algorithms in the way that increasing its window sizes slightly decreases the good matches percentage but reduces the error distribution. Census was by far the slowest, however it was coded on a standard PC whereas its bit patter use makes it a good choice for hardware implementation.

Because Census was the slowest and not as good performing as the other algorithms, two variants have been evaluated. The first one modified Census's costs by weighting them with their distance to the centre of the window. Computing a 'float'

Algorithm	Percentage of good matches					Complexity
	Corridor	Map	Sawtooth	Tsukuba	Venus	
$Census(4,3)$	64.3	83.0	85.9	62.4	85.6	$O(\alpha^2\beta^2\Delta)$
$P2P(5,6)$	64.8	75.8	85.6	74.6	80.4	$O(\Delta^2)$
$Corr1(4)$	62.9	79.7	82.8	70.5	82.3	$O(w^2\Delta)$
$Corr2(4)$	57.7	79.5	82.7	71.0	83.4	$O(w^2\Delta)$
$SAD(4)$	65.4	80.8	83.4	69.2	81.8	$O(w^2\Delta)$

TABLE 12.1: Initial results for the algorithms

distance inside the bit patterns of the Census algorithm affects its performance for hardware implementation. This approach did not show clear modifications of the Census results, it only changed the good matches percentages by a few percent both positively and negatively. The second one modified the way the bit patterns are formed inside the window, instead of comparing each pixel the the centre of the window, it compared them to the middle of their own line. This method showed most improvements - up to 17% - when not using the outer window at all and with a large enough inner window - at least a radius of 3 - as this method generates smaller bit patterns: one pixel is used as a reference in the original transform, several - *i.e.* as many as the window's height - are used in the variant.

Assessing stereo algorithms robustness to noise is a seldom tried experiment. Even when performed it is usually against a single value of signal to noise ratio. This information is vital if designing a stereo vision system. An experiment starting from a ray-traced image - *i.e.* noiseless - and corrupting it by carefully computed levels of additive white Gaussian noise was conducted.

- Pixel-to-Pixel performed best as long as its parameters were chosen correctly. The initial set, ($\kappa_{occ} = 5, \kappa_r = 6$), based on the noiseless - or near noiseless - images performed poorly against increasing levels of noise. A comparison of parameter sets against the overall best values led to the choice of ($\kappa_{occ} = 5, \kappa_r = 40$) as the most robust set against noise.

- Up to a SNR of 15dB Pixel-to-Pixel clearly outperforms the other algorithms of this study, and stays very stable up to a level of 33dB.

- Other algorithms do not show much differences, SAD is the second best and Census the worst both in terms of good matches percentages and error distribution.

Situations can be proposed where colour information would be essential, *i.e.* a red

and a blue pixel should not match even though they have the same intensity. This study tried two different approaches on the use of colour for correlation algorithms:

- separated: performing the algorithm on each colour band and select the best result, or
- combined: performing the algorithm on a combination of the colour bands.

These two approaches showed no real improvement *vs.* standard greyscale processing thus emphasizing the strong correlation between colour bands and intensity values. However, methods which improve upon greyscale matching have been reported in the literature, namely: colour block matching by Koschan *et al.* and the use of a colour gradient to qualify matching candidates by Jordan and Bovik. Koschan *et al.* claim a $25 \sim 30\%$ improvement of the matching. Jordan and Bovik claim between a few percent and +163%, such a range emphasizes the ambiguity of colour use: further research would be needed on these method to clearly evaluate the use of colour for stereo matching.

Although analysis suggests that increasing the baseline also increases the depth accuracy, this study highlighted practical problems introduced by larger baselines, *i.e.* :

- occlusions increase thus penalising the quality of matching,
- simple matching algorithms assume Lambertian or perfect scatterers surfaces, but increasing the baseline makes it harder to justify this assumption,
- increasing the baseline also increases the difference in pixels of the subtended surfaces.

This experiment used Pixel-to-Pixel and SAD as the two best candidates over this whole study. The range of baselines used was from 0.1m to 0.9m, the number of pixels in the processing window ranged from 44280 to 25488 respectively and the percentage of good matches from almost 70% to $20 \sim 25\%$, with a standard deviation between 0.5 and 18. This experience was also used to experimentally verify the accuracy given in the geometry study.

Stereo vision can be used as a non-invasive way of measuring objects. Following the geometrical study and setup descriptions a stereo pair of a fossil shell was taken both under normal lights conditions and with a colour pattern projected onto it. Active illumination - monochromatic or colour - has been reported in the literature - Kanade *et al.* and Koschan *et al.* - as clearly improving matching results. In this experiment also algorithms performed better on the active colour illuminated

set than on the ambient light one even though no ground truth was available to precisely quantify the improvement. The parameter set chosen for Pixel-to-Pixel as the best against noise - ($\kappa_{occ} = 5, \kappa_r = 40$) - did not provide nearly good results. This showed that Pixel-to-Pixel is really sensitive to the choice of its parameters. Finally ($\kappa_{occ} = 150, \kappa_r = 40$) was found as providing better results.

Another real experiment has been conducted using an aerial rectified stereo pair provided by the MATIS laboratory from the French national geographical institute (IGN). Two depth maps were given: a manual entry one and one obtain from laser range scanning. The manual entry depth map has been used as a ground truth to compare the stereo algorithms of this study *vs.* a different method: laser range scanning. The chosen subset illustrated a situation breaking the ordering constraint, where laser range scanning performs accurately and stereo algorithms are at a loss. Also occlusions due to building walls for instance introduce matching errors for standard algorithms whereas laser range scanning provides better results. The best algorithm - SAD - showed 20% good matches percentage with a standard deviation of 16 and an error range of 195 pixels, $-74 \sim +120$, *vs.* 38% good matches percentage with a standard deviation of 6 and an error range of 131 pixels, $-63 \sim +68$, for laser range scanning.

Chapter 13

Further Developments

Using colour information for matching seems obviously reasonable: a blue and a red point should not match even though they have the same intensity. However, the literature and this study illustrate a need for a complete evaluation of the use of colour to improve matching quality by reporting opposite results. Using colour means increasing the computational cost, therefore one wants to make sure that using colour will *consistently* improve the matching quality: this study showed no improvement with the use of colour and literature shows improvements from only a few percents to +163%.

Both the literature and this study showed a drastic improvement in the quality of matching when using active illumination - even monochromatic. In the case of obstacle avoidance, for cars for instance, projecting a "eye safe" pattern - *e.g.* infra-red - would significantly improve the quality of matching. Moreover using colour does not seem to *consistently* improve the quality of matching. Thus monochromatic active infra-red illumination would be an interesting candidate for robust stereo vision implementation.

The chapter on the aerial stereo pair started a first comparison of binocular stereo *vs.* laser generated disparity map. However this problem does not require, in general, the real-time constraint and more comparison with better performing algorithms is required: for instance Zabih *et al.*'s Graph Cut given as the best performing algorithm by Scharstein and Szeliski but which was not in the scope of this study. Depths for maps are usually obtained by manual entry, therefore methods to pre-process data are vital.

Usual approaches to binocular stereovision keep the two cameras fixed - with parallel or crossing axes - and try to match the pixels on the two images over a disparity range. However, another approach would be to have the two cameras to actively scan the scene - adjusting their fixation points. Objects at the fixation

point appear at zero disparity in the images presenting the potential to considerably reduce the range of disparities which needs to be checked.

Part IV

Appendix

Appendix A

StereoLib Software

A.1 Introduction

All the program has been written in ANSI C and been compiled on several environments like UNIX, LINUX and mostly CygWin[1]. The main modules are:

- StereoLibConfig configuration file defining the stereo pairs, algorithms and experiments.
- Census algorithm.
- Correlation algorithms.
- Pixel-to-Pixel algorithm.
- Histogram handles the histogram and metrics functions for the experiments.
- Picture, WorkMatrix, ... handles ppm images and internal float images formats.
- Data structure to handle all different used values: image coordinates, Census vectors, ...
- Data processing to provide functions like noise generator, endian format, ...
- MemInfo set of memory tracking function for debugging purpose.
- GNUMakefile to compile a program, create the documentation, backup data, ...

Most of the functions and header files are documented in the code itself using the ROBODoc format. The main feature are briefly described here.

[1] Cygwin is a Linux-like environment for Windows, see `www.cygwin.com`

A.2 Directories Organisation

The standard directories are defined starting from the user given root:

root '/cygdrive/c/Stereolib/' containing the *StereoLib.conf* configuration file - see section A.3 -, the *GNUMakefile* and the following folders:

- ColourMaps: containing a list of colour names (the rgb.txt from Emacs),
- Doc: containing the log files generated by ROBODoc[2]
  - html
  - rtf
  - LaTeX
- Exemples: containing exemple C programs using the StereoLib
- Lib: containing needed libraries
  - *multiPTC* [3] to create X graphic windows.
  - *Picture_XDisplay* to manage the windows created by *multiPTC* and convert *StereoLib* 's buffers to display them.
- Source: containing the source code for StereoLib
- StereoSets: containing the used StereoSets described in *StereoLib.conf*
  - StereoSet name
    - Experience name
- Algo directory: each algorithm has its own directory having its name (Census, Pixel-to-Pixel, SAD, ...)
  - StereoSet name
    - Experience name
- Temp directory

[2] Automating software documentation tool (ver. 4.0.6) , see `http://www.xs4all.nl/~rfsber/Robo/robodoc.html` for more information.

[3] *TinyPTC - www.gaffer.org/tinyptc*

A.3 Configuration File

The used configuration file is called *StereoLib.conf* and is in the root directory. The function `StereoLibConfig` - in `StereoLib.c` - is used to read the user specified file.

The format is the following:

- '#' is used for comments,
- Every line should finish with a semi-colon, except comment lines,
- All spaces are ignored, *i.e.* file and directory names cannot have spaces.

The configuration file should contain as follow:

- 'UNIXFILEFORMAT' or 'DOSFILEFORMAT' to choose the format separator '/' or '\'
- The StereoLib root directory, *i.e.* '/cygdrive/c/Stereolib/'
- Three different information lists, lists starts with a '{' and ends with a '}',
  - A list of possible stereosets directories: each line should contain the stereoset name, the width, height and the window to use, the function `TraceImgWindow` in the *ImgWindow* library can be used for defining the image processing window, the `StereoSetsSize.c` example illustrates the definition of stereosets.
  - A list of possible algorithm directories, each line gives an algorithm directory, an algorithm name and the number of parameters needed by the algorithm:
    - if the parameter number is missing it is set 0 by default,
    - if the algorithm directory or name is missing, it is replaced by a copy of the other one,
    - if both are missing they are replaced by '\0' (NULL string),
  - A list of possible experiments (*i.e.* Colour, AWGN, Baseline, ...): each line provides an experience name, an experience indicator, the number of parameters for the experiment number and a suffix, rules are the same than for the algorithm information.

A set of functions in the *StereoLibConfig* library access the different information from the algorithm and experiments. These functions retrieve the different information using either the number in the list or the name of the algorithm or experiment.

Note that the directory information is referred to as 'list' and the name is an alternative way to describe it, *i.e.* 'Corr1' for '/Correlation/Corr1'; so that the list directory can be called by a shorter reference name.

A.3.1 StereoSet File Name Format

The filename format for stereosets is inspired from the 'MRTStereo' file names:

- StereoSet name
  - P for pictures,
  - D for disparity maps and
  - O for occlusion maps,
    - L for left image,
    - R for right image,
      - Experience name,
      - Experience parameters,
      - Experience suffix.

A.3.2 Experience Results File Name Format

- StereoSet name
  - Algorithm name,
  - Experience name,
    - algorithm parameters,
      - Experience name,
      - Experience parameters,
      - Experience suffix.

A.4 Algorithms Description

A.4.1 Census algorithm

The main function *CensusAlgo* sets the right function pointers for the Census transform function and the vector creation function:

- **CensusRank* and
- **ImageVector* .

Additional libraries have been developed to handle specific objects related to the Census algorithms, like individual Census vectors or arrays (concatenated vectors over a window).

A.4.2 Correlation-based algorithms

The *CorrelationAlgo* library contains the different correlation algorithms, *CorrelationAlgoCheck* is used internally to associate the right function pointer to **CorrWindow* . Several correlation functions have been written for the different experiments. Function pointers makes it convenient to generate more correlation experiments. The main function , *CorrelationAlgo* , has two loops, one looking for a maximum correlation, the other one to look for a minimum, this is to prevent having the test inside the loop while running the timing for the program, the choice is made once and the chosen loop is ran.

A.4.3 Pixel-to-Pixel algorithm

The *Pixel2Pixel* has the main algorithms functions and uses *MatchArray* to handle the cost array of the algorithm and *Scanline* used to buffer one line of the image. The main function, *Pixel2Pixel* , takes as parameters a stereopair, its working window - see section 3.5.2 - the chosen type of loop (forward, backward or faster) and the occlusion cost and match reward.

A.5 Histogram

The *Histogram* handles the histogram creation, saving and reading. The *DispError* is used to create the disparity errors histogram as well as several plots. Finally the *LogArray* library manages an array of histogram to save the results of a complete experiment in one file.

A.6 Images structures

Two main image structures have been created, one to handle the PGM/PPM formats - *Picture* - for reading input images and writing easily viewable results. The second format contains 'double' floats - *WorkMatrix* - to keep as much precision as possible during the processing; functions have been written to convert one format into the other either for processing or for dumping the results in a viewable format.

A.7 Data

All in all there are lots of different structures to be considered. Some specific to the algorithms, like the Census vectors, some are shared, like the coordinates (two integers to represent a pixel position, three floats to save the minimum, maximum and normalisation factor for a matrix, ...). All structures have been designed with a constructor and a destructor. Different functions have been implemented to take care of the eventual endian differences, add noise to the images, apply filters, ...

A.8 Memory Information

The *MemInfo* library is an encapsulation of the *malloc* / *free* functions for debugging purpose. Linked lists are created with every memory allocations to keep track of them and check that all memory is properly freed.

A.9 GNUMakefile

A sample GNUMakefile is given, it combines different options from compiling the software with or without X windows[4] window support, create backups, build documentation using ROBODoc, ...

- Compilation
  - prog: makes the complete program using the standard StereoLib library.
  - progxdisp: makes the complete program using the standard StereoLib library and XDisplay libraries.
  - clean: removes all object files and libraries files.

[4] *TinyPTC* - *www.gaffer.org/tinyptc* - has been interfaced by *Picture_XDisplay* to display images using X-window.

  - Note that:
    * The program source and executable name can be overridden using:
      · PROGNAME=newprogname
      · EXECNAME=newexecname
    * If the program needs the tiff library add:
      · USETIFF=yes
    * To compile without the *assert* functions and without the *MemInfo* as debugging tool, add:
      · GO_ TIGER=yes

- Backup:
  - save: backups the files described in the BACKUPFILES variable,
  - The name of the archive and files to backup can be overridden using:
    * SAVENAME=backupdir/backupname
    * BACKUPFILES=dirs files.ext –exclude= –exclude=*.bak
  - Two variables can also be used:
    * BACKUPDIR contains the default backup dir,
    * TIMESTAMP creates a day and time string to concatenate with the file name.
  - The following situations are checked in order:
    * error if *tar* is not found,
    * archive *tar* and *bzip2* ,
    * archive *tar* and *gzip* if *bzip2* does not exist,
    * internal *tar* bzip2 format (-j) if *bzip2* and *gzip* do not exist.

- Predefined backup rules:
  - savedoc: backups the documentation related files,
  - savelib: backups the *StereoLib* 's related source files and libraries,
  - savestereosets: backups the stereosets used for the different experiments,
  - cleansave: cleans all the archive files.

- Documentation:

  - html: makes a html reference directory ./Doc/html, main file is masterindex.html
  - cleanhtml: removes files in the html reference directory and the log file in ./Doc
  - rtf: makes a rtf reference file in ./Doc/rtf
  - cleanrtf: removes files in the rtf reference directory and the log file in ./Doc
  - latex: makes a TeX reference file in ./Doc/LaTeX
  - cleanlatex: removes files in the latex reference directory and the log file in ./Doc
  - dvionly: creates the dvi file only in ./Doc/LaTeX, needs to have a TeX file already made.
  - dvi: calls latex to create the TeX file and makes the dvi file in ./Doc/LaTeX
  - pdfonly: creates the pdf file only in ./Doc/LaTeX, needs to have a dvi file already made.
  - pdf: calls dvi to create the dvi file and makes the pdf file in ./Doc/LaTeX
  - psonly: creates the ps file only in ./Doc/LaTeX, needs to have a dvi file already made.
  - ps: calls dvi to create the dvi file and makes the ps file in ./Doc/LaTeX
  - docall: makes all the documents file format using: html rtf latex dvionly pdfonly psonly
  - cleandoc: cleans all the documents directories and log files using: cleanascii cleanhtml cleanrtf cleanlatex
  - cleanall: clean program, documentation and archive files using: clean cleandoc and cleansave

- Note that:
  - help: displays this help
  - config: shows the main configuration for the makefile '(names, directories and need commands)',
  - the makefile must be started from its directory,
  - the documentation is extracted from the source files using "ROBODoc" version 4.0.6,

- once the TeX file created *latex* and *makeindex* commands are needed to create the reference file with an index,
- if the makeindex command is missing a warning is given and the dvi file is created without the index,
- the pdf format needs *dvipdf* to create the reference file in pdf format,
- the ps format needs *dvips* to create the reference file in ps format,
- all needed command must be found in the 'PATH', if not an error is generated.

Appendix B

Corridor Stereo Pair Scene Description File

B.1 Corridor.msd : main file

```
// GLOBAL PARAMETERS

// LEFT
vector eyecenter = (250,-150,170),
       lookpoint = (100,1600,100);

// RIGHT
// vector eyecenter = (259.963,-149.146,170),
//        lookpoint = (109.963,1600.85,100);

COLORRANGE   255
EYEP         eyecenter
LOOKP        lookpoint
UP           (0,0,1)
FOV          <30,30>
SCREEN       <100,100>

BACKGROUND   (0,0,0)

AMB_LIGHT    (50,50,50)

// ======= rows of 2 LIGHTS ==============
SPOT_LIGHT    (90,255,90) (250,-140,160) (0,280,-190) 15
```

```
POS_LIGHT    (150,150,150) (100,250,300)
POS_LIGHT    (120,120,120) (300,250,300)
POS_LIGHT    (150,150,150) (100,650,300)
POS_LIGHT    (150,150,150) (300,650,300)
POS_LIGHT    (150,150,150) (100,1050,300)
POS_LIGHT    (150,150,150) (300,1050,300)

/// ============== FLOOR ================
//  200x200 area of 50x50cm plates (see tile.inc)
//
TRANS(0,0,0)
  INCLUDE boden.inc
TRANS(200,0,0)
  INCLUDE boden.inc
TRANS(0,200,0)
  INCLUDE boden.inc
TRANS(200,200,0)
  INCLUDE boden.inc
TRANS(0,400,0)
  INCLUDE boden.inc
TRANS(200,400,0)
  INCLUDE boden.inc
TRANS(0,600,0)
  INCLUDE boden.inc
TRANS(200,600,0)
  INCLUDE boden.inc
TRANS(0,800,0)
  INCLUDE boden.inc
TRANS(200,800,0)
  INCLUDE boden.inc
TRANS(0,1000,0)
  INCLUDE boden.inc
TRANS(200,1000,0)
  INCLUDE boden.inc
TRANS(0,1200,0)
  INCLUDE boden.inc
TRANS(200,1200,0)
  INCLUDE boden.inc
TRANS(0,1400,0)
```

```
  INCLUDE boden.inc
TRANS(200,1400,0)
  INCLUDE boden.inc

// =============== CEILING =================
SURFACE  8 .2; (200,200,200)  .8;  (0,0,0)    0, 0; (0,0,0) 0
QUADRANGLE 8 (0,0,350) (0,1600,350) (400,1600,350) (400,0,350)

// =============== BACK ===============
SURFACE  9 0.6; (200,200,200) 0.4; (0,0,0)  0, 0; (0,0,0) 0
QUADRANGLE 9 (0,1600,0) (400,1600,0) (400,1600,350) (0,1600,350)

// =============== WALLS ===============
SURFACE  0 .8;  (175,175,175) .4;  (0,0,0)  0, 0; (0,0,0) 1
SURFACE  1 .2; (200,200,200) .8;  (0,0,0)  0, 0; (0,0,0) 1
SURFACE  2 .5; (70,20,20)     1; (0,0,0)  0, 0; (0,0,0) 1
SURFACE  3 .1;  (190,190,190) .6; (0,0,0)  0, 0; (0,0,0) 1

TRANS(0,0,0)
  INCLUDE wand.inc
TRANS(0,400,0)
  INCLUDE wand.inc
TRANS(0,800,0)
  INCLUDE wand.inc
TRANS(0,1200,0)
  INCLUDE wand.inc
ROT_Z(180)*TRANS(400,400,0)
  INCLUDE wand.inc
ROT_Z(180)*TRANS(400,800,0)
  INCLUDE wand.inc
ROT_Z(180)*TRANS(400,1200,0)
  INCLUDE wand.inc
ROT_Z(180)*TRANS(400,1600,0)
  INCLUDE wand.inc

// ================ OBJECTS ====================
//
SURFACE  5 1;  (130,130,240) 0.7;  (0,0,0) 0, 0;  (0,0,0) 1
```

```
TRANS(250,140,0)
  SPHERE   5 (0,0,33) 33

SURFACE  6 .6;  (255,100,100) .9;  (0,0,0) 0, 0;  (0,0,0) 1

TRANS(150,320,0)
  CONE   6 (0,0,0) (0,0,100) 30


SURFACE 10 TEXTURE_2D24BIT 0.3; (250,250,250) 1; (0,0,0) 0, 0; (0,0,0) 1; "lena.ppm"
SURFACE 11 TEXTURE_2D24BIT 0.3; (250,250,250) 1; (0,0,0) 0, 0; (0,0,0) 1; "teaset.ppm"

TRANS(51,300,80)
    PARALLEL 10 (0,0,0) (0,195,0) (0,0,212)
ROT_Z(180)*TRANS(349,510,100)
    PARALLEL 11 (0,0,0) (0,200,0) (0,0,200)


SURFACE  4 .2;  (120,85,85) .9;  (0,0,0) 0, 0;  (0,0,0) 1
TRANS(220, 1100, 0)
    SUPERQ 4  (0,0,30)  <50,50,30>  <2.9,2.9,3.5>


ENDFILE
```

B.2 Boden.inc : included files

```
SURFACE  1  .3;  (  0,  0,  0) .7;  ( 80, 80, 80) 0, 0;  (0,0,0) 1
SURFACE  2  .3;  (200,200,200) .7;  (200,200,200) 0, 0;  (0,0,0) 1

// 4 x 4 tiles, alternating in black and white
// each tile measures 50 x 50 resulting in a 200 x 200 square
// with a black tile in the lower left (at (0,0))

PARALLEL 1  (0,0,0)  (50,0,0)  (0,50,0)
PARALLEL 2  (50,0,0) (100,0,0) (50,50,0)
PARALLEL 2  (0,50,0)  (50,50,0) (0,100,0)
PARALLEL 1  (50,50,0) (100,50,0)  (50,100,0)
```

```
TRANS(100,0,0) BEGIN
  PARALLEL 1  (0,0,0)  (50,0,0)  (0,50,0)
  PARALLEL 2  (50,0,0) (100,0,0)  (50,50,0)
  PARALLEL 2  (0,50,0)  (50,50,0)  (0,100,0)
  PARALLEL 1  (50,50,0) (100,50,0)  (50,100,0)
END

TRANS(0,100,0) BEGIN
  PARALLEL 1  (0,0,0)  (50,0,0)  (0,50,0)
  PARALLEL 2  (50,0,0) (100,0,0)  (50,50,0)
  PARALLEL 2  (0,50,0)  (50,50,0)  (0,100,0)
  PARALLEL 1  (50,50,0) (100,50,0)  (50,100,0)
END

TRANS(100,100,0) BEGIN
  PARALLEL 1  (0,0,0)  (50,0,0)  (0,50,0)
  PARALLEL 2  (50,0,0) (100,0,0)  (50,50,0)
  PARALLEL 2  (0,50,0)  (50,50,0)  (0,100,0)
  PARALLEL 1  (50,50,0) (100,50,0)  (50,100,0)
END

ENDFILE
```

B.3 Wand.inc : included files

```
///
// WALLS of a room
QUADRANGLE 1 (50,0,350) (50,0,0) (50,150,0) (50,150,350)
QUADRANGLE 0 (50,150,350) (50,150,0) (0,150,0) (0,150,350)
QUADRANGLE 1 (50,250,350) (50,250,0) (50,400,0) (50,400,350)
QUADRANGLE 0 (0,250,350) (0,250,0) (50,250,0) (50,250,350)

// DOOR  of a room
QUADRANGLE 2 (0,150,350) (0,150,0) (0,250,0) (0,250,350)

// SIGN  of a room
BOX 3 (50.5,280,157.5) <1,10,7.5>

ENDFILE
```

Appendix C

Assessing Stereo Algorithm Accuracy (IVCNZ'02)

Assessing Stereo Algorithm Accuracy

Philippe LECLERCQ*
CIIPS, Centre for Intelligent Information Processing Systems
The University of Western Australia
Visiting the Department of Computer Science
The University of Auckland

John MORRIS†
Department of Computer Science
The University of Auckland

Abstract

Despite the large numbers of papers proposing new algorithms for the stereo matching process, there is a dearth of quantitative *comparisons of proposed algorithms. Scharstein and Szeliski have recently compared area-based stereo matching algorithms on two metrics: fraction of pixels for which the disparity is not calculated correctly and the root mean square error using a small set of real images with known disparities. In this work, we start with 'perfect' images produced by a ray-tracer from a scene model and corrupt the images with varying amounts of additive white gaussian noise. This enables us to assess matching algorithm performance on perfect images and in the presence of defined amounts of noise. We found that the Pixel-to-Pixel algorithm, although it performs well in the absence of noise, rapidly deteriorates compared to the other algorithms. In the low noise region, there is very little difference between the algorithms we studied and thus one could chose the simplest and fastest - sum of absolute differences - as any benefit from others is very small. At higher noise levels (SNR $\leq$ 16dB), a sum of squared differences (computationally the next cheapest) performs consistently better.*

1 Introduction

This work was originally motivated by an attempt to implement a stereo algorithm in hardware. Zabih and Woodfill claim that their Census transform is suitable for this [Zabih and Woodfill 1994], but, noting that there are several major groups of stereo matching algorithms and dozens of variants of the individual algorithms within those groups, we felt that a feasibility study to determine which algorithm(s) perform best was needed. We soon discovered, that, save one very recent work [SZELISKI 2002], there has been no serious effort to compare the performance of stereo algorithms and thus provide researchers with benchmarks by which to judge new algorithms and implementers with cost-performance trade-off data which can guide, for example, hardware implementation efforts [1]. Scharstein *et al.* provide the first thorough study in this area - comparing over 20 algorithms or variants on two quality measures [SZELISKI 2002]. Our work extends theirs by including assessments of robustness to noise.

1.1 Motivation

When one compares the computed disparity maps to the correct disparity map (the 'ground truth') in figure 1, it is not at all obvious which algorithm is best. The two disparity maps have about the same number of correct disparities (62.74% and 62.65%) but the standard deviation indicates that the spread of disparity errors for Pixel-to-Pixel is less than for Census (1.51 *vs* 3.45). This illustrates the importance of metrics that enable quantitative comparisons.

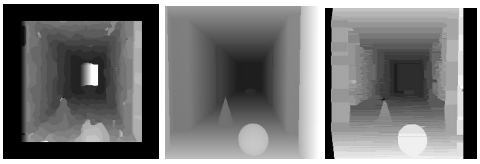


Figure 1: Disparity maps produced by the Census (left) and Pixel-to-Pixel (right) algorithms compared with the ground truth disparity map (centre)

1.2 Images

We used Gerdes' MRTStereo tool [Gerdes 2001]: it takes a scene description and, by ray-tracing, generates left and right colour images for a specified camera baseline as well as precise disparity maps (64-bit floating point values) and occlusion maps. The precise disparity maps enable precise measurement of an algorithm's performance. To simulate real images, we contaminated the perfect images with different levels of additive white Gaussian noise. The red, green and blue channels of the colour images were treated independently: noise was added to each channel separately. Greyscale images were generated by summing the intensities in the R, G and B channels. The amount of noise is described by the signal-to-noise ratio and measured in dB. A precise definition of the meaning of "*xdB*" of noise in our experiments is given in the appendix.

Most algorithms match over a window of pixels to overcome noise problems. To avoid edge effects from these windows affecting our results, metrics were calculated on a central window of $200 \times 200 = 40{,}000$ pixels in the calculated 256×256 pixel images. The black borders in the disparity maps in figure 1 represent the areas where disparities were not calculated.

1.3 Algorithms

Stereo matching algorithms can generally be classed as area- or feature-based. This study focuses on basic area based algorithms: parallel work in our laboratory is assessing feature based algorithms [Davey 2002].

For this study, we chose a sample of basic area matching algorithms, Birchfield and Tomasi's Pixel-to-Pixel algorithm [Birchfield and Tomasi 1996] as a representative of the dynamic algorithms and for its ability to handle occlusions (it is the only one in this study with this capability) and Zabih and Woodfill's non-parametric Census algorithm [Zabih and Woodfill 1994]. The algorithms used are summarized in table 1.

*e-mail: plec003@cs.auckland.ac.nz
†e-mail: jmor159@cs.auckland.ac.nz

[1] The high degree of parallelism present in the matching algorithm makes stereo matching a classic problem for specialized hardware [Morris 2001].

Corr1: Normalized Intensity Difference	$\frac{\sum(I_L-I_L)^2}{\sum I_L^2 \cdot \sum I_R^2}$	[Faugeras et al. 1993]
Corr2: Correlation	$\frac{\sum I_L \cdot I_R}{\sqrt{\sum I_L^2} \cdot \sqrt{\sum I_R^2}}$	[Faugeras et al. 1993]
SAD: sum of absolute differences	$\sum \lvert I_R - I_L \rvert$	[Faugeras et al. 1993]
Pixel-to-Pixel	Dynamic	[Birchfield and Tomasi 1998]
Census	Census Transform	[Zabih and Woodfill 1994]

Table 1: Algorithms used
In the first four algorithms, the summations are taken over all pixels in the matching windows.

1.4 Method

For each algorithm, we first identified the best set of parameters for the perfect images. In the first four algorithms, this involved determining the best window size. For Pixel-to-Pixel there are two parameters the match reward and occlusion penalty. For Census, there are two 'windows': the length of the Census vector (α) and the radius of the correlation window (β). Once the best parameters for the perfect images were found, they were used in the noise experiments. Here we first compute the algorithm on perfect images with different sets of parameter, once the best set found, we submit it to increasing levels of noise. For every pixel in an image, we use 200x200 windows i.e. 40000 individual error measures

Scharstein & Szelinsky [SZELISKI 2002] used two measures for comparison: the percentage of wrong disparity pixels for which the disparity was more than one unit and the root mean square error:

$$RMS = \frac{1}{N} \sum_{(x,y)} |d_c(x,y) - d_t(x,y)|^2 \quad (1)$$

We used three metrics: the percentage of accurate disparity pixels but we describe the histogram of disparity errors by its mean and standard deviation.

$$\sigma = \frac{1}{N} \sum_{(x,y)} \left((d_c(x,y) - d_t(x,y)) - \overline{d}_{err} \right)^2 \quad (2)$$

where $d_c(x,y)$ is the computed disparity and $d_t(x,y)$ the ground truth disparity at (x,y) and $\overline{d}_{err}$ is the mean of the disparity error, $d_{err} = d_c - d_t$.

As the formulae show, the difference between the RMS and the standard deviation is that the standard deviation is centred around its mean, whereas the RMS takes the bias of the mean. Thus we report both the histogram spread and its mean so as to reveal any bias an algorithm might show, whereas the RMS merges both factors into a single metric.

1.5 Results

Initially, we used every algorithm on 'perfect' (noise-free) image pairs with a wide range of parameters. We looked for the parameter set giving the best zero error fraction value as well as the lowest standard deviation. Note that these do not always coincide, *cf.* figure 2 and figure 3. The following graphs plot zero error fraction and standard deviation for Pixel-to-Pixel and Census algorithms.

For the Pixel-to-Pixel algorithm, we did not implement the 'post-processing' step: our intention here is to compare the basic area-matching algorithms. Heuristics such as the scanline to scanline

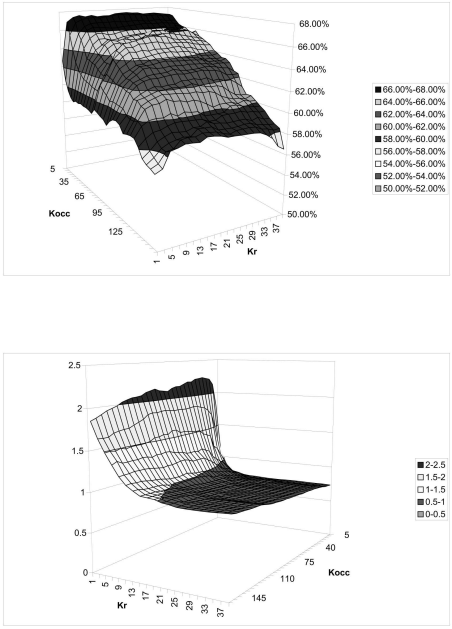
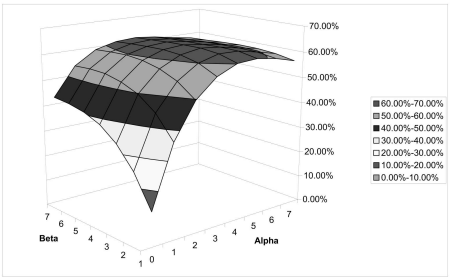


Figure 2: Pixel-to-Pixel algorithm zero error fraction and standard deviation *vs* the two adjustable parameters, κ_{occ} and κ_r



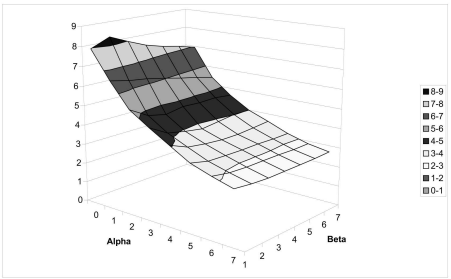
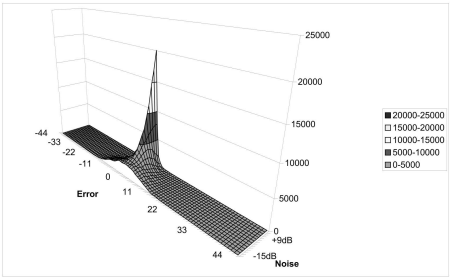


Figure 3: Census algorithm zero error fraction and standard deviation *vs* the window size parameters, α and β



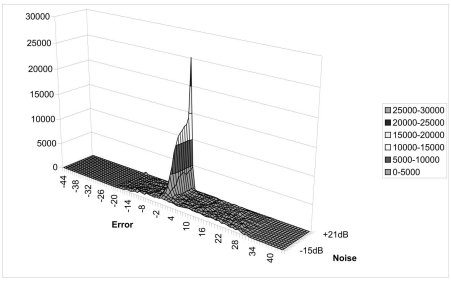


Figure 4: Pixel-to-Pixel and Census algorithms: Disparity error histograms for different SNR values.

post-processing used by Birchfield and Tomasi could be applied by any other approach.

Once we had determined the best working parameter sets, we ran the algorithms on noise corrupted images. We ran experiments on increasing amounts of noise until we reached a point ($SNR = -12dB$) where the number of correct disparity matches was the same as that which would have been generated by simply guessing. Our images have a maximum disparity of 22, so that for a window containing 40,000 pixels, random guessing would produce a correct result for $40,000/22 \sim 1,800$ or $\sim 4.5\%$ of pixels. The SNR=-12dB image in figure 6 shows that very little information remains. Figures 4 show the Pixel-to-Pixel and Census histograms for noise values from SNR=-15dB to +21dB and $+\infty$ dB. Note that the Census results show a plateau in matching performance at moderately high noise levels whereas the Pixel-to-Pixel algorithm's performance continues to drop from an initial high value.

Finally we compared algorithms in terms of zero error fraction and standard deviation against SNR as a parameter.

Figure 5 shows that most of the correlation algorithms, or correlation related (*e.g.* Census), have a better robustness to increasing noise levels with an almost flat section from +9dB to +21dB. The dynamic approach (Pixel-to-Pixel) performs better on noiseless images but its performance degrades much faster when noise is added.

Figure 5 shows the spread from the mean in each histogram. All the standard correlation algorithms (Corr1, Corr2, SSD and SAD) perform quite well - producing results in a narrow band for all noise levels. However Census has a wide spread of disparity errors and Pixel-to-Pixel shows slightly narrower spreads. Again Pixel-to-

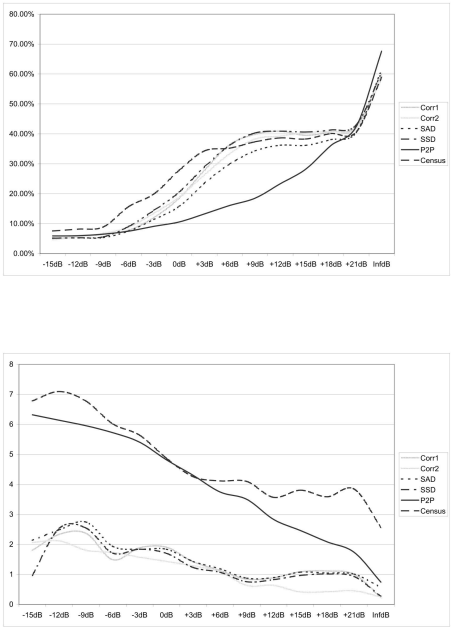


Figure 5: Zero error (left) and standard deviation (right) for all algorithms *vs* SNR.

Pixel performs much better with very little noise.

These two graphs show that normalising factors in the correlation formulae do not improve the results and simply add a lot of computing time (*cf.* Corr1 and Corr2 formulae *vs* SAD and SSD). In fact, the computationally simpler SSD algorithm produces better results in less time than all other algorithms save that Pixel-to-Pixel algorithm performs better on 'good' (*i.e.* low noise) images.

The Census algorithm shows a sharp degradation in performance as noise is added: this is probably due to the fact that the census transform does not take the distance of a pixel from the reference pixel into account so that the Hamming distance might be significantly perturbed by a few differences due to noise.

2 Conclusion

The metrics we have defined enable unbiased assessments of the quality of stereo matching algorithms and thus provide an objective means for designers of new algorithms to compare their results with previous work. Our initial measurements would seem to favour computationally simpler algorithms because their matching performance is roughly equivalent to that of much more complex and time consuming algorithms. However, Birchfield and Tomasi's Pixel-to-Pixel algorithm showed better performance on images with high SNR's. From the streaks which appear in disparity maps generated by Pixel-to-Pixel, it would appear that once it 'gets lost', *i.e.* calculates a wrong disparity - by choosing a bad path, it becomes very difficult for it to recover. Thus an error in one region of an image propagates to other regions. Noise increases the probability that a sub-optimal path will be chosen and this will often be reflected in poor choices for subsequent pixels on the path.

In future work, we intend to use some better statistical methods (*e.g.* χ^2) to confirm, for example, that in the plateau region of figure 5 all the algorithms are essentially equivalent in performance and that the best choice is therefore the computationally cheapest one. This emphasizes the need for a better descriptive metrics.

We are also interested in using our metrics to evalutate the effect of colour information on stereo algorithms. Most work to date has been based on greyscale images. Defined metrics will enable us to measure exactly any improvement that might appear.

Current work involves a precise understanding of the baseline distance as well as using the texture to adaptively adjust matching parameters.It has been found experimentally that the Census algorithm parameters are sensitive to the level of texture of the input images. Using a texture metric [SZELISKI 2002] we want to adapt parameters depending on the local texture.

Noise is usually measured by a signal to noise (SNR) ratio expressed in dB and defined as:

$$SNR = 10 \cdot \log_{10}\left(\frac{P_{signal}}{P_{noise}}\right) \tag{3}$$

where P is the power. A SNR of 0dB means that the power of the signal is equal to the power of the noise and increasing the SNR 3dB means doubling the power of the signal.

For a discrete image, we can define the power as the mean of the squared intensity values:

$$P_{signal} = \frac{1}{n_{pixels}} \cdot \sum_{image} I_p^2$$

where I_p is the intensity of a pixel at p.

We now define a Gaussian with parameters μ and σ: $N(\mu,\sigma)$ where μ is the mean and σ the standard deviation. For a centred Gaussian ($\mu = 0$): the power is defined as the variance of the distribution:

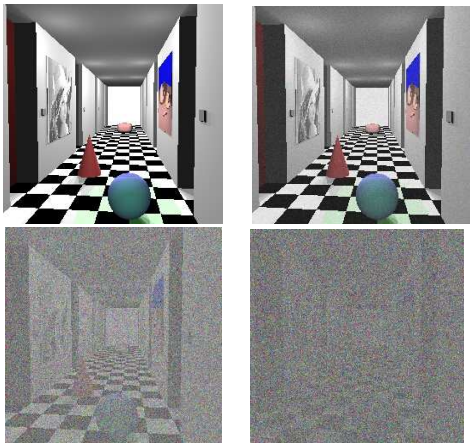


Figure 6: Greyscale images with no added noise, 24dB, 0dB and -12dB.

$$P_{noise} = Var = \sigma^2 \tag{4}$$

If we know the input image's power, P_{signal}, then the standard deviation of the noise, σ, to be added to produce any given *SNR* is:

$$\sigma = \sqrt{Var} = \sqrt{\frac{P_{signal}}{10^{\frac{SNR}{10}}}} \tag{5}$$

To generate noise, we use two random number generators taken from Press *et al.*[Willian H. Press and Flannery 1992]: the uniform deviate was taken from the `ran1` function[2] and the normal centred deviate, $N(0,1)$, calculated using the `gasdev` function[3]. This is converted to the desired normal distribution using the property:

$$\text{if } X \text{ follows } N(0,1) \text{ then } a \cdot X + b \text{ follows } N(b, a^2)$$

Figure 6 shows on the left the original left image with no noise and the same image with increasing levels of noise. The highest level of noise (SNR=-12dB) shows that the image no longer contains any valid information. The lowest represented level of noise (SNR=+24dB) looks similar to the 'perfect' image, but, in fact, a SNR of +24dB in power means a Gaussian noise with a $\sigma \sim 9$ and therefore noise values of $\sim 6\%$ of signal values with a mean intensity of ~ 153. This explains the initial sharp drop seen in figures 4 and 5.

References

BIRCHFIELD, S., AND TOMASI, C. 1996. Depth discontinuities by pixel-to-pixel stereo. Technical Report CS-TR-96-1573, Stanford University, Department of Computer Science, July.

BIRCHFIELD, S., AND TOMASI, C. 1998. Depth discontinuities by Pixel-to-Pixel stereo. In *Procedings of the Sixth International Conference on Computer Vision (ICCV-98)*, Narosa Publishing House, New Delhi, S. Chandran and U. Desai, Eds., 1073–1080.

DAVEY, C. 2002. *Feature-based Stereo Matching*. BE(Hons) Thesis, UWA (work in progress).

FAUGERAS, O., HOTZ, B., MATHIEU, H., VIÉVILLE, T., ZHANG, Z., FUA, P., THÉRON, E., MOLL, L., BERRY, G., VUILLEMIN, J., BERTIN, P., AND PROY, C. 1993. Real time correlation-based stereo: algorithm, implementatinos and applications. Tech. Rep. 2013, Institut National De Recherche en Informatique et en Automatique (INRIA), 06902 Sophia Antipolis, France.

GERDES, V. 2001. *Modular Rendering Tools*. `www-student.informatik.uni-bonn.de/~gerdes/MRTStereo/index.html`.

MORRIS, J. 2001. Reconfigurable computing. In *Computer Engineering Handbook*, CRC Press, V. G. Oklobdzija, Ed., 37–1 – 37–16.

SZELISKI, D. S. . R. 2002. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision 47* (Apr.–June), 7–42.

WILLIAN H. PRESS, SAUL A. TEUKOLSKY, W. T. V., AND FLANNERY, B. P. 1992. *Numerical Recipes in C: The Art of Scientific Computing*. Press Syndicate of the University of Cambridge.

ZABIH, R., AND WOODFILL, J. 1994. Non-parametric local transforms for computing visual correspondence. *Lecture Notes in Computer Science 800*, 151–158.

[2] Chapter 7.1, page 280 [Willian H. Press and Flannery 1992]

[3] Chapter 7.2, page 289 [Willian H. Press and Flannery 1992]

Appendix D

Robustness to Noise of Stereo Matching (ICIAP'03)

Robustness to Noise of Stereo Matching

Philippe Leclercq and John Morris
Centre for Intelligent Information Processing Systems,
School of Electrical, Electronic & Computer Engineering,
The University of Western Australia

Abstract

We measured the performance of several area-based stereo matching algorithms with noise added to synthetic images. Dense disparity maps were computed and compared with the ground truth using three metrics: the fraction of correctly computed disparities, the mean and standard deviation of the distribution of disparity errors.
For a noise-free image, Birchfield and Tomasi's Pixel-to-Pixel - a dynamic algorithm - performed slightly better than a simple sum-of-absolute differences algorithm (67% correct matches vs 65%) - considered to be within experimental error. A Census algorithm performed worst at only 54%. The dynamic algorithm performed well until the S/N ratio reached 36dB after which its performance started to drop. However, with correctly chosen parameters, it was superior to correlation and Census algorithms until the images became very noisy (~ 15dB). The dynamic algorithm also ran faster than the fastest correlation algorithms using an optimum window radius of 4 and more than 10 times faster than the Census algorithm.

1 Motivation

This work was originally motivated by an attempt to implement a stereo algorithm in hardware. Zabih and Woodfill claim that their Census transform is suitable for this [10], but, noting that there are several major groups of stereo matching algorithms and many variants of the individual algorithms within those groups, we felt that a feasibility study to determine which algorithm(s) perform best was needed. We soon discovered, that there has been little serious effort to compare algorithms and provide benchmarks for assessing new algorithms or cost-performance trade-off data which can guide, for example, hardware implementation efforts[1]. Scharstein and Szeliski[9] provide the first thorough study in this area - comparing over 20 algorithms or variants on two quality measures. Our work extends theirs by including assessments of robustness to noise.

[1]The high degree of parallelism present in the matching algorithm makes stereo matching a classic problem for specialized hardware[7].

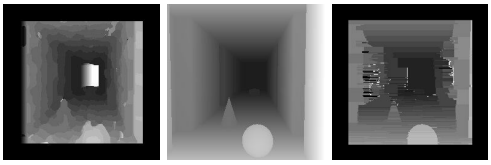


Figure 1. Disparity maps produced by the Census (left) and Pixel-to-Pixel (right) algorithms against the ground truth (centre).

When one compares the computed disparity maps in figure 1, it is not at all obvious which algorithm is best. The two disparity maps have about the same number of correct disparities (62.7% and 62.6%) but the standard deviation indicates that the spread of disparity errors for Pixel-to-Pixel is less than for Census (1.5 *vs* 3.4). Thus metrics that enable quantitative comparisons are important.

1.1 Algorithms

Stereo matching algorithms are generally classed as area- or feature-based. This study focuses on basic area-based algorithms: parallel work in our laboratory has assessed feature-based algorithms[3].

For this study, we chose a small group of basic area matching algorithms: the Pixel-to-Pixel algorithm[1] as a representative of the dynamic algorithms and for its ability to handle occlusions (it is the only one in this study with this capability) and several correlation-based algorithms, including the non-parametric Census algorithm[10], which has features permitting efficient hardware implementation. The algorithms are summarized in table 1.

Corr1: Normalized Intensity Difference	$\frac{\sum (I_L - I_L)^2}{\sum I_L^2 \cdot \sum I_R^2}$	[4]
Corr2: Correlation	$\frac{\sum I_L \cdot I_R}{\sqrt{\sum I_L^2} \cdot \sqrt{\sum I_R^2}}$	[4]
SAD: sum of absolute differences	$\sum \lvert I_R - I_L \rvert$	[4]
Pixel-to-Pixel	Dynamic	[2]
Census	Census Transform	[10]

Table 1. Algorithms used: In the first three algorithms, summations are over all pixels in the matching windows.

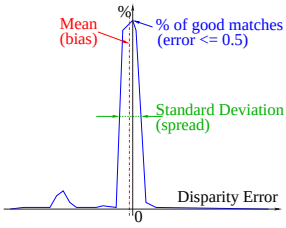


Figure 2. Sketch of typical distribution of disparity errors showing matching metrics

2 Metrics

We compared dense disparity maps computed from an image pair to a ground truth map: the differences were stored in a histogram, from which we calculated:

1. the fraction of good matches, *i.e.* calculated disparities within ± 0.5 of the correct disparity,
2. the mean of the disparity error distribution *and*
3. the standard deviation of the distribution.

Figure 2 shows a typical distribution. Note that no sub-pixel estimation techniques were employed. An appropriate disparity error clearly depends on the actual application using the disparity map. Less stringent criteria, such as that used by Scharstein and Szeliski[9], who consider within ± 1.5 pixels of the correct match as 'good', may be acceptable in many applications. Our stringent criterion produces numbers of good matches which may seem low when compared with other work. Since we are assessing relative algorithm performance, any *consistent* error may be used.
The fraction of good matches is clearly the most important metric but the mean and the standard deviation highlight characteristics of the algorithms or the way they are implemented. For example, a small peak to one side of the main peak (*cf.* figure 2) is due to aliasing. Such peaks appear often: when there is more than one match, the the direction in which the program scans the possible disparities causes the first one to be chosen resulting in a bias in the calculated disparity. The Census algorithm is particularly susceptible to such artefacts because the cost function is a sum of small integer values (Hamming distances).

3 Results

3.1 Images

We used the MRTStereo tool[5] to generate images: it takes a scene description and, by ray-tracing, generates left and right colour images for a specified camera baseline as well as precise disparity and occlusion maps. To simulate real images, we added different levels of Additive White Gaussian Noise to the R, G and B channels of the 'perfect' images independently. Greyscale images were generated by summing the intensities in the R, G and B channels. The amount of noise is described by the signal-to-noise ratio and measured in dB, *cf.* the appendix.
Most algorithms match over a window of pixels to overcome problems of noise and other artefacts introduced by, for example, lighting and perspective variations. To prevent edge effects, metrics were calculated on a central window of $216 \times 194 = 41,904$ pixels (in 256×256 images). The black borders in the disparity maps in figure 1 represent the areas where disparities were not calculated.

3.2 Choosing the best parameters

All the algorithms have parameters which govern their operation: typically the size of the window over which costs are aggregated although others such as the matching 'reward' and occlusion 'penalty' of the Pixel-to-Pixel algorithm are found. Even though the Census algorithm is claimed to be 'non-parametric', it uses two windows: one for the transform and one for the aggregation; whilst these window sizes may be chosen arbitrarily, matching performance depends strongly on them (*cf.* figure 3).
The optimal parameters for any algorithm also depend on the images themselves. A useful parameter set must produce reasonable results for any unknown image. In addition to two image pairs generated by the MRT tool ('Corridor' and 'Madroom'), we tested the algorithms on four sets captured by real cameras for which ground truth data is available and used by Scharstein and Szeliski[9] in their studies ('Tsukuba', 'Map', 'Sawtooth' and 'Venus'). Containing unknown levels of noise, they were used to ensure that parameters chosen for each algorithm were reasonable *i.e.* that they would lead to fractions of good matches close to the best obtainable with that algorithm.

The 'Madroom' images contain alternating black and white bands and present a difficult challenge for matching. None of the simple algorithms tested here handles them well - a typical set of results are shown in figure 4(b): at best, just over 30% of disparities are calculated correctly compared to $> 60\%$ for all other images. A more sophisticated algorithm, including additional heuristics such as the uniqueness constraint, is clearly needed for this scene. This work aimed to provide fundamental data on basic algorithms as the basis for further work, so we did not add additional rules.

In this work, a window radius of w implies a $(2w + 1) \times (2w + 1)$ square window.

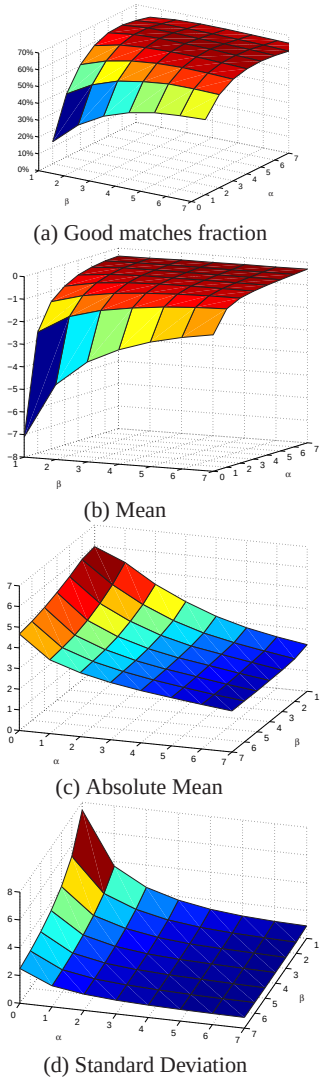
3.2.1 Census algorithm

Census concatenates the results of a transform over an 'inner' window of radius β around a pixel into a **census vector** over an 'outer' (correlating) window of radius α[10]. The transform compares the pixel intensities and thus Census is claimed to be 'non-parametric'. However, a bad choice of the sizes of the two windows can degrade performance, see figure 3.

Figure 3(a) shows that choosing ($\alpha = 4$, $\beta = 3$) produces near to optimal numbers of good matches for all image sets (even the Madroom pair, although the overall performance of the algorithm is poor).

Figure 3 plots the disparity error's distribution mean, absolute mean and standard deviation *vs* α and β. Optimum values for α are in the region 3 to 7: performance is not significantly affected by the choice of β. Since computation time is $\mathcal{O}(\alpha^2\beta^2\Delta)$, these curves also guide the selection of parameters for real-time applications where one may sacrifice accuracy for speed.

Plots of the disparity error's distribution means and standard deviation in figure 3 show that: increasing the window radius decreases the bias - the mean approaches the centre of the distribution - and narrows the peak. The bias is typically less than one disparity unit for reasonably sized windows ($\alpha \geq 3$). The distribution keeps narrowing as both window sizes increase, even though the fraction of good matches drops slightly. This is the blurring associated with larger windows informally described by many researchers: it indicates that larger windows may be appropriate when applications can tolerate a larger disparity error in return for a higher number of acceptably matched points. Our experiments thus include two sets of parameters for all the correlation algorithms: one using a small window which optimizes the number of good matches and a larger one which produces a much narrower distribution. For Census, ($\alpha = 3, \beta = 4$) produces high match fractions for all images and ($\alpha = 7, \beta = 5$) produces the narrowest distributions.



(a) Good matches fraction

(b) Mean

(c) Absolute Mean

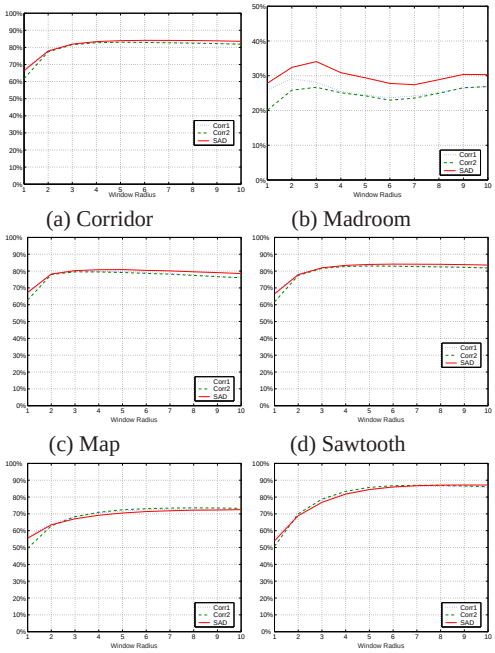
(d) Standard Deviation

Figure 3. Census: fraction of good matches, mean, absolute mean and standard deviation of the disparity error distribution *vs* the window radius parameters, α and β. Images: Corridor

3.2.2 'Correlation' algorithms

Here window size is the key parameter, so we measured how it affects matching performance.

From figure 4, it is apparent that for all images the num-



(a) Corridor (b) Madroom

(c) Map (d) Sawtooth

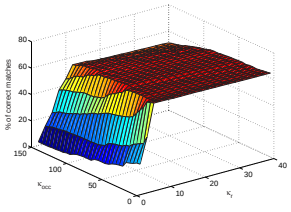
(e) Tsukuba (f) Venus

Figure 4. Good matches *vs* window radius for the six image sets: Cost functions: Corr1, Corr2, SAD

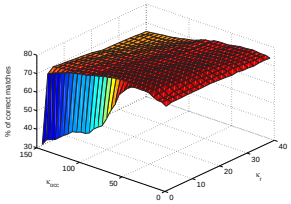
ber of correct matches increases until $w = 4$. For larger windows, any improvement is small and slight degradations were seen for $w \geq 6$. The 'Madroom' pair is the exception - showing a different behaviour to all the others. Its alternating black and white stripes make it prone to high numbers of false matches without additional constraints (such as the uniqueness constraint) with all the simple algorithms studied here. Whilst it challenges matching algorithms, this image is unlikely to represent a real scene, so matching problems with it were not considered indicative of likely problems with real scenes.

3.2.3 Pixel-to-Pixel

Birchfield and Tomasi's Pixel-to-Pixel algorithm is a dynamic algorithm which attempts to find an optimum sequence of moves through the space of disparity *vs* pixel position on a scanline. It uses a cost function which includes κ_r - a reward for a match and κ_{occ} - a penalty for an occlusion. It is the only algorithm in the study that explicitly identify occlusions. The number of occlusions in the images used here is small (4.2% of pixels), so that ability to correctly identify them does not, in itself, add significantly to overall matching performance.



(a) Corridor



(b) Tsukuba

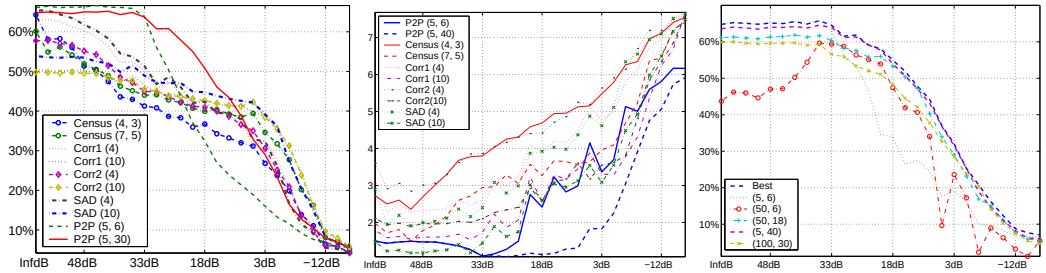
Figure 5. Pixel-to-Pixel: Correct match fraction *vs* cost function parameters, κ_r and κ_{occ}

We measured the performance of the Pixel-to-Pixel algorithm for all of the test images *vs* the two adjustable parameters, κ_r and κ_{occ}, see figure 5. In all image pairs except Madroom, results are relatively insensitive to κ_r and κ_{occ} values with two images showing slight degradations at large κ_{occ} values.

The values giving best results for the Corridor images ($\kappa_{occ} = 5$, $\kappa_r = 6$) were initially chosen for the noise experiments; they give near optimal performance (degradations of a few percent only) with other images. Note that this differs from Birchfield and Tomasi's original suggestion - ($\kappa_{occ} = 25$, $\kappa_r = 5$)[2] - which they appear to have derived empirically based on the need to penalize long occlusion sequences[1]. As before, the Madroom images are the exception; they show a much lower number of correct matches and the shape of the surface is quite different. Clearly additional constraints must be added to the Pixel-to-Pixel algorithm for images of this type also.

4 Effect of Noise

Using optimum parameters chosen from noise-free images ($\kappa_r = 5, \kappa_{occ} = 6$), the Pixel-to-Pixel algorithm is very robust to low noise levels, showing little degradation for SNR > 36dB. Beyond this point, it degrades rapidly becoming similar to the other algorithms for SNR ~ 24dB - see figure



(a) Fraction of good matches (b) Standard Deviation (c) P2P: Effect of (κ_{occ}, κ_r) values

Figure 6. Effect of noise

'InfdB' labels the 'perfect' image generated by the ray tracer. Parameter values for each algorithm follow its name in brackets: they are: Census (α, β); SAD, Corr1, Corr2 (window radius); Pixel-to-Pixel(κ_{occ}, κ_r) In (c), the curve labelled 'Best' shows the percentage of good matches obtainable at each noise level with the optimum choice for κ_{occ} and κ_r at that level.

6(a). Since this algorithm shows very little sensitivity to the actual values of κ_r and κ_{occ} for the synthetic and noise-free Corridor images (figure 5(a)) and for the apparently low-noise Tsukuba images (figure 5(b)), we checked the effect of noise for selected values of κ_r and κ_{occ} with the results shown in figure 6. *The optimum values for perfect images do not perform well in the presence of noise*: larger values of κ_r perform better. Using ($\kappa_{occ} = 5$, $\kappa_r = 40$) has negligible effect on performance for noise-free images but substantially improves the performance with noise present - see figure 6 which compares algorithms. Pixel-to-Pixel retains its advantage in the presence of noise *as long as good values of κ_r and κ_{occ} are chosen*. This casts doubt on the proposition[2] that the occlusion penalty must prevent failure to match long sequences of pixels due to noise: in our experiments larger κ_r values than those which are effective for noise-free (Corridor) or relatively noise-free images (Tsukuba) are needed.

Of the correlation-style algorithms, the simplest (SAD) tolerates noise as well as any. Its performance degrades steadily as noise is added and exceeds that of Pixel-to-Pixel for images with SNR $\leq$ 15 dB. For high levels of noise, SAD's aggregation over a window surrounding the pixel of interest (Pixel-to-Pixel uses only information from the current scanline) allows it to recover useful information.

The Census algorithm which has good matching rates at low noise degrades very quickly as noise is added. This is certainly due to the ordering relation - small amounts of noise can cause a large number of bits to flip in the transforms - particularly in low contrast areas. However with large windows, Census shows reasonable performance for noisy images: its performance comes close to that of the simpler and faster correlation algorithms.

Correlation algorithms with large windows consistently produce narrow distributions of disparity errors over all noise values; Pixel-to-Pixel also consistently produces narrow distributions - see figure 6 (b).

Algorithm	Parameters	Time (sec)	Complexity per point
Census	$\alpha = 4, \beta = 3$	58.4	$\mathcal{O}(\alpha^2\beta^2\Delta)$
Census	$\alpha = 7, \beta = 5$	361	
Corr1	$w = 4$	3.9	$\mathcal{O}(w^2\Delta)$
	$w = 10$	20.1	
Corr2	$w = 4$	3.4	$\mathcal{O}(w^2\Delta)$
	$w = 10$	16.6	
SAD	$w = 4$	2.0	$\mathcal{O}(w^2\Delta)$
	$w = 10$	9.6	
Pixel-to-Pixel	$\forall(\kappa_{occ}, \kappa_r)$	1.8	$\mathcal{O}(\Delta^2)$

Table 2. Running Times (w - window radius; Δ - maximum disparity)

5 Timing

Our tests show similar matching performance for several algorithms, so we timed their execution *cf.* table 2.

Clearly, Pixel-to-Pixel is significantly faster than all the others - as expected from the time complexities. The correlation algorithms are fast for small windows but degrade significantly ($\mathcal{O}(w^2)$) with the window size. Census is extremely slow: it has to work over two windows - small windows are not effective and times increase quadratically with both windows. However the Census algorithm's potential for efficient hardware implementation has led us to study some improvements - targeted at hardware applications[6].

6 Conclusion

It was clear that Birchfield and Tomasi's Pixel-to-Pixel algorithm, a dynamic algorithm, shows the best performance of the area-based algorithms examined in this work: it provides the same or better correctly matched pixel fractions, is significantly faster and tolerates noise, with negligible degradation in performance up to a SNR of 36dB. Below this level, matching performance degrades faster, but is still superior to any of the correlation style algorithms until very high levels of noise (SNR $\leq 15dB$) are present.
Simple correlation algorithms performed similarly and there is little reason to prefer any one over the simplest sum-of-absolute differences (SAD) one. Normalizing the signal intensity may increase performance when the images are poorly matched for intensity, but this was not evident in the 'real' images studied here: we contend that it is preferable and relatively easy to calibrate the cameras themselves either electronically or by pre-processing software.
We have started work on Census algorithm improvements: despite relatively poor performance, their potential for efficient hardware implementations makes them attractive.

A Noise

Noise is usually measured by a signal to noise ratio (SNR) expressed in dB and defined as (where P is the power):

$$SNR = 10 \cdot \log_{10}\left(\frac{P_{signal}}{P_{noise}}\right) \quad (1)$$

An SNR of 0dB implies equal signal and noise powers: increasing an SNR by 3dB doubles the signal's power. For a discrete image, in which I_p is the intensity of a pixel at p, the power is:

$$P_{signal} = \frac{1}{n_{pixels}} \cdot \sum_{image} I_p^2$$

If the noise is assumed to have a Gaussian distribution, $N(\mu, \sigma)$, and the Gaussian is centred ($\mu = 0$), the power of the noise is:

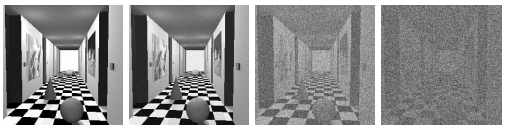
$$P_{noise} = Var(N(0, \sigma)) = \sigma^2 \quad (2)$$

For an image with power, P_{signal}, to produce a given SNR, we add noise with standard deviation:

$$\sigma = \sqrt{Var} = \sqrt{\frac{P_{signal}}{10^{\frac{SNR}{10}}}} \quad (3)$$

To generate noise, calculate the normal centred deviate $N(0,1)$[2] which relies upon a uniform random deviate[3].
Convert $N(0,1)$ to $N(\mu, \sigma)$ using:

[2] Using the `gasdev` function - chapter 7.2, page 289 [8]
[3] `ran1` function - chapter 7.1, page 280 [8]



No noise SNR = 24dB SNR = 0db SNR = -12dB

Figure 7. Images with varying SNR's
if X follows $N(0,1)$ then $a \cdot X + b$ follows $N(b, a^2)$

Figure 7 shows images with increasing levels of noise. At SNR=-12dB the image contains little valid information. To place this noise definition in context, observe that SNR=+24dB produces images which appear similar to the 'perfect' one, but this SNR implies noise with $\sigma \sim 9$ and therefore noise values of $\sim 6\%$ of signal values with a mean intensity of ~ 153. This represents a large error, $\sim 4\%$ (6 in 153). Thus a good camera in a well-lighted scene would produce SNR's of 40dB or more.

References

[1] S. Birchfield and C. Tomasi. Depth discontinuities by pixel-to-pixel stereo. Technical Report CS-TR-96-1573, Stanford University, Department of Computer Science, July 1996.
[2] S. Birchfield and C. Tomasi. Depth discontinuities by Pixel-to-Pixel stereo. In S. Chandran and U. Desai, editors, *Procedings of the Sixth International Conference on Computer Vision (ICCV-98)*, pages 1073–1080, New Delhi, Jan. 4–7 1998. Narosa Publishing House, Narosa Publishing House.
[3] C. Davey. *Feature-based Stereo Matching*. BE(Hons) Thesis, University of Western Australia, 2002.
[4] O. Faugeras *et al.*. Real time correlation-based stereo: algorithm, implementatinos and applications. Technical Report 2013, INRIA, 06902 Sophia Antipolis, France, 1993.
[5] V. Gerdes. *Modular Rendering Tools*. `www-student.informatik.uni-bonn.de/~gerdes/MRTStereo/index.html`, 2001.
[6] P. Leclercq. *Stereo Matching Algorithms*. PhD thesis, University of Western Australia, 2003.
[7] J. Morris. Reconfigurable computing. In V. G. Oklobdzija, editor, *Computer Engineering Handbook*, pages 37–1 – 37–16. CRC Press, CRC Press, 2001.
[8] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. University of Cambridge, 1992.
[9] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Intl Jnl of Computer Vision*, 47:7–42, Apr.–June 2002.
[10] R. Zabih and J. Woodfill. Non-parametric local transforms for computing visual correspondence. *Lecture Notes in Computer Science*, 800:151–158, 1994.

Bibliography

[1] Doo Huyn Lee, In So Kweon, and Roberto Cipolla, "A biprism-stereo camera system," IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'99), pp. 82–87, 1999.

[2] Zhong-Dan Lan and Roger Mohr, "Robust matching by partial correlation," Tech. Rep. RR-2643, INRIA, August 1995.

[3] Andreas Koschan, Volker Rodehorst, and Kathrin Spiller, "Color stereo vision using hierarchical block matching and active colour illumination," in 13th International Conference on Pattern Recognition (ICPR'96), Vienna, Austria, August 1996, pp. 835–839.

[4] Andreas Koschan and Volker Rodehorst, "Dense depth maps by active color illumination and image pyramids," in Advances in Computer Vision, R. Klette F. Solina, W.G. Kropatsch and R.Bajcsy, Eds., Vienna, Austria, 1997, pp. 137–148, Springer.

[5] Ingemar J. Cox, "A maximum likehood n-camera stereo algorithm," in IEEE International Conference on Pattern Recognition (ICPR'94), 1994, pp. 437–443.

[6] Ingemar J. Cox, Sunita L. Hingorani, and Satish B. Rao, "A maximum likehood stereo algorithm," Computer Vision and Image Understanding, vol. 63, no. 3, pp. 542–567, May 1996.

[7] Sébastien Roy, Jean Meunier, and Ingemar J. Cox, "Cylindrical rectification to minimize epipolar distortion," in IEEE Proceedings of International Conference on Computer Vision and Pattern Recognition (CVPR'97), Puerto Rico, June 1997, pp. 393–399.

[8] Sébastien Roy and Ingemar J. Cox, "A maximum-flow formulation of the n-camera stereo correspondence problem," in IEEE Proceedings of International Conference on Computer Vision (ICCV'98), Bombay, 1998.

[9] Olivier Faugeras and Stéphane Laveau, "Representing three-dimensional data as a collection of images and fundamental matrices for image synthesis," in Proceedings of International Conference on Pattern Recognition (ICPR'94), 1994, pp. 689–691, Also INRIA Research Report 2205.

[10] Frédéric Devernay and Olivier Faugeras, "Computing differential properties of 3-D shapes from stereoscopic images without 3-D models," in Proceedings of (CVPR'94), 1994, pp. 208–213, Also INRIA Research Report 2304.

[11] Olivier Faugeras, "Stratification of 3-D vision: Projective, affine, and metric representations," Journal of the Optical Society of America A, vol. 12, no. 3, pp. 465–484, March 1995.

[12] Olivier Faugeras, Stéphane Laveau, Luc Robert, Gabriella Csurka, and Cyril Zeller, "3-D reconstruction of urban scenes from sequences of images," in Automatic Extraction of Man-Made Objects from Aerial and Space Images, O. Kuebler A. Gruen and P. Agouris, Eds. June 1995, Birkhauser, Also INRIA Research Report 2572.

[13] Luc Robert, Cyril Zeller, Olivier Faugeras, and Martial Hebert, "Applications of non-metric vision to some visually-guided robotics tasks," in Visual Navigation, Yiannis Aloimonos, Ed. June 1995, LEA, also INRIA Research Report 2584.

[14] Olivier Faugeras and Luc Robert, "What can two images tell us about a third one?," International Journal of Computer Vision (IJCV), vol. 18, no. 1, pp. 5–20, April 1996, Also INRIA Research Report 2018.

[15] Quang-Tuan Luong and Olivier Faugeras, "Camera calibration, scene motion and structure recovery from point correspondences and fundamental matrices," International Journal of Computer Vision (IJCV), vol. 22, no. 3, pp. 261–289, 1997.

[16] Olivier Faugeras, Luc Robert, Stéphane Laveau, Gabriella Csurka, Cyril Zeller, Cyrille Gauclin, and Imad Zoghlami, "3-D reconstruction of urban scenes from image sequences," in CVGIO-IU, 1998.

[17] Georgy Gimel'farb and Hao Li, "Experiments in probabilistic regularisation of symmetric dynamic programming stereo," in Proceeding APRS/IEEE Workshop on Stereo Image and Video Processing, Sydney, Australia, December 2000, pp. 29–32.

[18] Georgy Gimel'farb, "Probabilistic regularisation and symmetry in binocular dynamic programming stereo," Pattern Recognition Letters, vol. 23, no. 4, pp. 431–442, 2002.

[19] Hiroshi Ishikawa and Davi Geiger, "Occlusions, discontinuities, and epipolar lines in stereo," in Fifth European Conference on Computer Vision (ECCV'98), Freiburg, June 1998.

[20] C. Lawrence Zitnick and Takeo Kanade, "A volumetric iterative approach to stereo matching and occlusion detection," Tech. Rep. CMU-RI-TR-98-30, The Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213, Decembre 1998.

[21] Simon Baker, Terence Sim, and Takeo Kanade, "Characterization of inherent stereo ambiguities," in 8th International Conference on Computer Vision (ICCV'01), Vancouver, British Columbia, July 2001.

[22] D. Marr and T. Poggio, "A computational theory of human stereo vision," Royal Society of London, vol. 204, no. 1156, pp. 301–328, 1979.

[23] M. El Ansari, L. Masmoudi, and L. Radouane, "A new region matching method for stereoscopic images," Pattern Recognition Letters, vol. 21, no. 4, pp. 283–294, 2000.

[24] Mohammed Zaki, Abdallah El-Ramsisi, and Rostom Omran, "A genetic aggregate stereo algorithm for 3-D classification of occluded shapes," Pattern Recognition Letters, vol. 21, no. 5, pp. 349–363, 2000.

[25] G. McGunnigle and M.J. Chantler, "Rough surface classification using point statistics from photometric stereo," Pattern Recognition Letters, vol. 21, no. 6-7, pp. 593–604, 2000.

[26] A. Bigand, T. Bouwmans, and J.P. Dubus, "A new stereomatching algorithm based on linear features and the fuzzy integral," Pattern Recognition Letters, vol. 22, no. 2, pp. 133–146, 2001.

[27] Gonzalo Pajares and Jesús M. de la Cruz, "Local stereovision matching through the adaline neural network," Pattern Recognition Letters, vol. 22, no. 14, pp. 1457–1473, 2001.

[28] Jean Louchet, Maud Guyon, Marie-Jeanne Lesot, and Amine Boumaza, "Dynamic flies: a new pattern recognition tool applied to stereo sequence processing," Pattern Recognition Letters, vol. 23, no. 1-3, pp. 335–345, 2002.

[29] C.V. Jawahar and P.J. Narayanan, "An adaptive multifeature correspondence algorithm for stereo using dynamic programming," Pattern Recognition Letters, vol. 23, no. 5, pp. 549–556, 2002.

[30] Gonzalo Pajares and Jesús M. de la Cruz, "Stereovision matching through support vector machines," Pattern Recognition Letters, vol. 24, no. 15, pp. 2575–2583, 2003.

[31] Shengsheng Yu Qiuming Luo, Jingli Zhou and Degui Xiao, "Stereo matching and occlusion detection with integrity and illusion sensitivity," Pattern Recognition Letters, vol. 24, pp. 1143–1149, 2003.

[32] Kyu-Phil Han, Kun-Woen Song, Eui-Yoon Chung, Seok-Je Cho, and Yeong-Ho Ha, "Stereo matching using genetic algorithm with adaptive chromosomes," Pattern Recognition, vol. 34, pp. 1729–1740, 2001.

[33] Ramin Zabih and John Woodfill, "Non-parametric local transforms for computing visual correspondence," Lecture Notes in Computer Science (LNCS), vol. 800, pp. 151–158, 1994.

[34] Vladimir Kolmogorov and Ramin Zabih, "Computing visual correspondence with occlusions via graph cuts," in International Conference on Computer Vision (ICCV'01), July 2001.

[35] Vladimir Kolmogorov and Ramin Zabih, "What energy functions can be minimized via graph cut?," European Conference on Computer Vision (ECCV'02), pp. 65–81, 2002.

[36] Vladimir Kolmogorov and Ramin Zabih, "Multi-camera scene reconstruction via graph cuts," in In European Conference on Computer Vision (ECCV'02), A. Heyden et al., Ed., 2002, number 2352 in Lecture Notes in Computer Science (LNCS), pp. 82–96.

[37] Vladimir Kolmogorov, Ramin Zabih, and Steven Gortler, "Generalized multi-camera scene reconstruction using graph cuts," in In Fourth International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR'03), July 2003, pp. 501–516.

[38] Junhwan Kim, Vladimir Kolmogorov, and Ramin Zabih, "Visual correspondence using energy minimization and mutual information," in In International Conference on Computer Vision (ICCV'03), October 2003, pp. 508–515.

[39] Vladimir Kolmogorov and Ramin Zabih, "What energy functions can be minimized via graph cut?," IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), vol. 28, no. 2, pp. 147–159, February 2004.

[40] R.A. Lane and N.A. Thacker, "Stereo vision research: An algorithm survey," *citeseer. ist. psu. edu/ lane96stereo. html*, January 1996.

[41] Andreas Koschan, "A framework for area-based and feature-based stereo vision," Machine Graphics and Vision, vol. 2, no. 4, pp. 285–308, 1993.

[42] Daniel Scharstein and Richard Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," Tech. Rep. MSR-TR-2001-81, Microsoft Research, Microsoft Corporation, One Microsoft Way, Redmond, WA 98052, November 2001.

[43] Daniel Scharstein and Richard Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," International Journal of Computer Vision, vol. 47, pp. 7–42, Apr.–June 2002.

[44] Daniel Scharstein, Richard Szeliski, and Ramin Zabih, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," in Workshop on Stereo and Multi-Baseline Vision (in conjunction with IEEE CVPR'01), Kauai, Hawaii, December 2001, pp. 131–140.

[45] Giovanni Garibotto, Marco Corvi, Carlo Cibei, and Sara Sciarrino, "3DMODS 3D moving obstacle detection system," in Proceedings of the International Conference on Image Analysis and Processing (ICIAP'03), IEEE Computer Society, Ed., September 2003, pp. 618–623.

[46] Peter Corke, "Real-time stereo vision for height estimation on an autonomous helicopter," in Proceeding APRS/IEEE Workshop on Stereo Image and Video Processing, Sydney, Australia, December 2000, pp. 19–22.

[47] Catherine Davey, "Feature-based stereo matching," BE(Hons) Thesis, University of Western Australia, 2002.

[48] Gang Xu and Zhengyou Zhang, Epipolar Geometry in Stereo, Motion and Object Recognition, A Unified Approach, Computational Imaging and Vision. Kluwer Academics, 1996.

[49] Mark Chan, Chia-Yen Chen, Gareth Barton, Patrice Delmas, Georgy Gimel'farb, Philippe Leclercq, and Thomas Fischer, "A strategy for 3D face

analysis and synthesis," in Proceedings of the International Conference on Image and Vision Computing New Zealand (IVCNZ'03), Donald G. Bailey Editor, Ed., Massey University, Palmerston North, New Zealand, November 2003, pp. 384–389.

[50] Olivier Faugeras and Quang-Tuan Luong, "The fundamental matrix: Theory, algorithms and stability analysis," International Journal of Computer Vision (IJCV), vol. 17, no. 1, pp. 43–76, 1996.

[51] Donald G. Bailey, "A new approach to lens distortion correction," in Proceedings of the International Conference on Image and Vision Computing New-Zealand (IVCNZ'02), David Kenwright, Ed., December 2002, pp. 59–64.

[52] Robert J. Valkenburg and Peter L. Evans, "Lens distortion calibration by straightening lines," in Proceedings of the International Conference on Image and Vision Computing New-Zealand (IVCNZ'02), David Kenwright, Ed., December 2002, pp. 59–64.

[53] Hanspeter A. Mallot, "Stereopsis - geometrical and global aspects," in Handbook of Computer Vision and Applications, Bernd Jähne, Horst Haußecker, and Peter Geißler, Eds., vol. 2, chapter 17, pp. 485–504. Academic Press, 1999.

[54] Robert C. Weast, Ed., Handbook of Chemistry and Physics, CRC Press, April 1979.

[55] O. Faugeras, B. Hotz, H. Mathieu, T. Viéville, Z. Zhang, P. Fua, E. Théron, L. Moll, G. Berry, J. Vuillemin, P. Bertin, and C. Proy, "Real time correlation-based stereo: algorithm, implementations and applications," Tech. Rep. 2013, Institut National De Recherche en Informatique et en Automatique (INRIA), 06902 Sophia Antipolis, France, 1993.

[56] V. Carl Hamacher, Zvonko G. Vranesic, and Safwat G. Zaky, Computer Organization, McGraw-Hill Computer Science Series. McGraw-Hill, 4th edition, 1996.

[57] Ramin Zabih and John Woodfill, "A non-parametric approach to visual correspondence," Submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence,
citeseer. ist. psu. edu/46052. html.

[58] John Morris, "Reconfigurable computing," in Computer Engineering Handbook, Vojin G. Oklobdzija, Ed. CRC Press, 2001, pp. 37–1 – 37–16, CRC Press.

[59] John Woodfill, Brian von Herzen, and Ramin Zabih, "Frame-rate robust stereo on a PCI board,
citeseer. ist. psu. edu/ 253389. html," .

[60] John Woodfill and Brian Von Herzen, "Real-time stereo vision on the PARTS reconfigurable computer," in IEEE Symposium on FPGAs for Custom Computing Machines, Kenneth L. Pocek and Jeffrey Arnold, Eds., Los Alamitos, CA, 1997, pp. 201–210, IEEE Computer Society Press.

[61] Stan Birchfield and Carlo Tomasi, "Depth discontinuities by pixel-to-pixel stereo," Technical Report CS-TR-96-1573, Stanford University, Department of Computer Science, July 1996.

[62] Stan Birchfield and Carlo Tomasi, "Depth discontinuities by Pixel-to-Pixel stereo," in Proceedings of the Sixth International Conference on Computer Vision (ICCV-98), Sharat Chandran and Uday Desai, Eds., New Delhi, Jan. 4–7 1998, Narosa Publishing House, pp. 1073–1080, Narosa Publishing House.

[63] Stan Birchfield and Carlo Tomasi, "A pixel dissimilarity measure that is insensitive to image sampling," IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), vol. 20, no. 4, pp. 401–406, April 1998.

[64] Daniel Scharstein and Richard Szeliski, "Stereo matching with non-linear diffusion," Tech. Rep. TR96-1575, Cornell University, Department of Computer Science, 5151 Upson Hall, Ithaca, NY 14853-7501, 18, 1996.

[65] Daniel Scharstein and Richard Szeliski, "Stereo matching with non-linear diffusion," in IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'96), San Francisco, June 1996, pp. 343–350.

[66] Daniel Scharstein and Richard Szeliski, "Stereo matching with non-linear diffusion," International Journal of Computer Vision (IJCV), vol. 28, pp. 155–174, 1998.

[67] Pascal Fua, "Combining stereo and monocular information to compute dense depth maps that preserve depth discontinuities," in Proceedings of the 12th International Joint Conference on Artificial Intelligence, Sydney, New South Wales, Australia, August 1991, pp. 1292–1298.

[68] Pascal Fua, "A parallel stereo algorithm that produces dense depth maps and preserves image features," Tech. Rep. 1369, INRIA, 1993.

[69] Takeo Kanade and Masatoshi Okutomi, "A stereo matching algorithm with an adaptive window: Theory and experiment," IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), vol. 16, no. 9, pp. 920–932, September 1994.

[70] Georgy Gimel'farb, "Stereo terrain reconstruction by dynamic programming," in Handbook of Computer Vision and Applications, Bernd Jähne, Horst Haußecker, and Peter Geißler, Eds., vol. 2, chapter 18, pp. 505–530. Academic Press, 1999.

[71] Georgy Gimel'farb, "Binocular stereo by maximising the likelihood ratio relative to a random terrain," in International Workshop Robot Vision '01, Reinhard Klette, Shmuel Peleg, and Gerald Sommer, Eds., Auckland, New-Zealand, February 2001, number 1998 in Lecture Notes in Computer Science (LNCS), pp. 201–208.

[72] G. Gimel'farb and U. Lipowezky, "Accuracy of computational stereo: symmetric dynamic programming versus correlation," in Proceedings of the 2nd Pattern Recognition in Remote Sensing Workshop, BMVA Press, Ed., Niagara Falls, Canada, August 2002, pp. 49–56.

[73] Yuri Boykov, Olga Veksler, and Ramin Zabih, "Fast approximate energy minimization via graph cuts," in ICCV 1999, September 1999.

[74] Yuri Boykov, Olga Veksler, and Ramin Zabih, "Fast approximate energy minimization via graph cuts," IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), vol. 23, no. 11, pp. 1222–1239, November 2001.

[75] C. Lawrence Zitnick and Takeo Kanade, "A cooperative algorithm for stereo matching and occlusion detection," Tech. Rep. CMU-RI-TR-99-35, The Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213, October 1999.

[76] C. Lawrence Zitnick and Takeo Kanade, "A cooperative algorithm for stereo matching and occlusion detection," IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), vol. 22, no. 7, pp. 675–684, July 2000, Earlier version appears as technical report CMU-RI-TR-99-35.

[77] Changming Sun, "Fast stereo matching using rectangular subregioning and 3d maximum-surface techniques," International Journal of Computer Vision (IJCV), vol. 47, no. 1/2/3, pp. 99–117, May 2002.

[78] R. Klette, A. Koschan, K. Schlüns, and V. Rodehorst, "Surface reconstruction based on visual information," Tech. Rep. 95/6, The University of Western Australia, July 1995.

[79] W. Fellenz, K. Schlüns, A. Koschan, and M. Teschner, "An active vision system for obtaining high resolution depth information," in Proceedings of the 7th International Conference on Computer Analysis of Images and Patterns (CAIP'97), J.Pauli G. Sommer, K. Daniilidis, Ed., Kiel, Germany, September 1997, pp. 726–733.

[80] K. Schlüns, W. Fellenz, A. Koschan, and M. Teschner, "A modular 10-DOF vision system for high-resolution active stereo," Tech. Rep. CITR-TR-2/CS-TR-141, The University of Auckland, Technical University of Berlin and University of Erlangen-Nuremberg, 1997.

[81] Andreas Koschan, "Using perceptual attributes to obtain dense depth maps," Proceedings of the IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI'96), pp. 155–159, April 1996.

[82] Y.-I. Ohta, T. Kanade, and T. Sakai, "Color information for region segmentation," in Computer Graphics and Image Processing (CGIP'80), 1980, vol. 13, pp. 222–241.

[83] Andreas Koschan and Volker Rodehorst, "Towards real-time stereo employing parallel algorithms for edge-based and dense stereo matching," in Proceedings of the IEEE Workshop on Computer Architectures for Machine Perception (CAMP'95), Como, Italy, September 1995.

[84] Andreas Koschan, "Improving robot vision by color information," in Proceedings of the 7th International Conference on Artificial Intelligence and Information-Control Systems of Robots, I. Plander (Hrsg.), Ed., September 1997, pp. 247–258.

[85] John R. Jordan III and Alan C. Bovik, "Computational stereo vision using color," IEEE Control Systems Magazine, pp. 31–36, June 1988.

[86] John R. Jordan III and Alan C. Bovik, "Using chromatic information in edge-based stereo correspondence," Computer Vision, Graphics and Image

Processing (CVGIP): Image Understanding, vol. 54, no. 1, pp. 98–118, July 1991.

[87] S. B. Pollard, J. E. W. Mayhew, and J. P. Frisby, "Pmf: A stereo correspondence algorithm using a disparity gradient limit," in Perception, 1985, vol. 14, pp. 449–470.

[88] Karsten Mülmann, Dennis Maier, Jürgen Hesser, and Reinhard Männer, "Calculating dense disparity maps from color stereo images, an efficient implementation," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR'01), Kauai Marriott, Hawaii, December 2001.

[89] K. Mühlmann, D. Maier, J. Hesser, and R. Männer, "Calculating dense disparity maps from color stereo images, an efficient implementation," International Journal of Computer Vision (IJCV), vol. 47, no. 1, pp. 79–88, April 2002.

[90] John L. Smith, "Implementing median filters in xc4000e FPGAs," Xcell: The Quaterly Journal for Xilinx Programmable Logic Users, vol. 23, 1996.

[91] Daniel Scharstein and Richard Szeliski, "High-accuracy stereo depth maps using structured light," in IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'03), Madison, WI, June 2003, vol. 1, pp. 195–202.

[92] Sing Bing Kang, J.A. Webb, Charles Zitnick, and Takeo Kanade, "A multibaseline stereo system with active illumination and real-time image acquisition," in Proceedings of the Fifth International Conference on Computer Vision (ICCV'95), 1995, pp. 88–93.

[93] Petteri Kettunen Frans Slothouber and Jacco van Weert, "Robodoc 4.0.6," *www.xs4all.nl/~rfsber/Robo/robodoc.html*, July 2003.

[94] Volker Gerdes, "Modular rendering tools," *www-student.informatik.uni-bonn.de/~gerdes/MRTStereo/index.html*, 2001.

[95] Reserve Bank of New Zealand, "New Zealand coinage specifications," *www.rbnz.govt.nx/currency/money/0101459.html*, 2003.

[96] Guy Parsons, "Digital camera image sensor," *homepages.ihug.com.au/~parsog/photo/sensors1.html*, 2003.

[97] Philippe Leclercq and John Morris, "Assessing stereo algorithm accuracy," in Proceedings of the International Conference on Image and Vision Computing New Zealand (IVCNZ'02), David Kenwright, Ed., Auckland, New Zealand, December 2002, pp. 89–93.

[98] Philippe Leclercq and John Morris, "Robustness to noise of stereo matching," in Proceedings of the International Conference on Image Analysis and Processing (ICIAP'03), IEEE Computer Society, Ed., Mantova, Italy, September 2003, pp. 606–611.

[99] TopoSys GmbH, "Topographische systemdaten," *www. toposys. com*, 1995.

[100] TopoSys GmbH, "Falcon, lidar sensor system," *www. toposys. com/ pdf-ext/ Engl/ Falcon_ Folder_ Mar_ 2004. pdf*, March 2004.