# An investigation into stereo algorithms: An emphasis on local-matching

Thulani Ndhlovu

Submitted to the Department of Electrical Engineering,
University of Cape Town, in fullfillment of the requirements
for the degree of Master of Science in Engineering.
March 2011

## Declaration

I declare that this dissertation is my own work. It is being submitted for the degree of Master of Science in Engineering at the University of Cape Town. It has not been submitted before for any degree or examination at this or any other institution.

.....
Thulani Ndhlovu
(Candidate's Signature)

## Acknowledgments

I would like to thank the following people and institutions for their contribution towards this thesis

- Dr Fred Nicolls and Deon Sabatta for their guidance and help.
- The Council for Scientific and Industrial Reasearch, Mobile Intelligent Autonomous Systems unit for their financial assistance.
- My family and friends for their support.

## Abstract

Stereo vision attempts to reconstruct the 3D structure of a scene given two images. The main difficulty with stereo vision is the correspondence problem. Local-matching stereo algorithms are an attractive solution to this problem because they have a uniform structure and can be paralellized. This makes them applicable in real-time systems because they can be implemented on graphics processing units (GPUs) and field programmable gate arrays (FPGAs). This thesis closely analyses local-matching algorithms and explores two issues.

The first issue is that of using temporal seeding in stereo image sequences. In a stereo image sequence, finding feature correspondences is normally done for every frame without taking temporal information into account. Reusing previous computations can add valuable information. A temporal seeding technique is developed for reusing computed disparity estimates on features in a stereo image sequence to constrain the disparity search range. Features are detected on a left image and their disparity estimates are computed using a local-matching algorithm. The features are then tracked to a successive left image of the sequence and by using the previously calculated disparity estimates, the disparity search range is constrained. Errors between the local-matching and the temporal seeding algorithms are analysed on a short and long dataset. Results show that although temporal seeding suffers from error propagation, a decrease in computational time of approximately 20% is obtained when it is applied on 87 frames of a stereo sequence.

The second issue is that of developing a confidence measure for local-matching stereo algorithms. A confidence measure is developed and applied to individual disparity estimates in local-matching stereo correspondence algorithms. It aims at identifying textureless areas, where most local-matching algorithms fail. The confidence measure works by analyzing the correlation curve produced during the matching process. The measure is tested by developing an easily parallelized local-matching algorithm, and is used to filter out unreliable disparity estimates. Using the Middlebury dataset and the developed evaluation scheme, the results show that the confidence measure significantly decreases the disparity estimate errors at a low computational overhead. Furthermore, the confidence measure is used to improve start-up disparities in temporal seeding. Results show that the measure does not succeed in filtering out features producing high errors.

# Contents

# List of Figures

# List of Tables

Chapter 1

# Introduction

Computer vision is the field of study that concerns extracting descriptions of the world from images or sequences of images. This field is attractive because passive imaging sensors such as cameras are relatively cheap compared to alternative sensors such as laser range finders. We are interested in the stereo vision aspect of computer vision.

In stereo vision we attempt to reconstruct the 3D structure of a scene given two images. The main difficulty with stereo vision is the correspondence problem stated as follows: given two images of the same scene from slightly different viewpoints, find corresponding pixels and the disparity, the distance by which the pixel in one view is translated relative to its corresponding pixel in the other view. Solving the correspondence problem results in a 2.5D representation of the scene as shown in Figure 1.1. To recover the 3D coordinates, the corresponding pixels are triangulated.

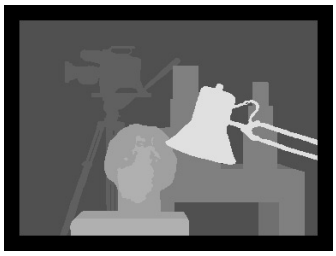
The process of retrieving 3D data from stereo cameras may appear simple at first, because as humans we have two eyes and we are able to perceive 3D data naturally. It turns out that for a computer this is a complex task, and determining correspondences between pixels is a challenging problem. Some of the challenges are caused by image variations between the two views of the scene. These differences might be caused by occlusions of objects, specular reflections and sensor noise.

A large number of stereo algorithms have been proposed to solve the stereo correspondence problem. However, the problem is ill-posed and a satisfying solution has not yet been reached [2, 3]. Nevertheless, many algorithms exist and can be useful depending on the application.

The fact that stereo vision uses cameras, which are passive sensors, makes it an attractive solution to capturing 3D data. While other sensors are available to capture 3D data, cameras are comparatively less expensive and produce a large volume of information. Applications for stereo vision include mobile robotics where a robot perceives the environment in order to navigate itself while avoiding obstacles [4]. In such an application, stereo vision is normally used for Simultaneous Localisation and Mapping (SLAM) [5]. In SLAM the robot builds a map and localises itself within the map while building it. Other applications include augmented reality [6] where virtual objects may be inserted into real world scenes, and human computer-

(a) Left image.

(b) Right image.



(c) Ground truth disparity map where objects closer to the camera are represented by the bright color (white) while objects further away are represented by the dark color (black).

Figure 1.1: Two images used in stereo-vision with the ground truth disparity map. The images are from the Middlebury dataset [1].

interaction [7] where a computer needs to recognize the pose or gesture of a subject.

## 1.1 Background

The aim of a stereo vision system is to retrieve depth information of a scene from two images which are taken from slightly different viewpoints. Figure 1.2 shows a block diagram of a stereo vision system. It consists of five main steps: (1) calibrating the cameras; (2) acquiring stereo images; (3) rectifying the stereo images; (4) finding pixel correspondences; and (5) triangulating the pixel correspondences.

The camera calibration stage is two-fold. Firstly, the two cameras are calibrated independently to find their intrinsic and extrinsic parameters. Secondly, the two cameras are calibrated with each other to find the geometric relationship between them.

After calibrating the cameras, images of the real-world can be acquired. The ac-


Figure 1.2: The block diagram of a stereo vision system consisting of five steps: image acquisition, calibration, rectification, stereo correspondence and triangulation.

quisition of stereo images requires the two images to be captured at the same time. This means that the two cameras have to be synchronized.

Image rectification is a pre-processing step for stereo correspondence. This stage makes use of the calibration between the two cameras and epipolar geometry to transform the images so that their scanlines are aligned. This simplifies the stereo correspondence search from 2D to 1D.

The stereo correspondence stage is concerned with finding corresponding pixels between the two stereo images. After finding the corresponding pixels, the disparity of a pixel is determined. Stereo correspondence is a challenging problem in stereo vision. This problem will be given a fair amount of attention.

The last stage is triangulation. The corresponding pixels in the two stereo images are used to find the depth of the scene points by triangulation. Furthermore, the Euclidean coordinates of the scene points can be determined.

## 1.2 Stereo correspondence

As mentioned before, stereo correspondence is a challenging problem. Many algorithms exist to solve this problem [2]. In this section, different stereo correspondence algorithms which form building blocks for much more complex algorithms are explained and compared.

### 1.2.1 Local-matching stereo algorithms

The earliest attempts into solving the stereo correspondence problem involve the use of local-matching algorithms [8, 9]. These algorithms generally use some kind of statistical correlation between colour or intensity patterns in local support windows. By using the local support windows, the image ambiguity is reduced efficiently while the discriminative power of the similarity measure is increased. A common local-matching algorithm is the sum of squared differences (SSD) algorithm.

The SSD algorithm takes a window of size $m \times n$ centered at a pixel of interest in the left image and searches in the right image, along a scanline, for a window with similar intensities. This is shown in Figure 1.3.


(a) Left image. (b) Right image.

Figure 1.3: Stereo images showing the local-matching process.

At each step of the search, the difference $\mathcal{E}$ of the two windows is calculated using the equation

$$\mathcal{E} = \sum_{m,n} (I_L(m,n) - I_R(m,n-d))^2, \tag{1.1}$$

where $I_L$ and $I_R$ are the intensity values of the left and right images respectively and $d$ is the current disparity offset. The pixel in the right image corresponding to the disparity value with the lowest $\mathcal{E}$ is then nominated as the best match for the current pixel in the left image.

In this type of approach finding the correct match can be challenging, especially in weakly-textured areas where there is very little information to distinguish one pixel from the next. It has been shown that by breaking down the algorithms and optimizing the components, they can produce high quality disparity maps [10]. One advantage of these algorithms is that they can be parallelized and implemented on Graphics Processing Units (GPUs) [11] or Field Programmable Gate Arrays (FPGAs) [12] to achieve real-time speeds. This makes these algorithms useful in applications such as mobile robotics.

### 1.2.2 Global optimization

Global optimization methods [13, 14, 15, 16] work by defining an energy function for the whole image. The problem then is to find a disparity $d$ that minimizes some global function. The most commonly used energy function is the Potts model [17],

$$E(d) = E_{data}(d) + E_{smooth}(d). \tag{1.2}$$

The data term, $E_{data}(d)$, measures how well the disparity function $d$ agrees with the input image pair. The smoothness term $E_{smooth}(d)$ encodes the smoothness assumptions made by the algorithm. Once the global energy has been defined, a variety of algorithms can be used to find a (local) minimum. One common algorithm is graph cuts (GC).

Using graph cuts, the stereo correspondence problem now becomes that of trying to find the maximum flow through a 3D graph. $G = (V; E)$ is defined as a directed graph, where $V$ is the set of vertices and $E$ is the set of edges. The set of vertices is based on the set of all possible matches and is defined as

$$V = L \cup \{s, t\}, \tag{1.3}$$

where $s$ is the source and $t$ is the sink. $L$ is defined as

$$L = \{(x, y, d), x \in [0, x_{max}], y \in [0, y_{max}], d \in [0, d_{max}]\}, \tag{1.4}$$

where $x_{max}$ and $y_{max}$ correspond to the width and height of the images and $d_{max}$ to the range of disparities. The set of edges is defined as

$$E = \left\{ \begin{array}{ll} (u, v) \in L \times L & : |u - v| = 1 \\ (s, (x, y, 0)) & : x \in [0, x_{max}] \\ ((x, y, d_{max}), t) & : y \in [0, y_{max}] \end{array} \right\}. \tag{1.5}$$

Figure 1.4 represents the outline of such a graph. The graph is six-connected except at the extremes, $s$ and $t$, and each vertex has a cost associated with it. Every edge of the graph has a flow capacity calculated as a function of the costs of the vertices it connects. The capacities limit the flow from the source to the sink.

A cut through the graph separates the set of vertices $V$ into two parts, the set containing the source $s$ and the set containing the sink $t$. The problem now becomes that of finding the cut through the graph with the highest flow capacity from the source to the sink. The capacity of a cut is the sum of the edge capacities that define the cut. The minimum cut through the graph represents the maximum flow from the source to the sink. The disparity labels leading to a minimum cut are assigned to the pixels.

### 1.2.3 Dynamic Programming

Stereo matching through dynamic programming (DP) [18, 19, 20] can be considered a semi-global method. It finds the global minimum for independent scanlines in polynomial time. The stereo problem in this context becomes that of finding the minimum cost path, or path of least resistance, through a cost matrix such as the one shown in Figure 1.5. This matrix is constructed by calculating the differences of all the pixels in the reference scanline and all the pixels of the corresponding scanline over a range of disparity levels.

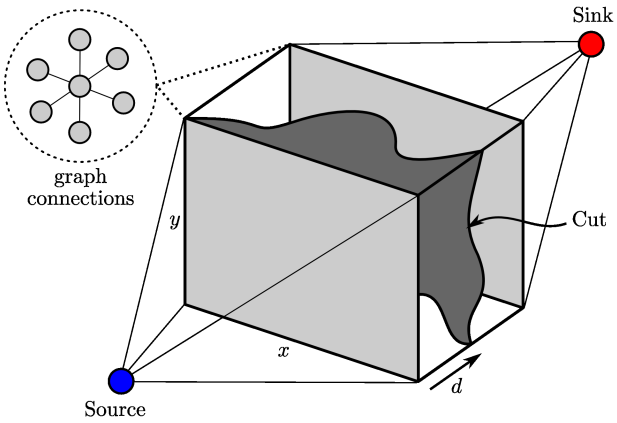



Figure 1.4: Image of the graph used in graph cuts.

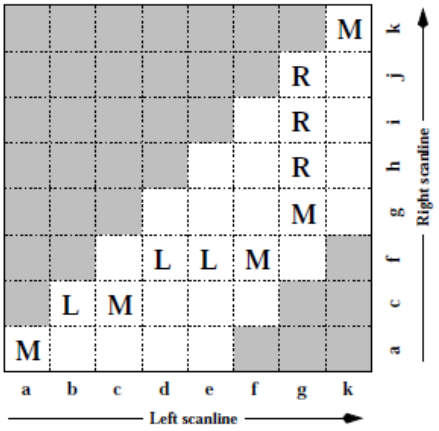



Figure 1.5: Stereo matching using dynamic programming taken from [2]. The letters $(a-k)$ represent the intensities along each scanline. The uppercase letters represent the path selected through the matrix. **M** represents a match, **L** and **R** represent partially occluded points corresponding to points only visible to the left and right images, respectively.

The name dynamic programming comes from a mathematical process through which

a problem is solved by breaking it into smaller pieces. The smaller problems are solved first and the collection of all the smaller problems is equivalent to the problem as a whole. The process is used to find the minimum cost path through a cost matrix derived from the image scanlines. First the possible paths through the cost matrix are defined using some constraints. The main constraint used in DP is the ordering constraint [21]. The cost of a point is defined in a similar manner to the cost in Equation 1.2 but instead of the smoothness term $E_{smooth}(d)$ there is an occlusion term $E_{occlusion}(d)$ to penalize occlusions:

$$E(d) = E_{data}(d) + E_{occlusion}(d). \tag{1.6}$$

The minimum cost to reach a particular point is then calculated and used to calculate the minimum cost of reaching the next point of any path that moves through the first point.

One advantage of global and semi-global methods is that, to a certain degree, they are insensitive to weakly-textured areas. In DP the path will not stray far from the disparities at the edges of the textureless area, because that would increase the cost of the path.

### 1.2.4 Cooperative optimization

Cooperative methods [22, 23] firstly use colour or grayscale information to segment the captured images, and then obtain the initial disparity estimate of the scene by using a known matching algorithm. Finally, a disparity fitting technique is employed to perform the task of disparity refinement for each region. Label-based optimization is used. Furthermore, the algorithms reduce the number of labels by clustering regions in the parameter space of the disparity plane before optimization.

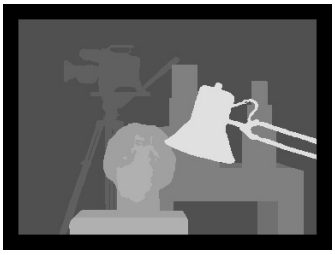
These forms of algorithms are amongst the best performing according to the Middlebury evaluation. However, their main drawback is that they are iterative in nature, making them very slow and inapplicable to real or near real-time systems. Thus, we only discuss them briefly.

### 1.2.5 Comparison

Figure 1.6 shows the depth maps obtained from the Middlebury Tsukuba dataset for the three different approaches described above. The implementation comes from [2]. For local-matching algorithms results of the SSD approach are shown, for semi-global matching results from the DP algorithm are shown, and for global matching results from the GC optimization algorithm are shown.

The SSD algorithm performs the worst out of the three with most errors occurring on the boundaries of the objects, which appear to be bigger than they actually are. This problem is due to the design of local-matching algorithms. Approaches

exist to address this problem [24, 25] at an extra computation cost. The DP algorithm produces better results than SSD but because the scanlines are optimized independently, the algorithm suffers from the streaking effect. The algorithm in [26] has been developed to address this problem at extra computational cost. Amongst the three algorithms, GC performs the best with few errors compared to the other two approaches. The advantage of using global reasoning is effectively demonstrated.



(a) Ground truth disparity map.

(b) Disparity map produced by using the SSD algorithm.

(c) Disparity map produced by using the DP algorithm.

(d) Disparity map produced by using the GC algorithm.

Figure 1.6: Comparison of results obtained by SSD, DP and GC on Tsukuba dataset.

Table 1.1 shows the values for the percentage of bad pixels produced by the algorithms on the Middlebury dataset when using the Middlebury evaluation scheme. The table agrees with the observations of the disparity maps. SSD gives most erroneous results while GC gives the least erroneous results. DP falls between the two other algorithms.

Table 1.2 shows the computation times of the different algorithms on the Middlebury dataset. On average, SSD is the fastest followed by DP and GC has the heaviest computational load. This demonstrates that there is a compromise between

Table 1.1: Table showing the percentage of bad pixels of the SSD, DP and GC algorithms on the Middlebury dataset.

| | SSD | DP | GC |
|---|---|---|---|
| % of bad pixels | 15.7 | 14.2 | 11.4 |

quality and speed in stereo correspondence algorithms. A simple algorithm such as SSD produces the most erroneous results at a low computational cost while a more complex algorithm such as GC produces fewer errors at a relatively higher computational cost.

Table 1.2: Computation times of the SSD, DP and GC algorithms on the Middlebury dataset.

| | Tsukuba | Sawtooth | Venus | Map |
|---|---|---|---|---|
| **Time (seconds)** | | | | |
| SSD | 1.1 | 1.5 | 1.7 | 0.8 |
| DP | 1.0 | 1.8 | 1.9 | 0.8 |
| GC | 23.6 | 43.8 | 51.3 | 22.3 |

## 1.3 Objectives

In this study, we firstly aim to understand the different stereo algorithms available. Then we choose to closely analyze local-matching stereo algorithms to explore the following:

1. *Temporal seeding in stereo image sequences*
   In a stereo image sequence, finding feature correspondences is normally done for every pair of frames without taking temporal information into account. Current imaging sensors can acquire images at high frequencies resulting in small movements between consecutive image frames. Since there are small inter-frame movements, the frame-to-frame disparity estimates do not change significantly. We explore using previously computed disparity estimates to seed the matching process of the current stereo image pair.

   Most conventional stereo correspondence algorithms contain fixed disparity search ranges by assuming the depth range of the scene. This range stays constant throughout a stereo image sequence. By decreasing the disparity search range the efficiency of the matching process can be improved.

2. *Confidence measure for local-matching stereo algorithms*
   Local-matching stereo algorithms generally fail in textureless regions. One way of dealing with these regions is to perform colour segmentation as a pre-processing step [27]. However, this adds a significant overhead on the stereo algorithm which is undesirable for real-time applications.

   A method of assigning a confidence to a disparity estimate for local-matching algorithms is explored. This approach is expected to give low confidences to disparity estimates in textureless regions, where many local-matching algorithms fail. While this approach is similar to a number of previously developed confidence measures [28, 29], in that the confidence of a disparity estimate is a by-product of the matching process, the analysis focuses on the basin of convergence (refer to Figure 6.2) of a disparity estimate. Furthermore, high confidence points are expected to work well in temporal seeding.

## 1.4 Overview of Thesis

This section provides a brief summary of the work being reported on by outlining the contents of the rest of the thesis.

In Chapter 2, the geometry of a canonical camera is briefly discussed. A second camera is introduced and it is shown how to infer depth information. Epipolar geometry and triangulation are also covered.

In Chapter 3 the literature concerning the stereo correspondence problem is covered.

In Chapter 4 local-matching stereo algorithms are discussed in detail. The algorithms are separated into their components and the different design considerations are discussed.

In Chapter 5 temporal seeding on a stereo image sequence is explored. It is shown how previously computed disparity maps can be used to aid the matching process in local-matching stereo algorithms.

In Chapter 6 a confidence to a disparity estimate is assigned and used to filter out errors in local-matching stereo algorithms. Furthermore, the confidence measure is used in temporal seeding.

In Chapter 7 the findings are discussed. Possibilities for future research are mentioned and concluding remarks are made.

Chapter 2

# Stereo Geometry

Stereo geometry refers to the geometric relationship between two cameras. In this chapter the objective is to reach a point where depth information can be inferred given stereo images. The journey begins by describing how to mathematically represent the geometry of a single ideal camera. A second camera is then introduced and the relationship between two camera views is described by using epipolar geometry. The process of image rectification is described and used to simplify epipolar geometry. The chapter is concluded by describing how to use triangulation to determine depth information from stereo images.

This chapter forms a basis for this thesis. Terms commonly used in stereo vision and notation used throughout the document are introduced.

## 2.1 Single view geometry

This section provides a brief summary of the geometry of a single camera.

### 2.1.1 Homogeneous coordinates

Homogeneous coordinates are useful in computer vision and they form a basis for the projective geometry used to project a three-dimensional scene onto a two-dimensional plane. A point $(x, y)$ in 2-D space is represented in homogeneous coordinates by the triple $(kx, ky, k)$. Given a homogeneous point $(kx, ky, k)$, one can transform to the original coordinates by dividing by the scalar $k$. There are two important properties of homogeneous coordinates. Firstly, scalar multiples of a point represent the same point, so $(kx, ky, k)$ is same as the point $(x, y, 1)$ for any non-zero value of $k$. Secondly, points at infinity are represented by the point $(x, y, 0)$, because when transforming the homogeneous coordinates back into the original coordinates we have to divide by zero.

### 2.1.2 Pinhole camera model

The process of mathematically representing a physical system usually begins with constructing an ideal mathematical model. The model is then extended by including deviations which occur in the real world. The most commonly used camera model is the pinhole camera. This model describes the relationship of a 3D point in space to its 2D projection on the image plane.

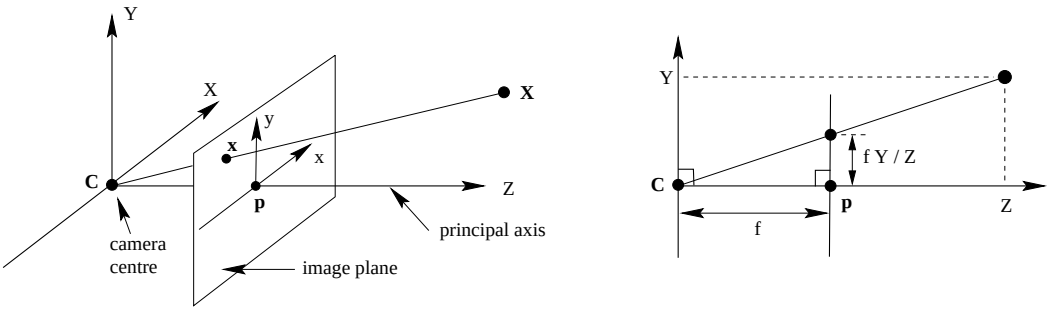



Figure 2.1: Pinhole camera geometry. Projection of a 3D point $\mathbf{X}$ onto point $\mathbf{x}$ on the image plane using a pinhole camera model.

The model assumes that every light ray entering the camera passes through a single point called the camera center, $\mathbf{C}$. The canonical camera center is taken as the origin of the Euclidean coordinate system. The image is produced as the light rays intersect the image plane which lies at a specific distance $f$ from the camera center. This distance $f$ is known as the focal length of the camera. A point $\mathbf{X} = (X, Y, Z)^T$ in $\mathbb{R}^3$ is projected onto a point $\mathbf{x} = (x, y)^T$ on the image plane in $\mathbb{R}^2$ where the ray emanating from $\mathbf{X}$ and passing through the camera center intersects the image plane. Referring to Figure 2.1 it can be shown using similar triangles that

$$(X, Y, Z)^T \mapsto \left( \frac{fX}{Z}, \frac{fY}{Z}, f \right)^T. \tag{2.1}$$

If the world and image points are represented by homogeneous vectors, the projection can be expressed as a linear mapping and can be written as

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}. \tag{2.2}$$

Now, if the world point $\mathbf{X}$ is represented as a homogenous 4-vector $(X, Y, Z, 1)^T$ and the image point $\mathbf{x}$ is represented as a 3-vector, Equation 2.2 can be written as

$$\mathbf{x} = P\mathbf{X}, \tag{2.3}$$

where $P$ is a $3 \times 4$ homogenous camera projection matrix.

### 2.1.3 Extension of the pinhole camera model

In order to better model a real camera, the pinhole camera model is extended [30]. These extensions include the following.

1. *Principal point offset*
   This takes into account the fact that the origin of the camera coordinates might not lie on the camera center.

2. *Distortion correction*
   A pinhole camera is not ideal for making images because it does not gather enough light for rapid exposure. In order to gather more light, cameras use lenses. The disadvantage of using lenses is that they introduce distortions. The two main lens distortions are radial distortions and tangential distortions. These distortions are modelled and corrected on the images.

3. *Pixel shape*
   The pinhole camera model assumes that the image coordinates are Euclidean coordinates having equal scales on both vertical and horizontal directions. In the case of CCD cameras, there is the additional possibility of having non-square pixels. If image coordinates are measured in pixels, then this has the effect of introducing unequal scale factors in each direction.

4. *Extrinsic parameters*
   Extrinsic parameters are the external parameters, which are the position and orientation of the camera in the world coordinates. The camera and world coordinates are related by a Euclidean transformation which gives a rotation and translation.

After extending the pinhole camera model, Equation 2.3 now becomes

$$\mathbf{x} = P\mathbf{X} = K[R|t]\mathbf{X}, \tag{2.4}$$

where $P = K[R|t]$. The matrix $K$ contains the intrinsic parameters of the camera and is called the intrinsic parameter matrix or the calibration matrix. $R$ and $t$ are the rotation and translation that relate the world coordinate frame to the camera coordinate frame.

### 2.1.4 Camera calibration

Camera calibration is used to estimate $P$. One of the most used methods for camera calibration is by Zhang [31]. This process uses a checkerboard pattern of known dimensions, as shown in Figure 2.2. Many calibration toolboxes exist that have the necessary steps for calibration built in. Some of the most common ones include OpenCV [32] and the Bouguet camera calibration toolbox [33].

To calibrate a camera, multiple views of the calibration object are taken at different orientations and angles as shown in Figure 2.3(a). The corners of the pattern are then detected using a corner detector such as the Harris corner detector [34] and their estimated locations are refined to sub-pixel accuracy. Figure 2.3(b) shows a typical result of the corner detection process.

An advantage of this pattern is that the corners of the checkerboard can be detected very accurately up to sub-pixel accuracy. Furthermore, the corners form a regular grid in a matrix form which means that errors in the detection of the corners

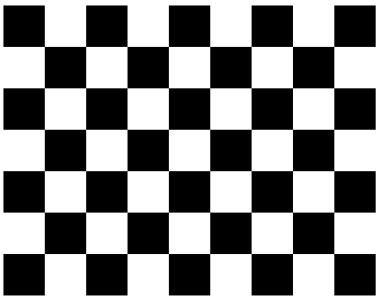


Figure 2.2: A picture of the checkerboard pattern used for calibration.

can be resolved by enforcing the grid shape. Also, accurate world coordinates are easily measured and specified for this pattern.

## 2.2 Binocular view geometry

A second camera is introduced to complete a stereo vision setup. This section details two-camera geometry necessary for stereo vision.

### 2.2.1 Stereo calibration

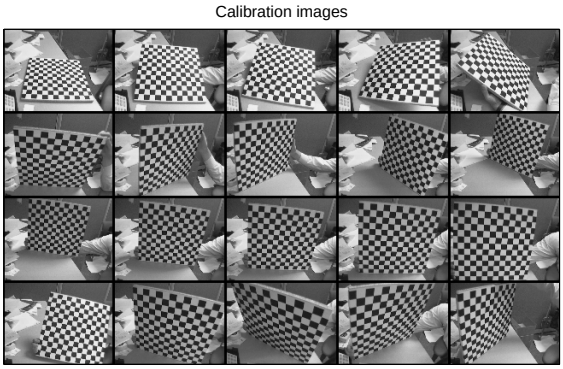
A second camera is now introduced in order to move a step closer to achieving the main objective namely inferring depth information from two cameras. Introducing a second camera arbitrarily would be of no use. For the second camera to be useful, one needs to know the spatial relationship between the two cameras. This means that the two cameras have to be calibrated to each other. This process is called stereo calibration [33]. Stereo calibration works in a similar way to single-camera calibration except that now there are two cameras. In stereo camera calibration we seek a single rotation matrix $R_s$ and a translation vector $T_s$ that relate the two cameras.

For any given 3D point $\mathbf{X}$ in world coordinates, single-camera calibration is used to map the point into the two camera coordinates

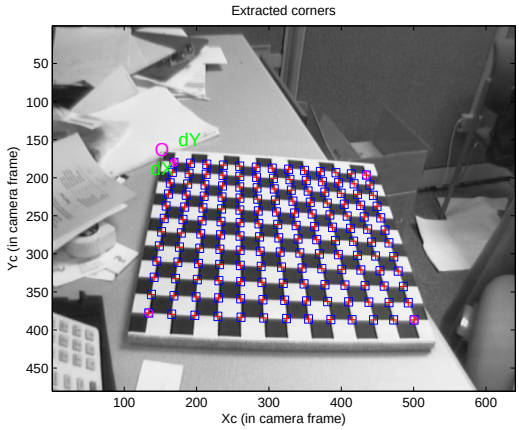
$$\mathbf{x} = R_1\mathbf{X} + T_1$$

and

$$\mathbf{x}' = R_2\mathbf{X} + T_2.$$




(a) Multiple images taken from different viewpoints for calibration using the Bouguet camera calibration toolbox.




(b) Image showing the checkerboard corners extracted on the first image in (a) using the Bouguet camera calibration toolbox.

Figure 2.3: Images showing stages of the calibration process.

The two views of $\mathbf{X}$ are then related by

$$\mathbf{x} = R_s^T(\mathbf{x}' - T_s). \tag{2.5}$$

The three equations mentioned above can now be used to solve for the rotation and translation separately as

$$R_s = R_2(R_1)^T, \quad T_s = T_2 - R_s T_1. \tag{2.6}$$

The stereo camera calibration process requires one to capture images of the same checkerboard with both cameras at the same time. Furthermore, to relate the point

correspondences from the two cameras, the checkerboard has to be clearly visible in both images.

### 2.2.2 Epipolar geometry

The epipolar geometry [30] between two views is essentially the geometry of the intersection of the image plane with a pencil of planes having the baseline as the axis, where the baseline is the line joining the camera centers.

Suppose a point $\mathbf{X}$ in 3-space is imaged in $\mathbf{x}$ in the first camera view and $\mathbf{x}'$ in the second. We seek to find the relationship between the corresponding points $\mathbf{x}$ and $\mathbf{x}'$. As shown in Figure 2.4, the image points $\mathbf{x}$ and $\mathbf{x}'$, the space point $\mathbf{X}$, and the camera centers $\mathbf{C}$ and $\mathbf{C}'$, are all coplanar. Denote this plane by $\pi$. The rays back-projected from $\mathbf{x}$ and $\mathbf{x}'$ intersect at $\mathbf{X}$ and the rays are coplanar, lying on $\pi$. It is this latter property that is most significant in searching for a correspondence.

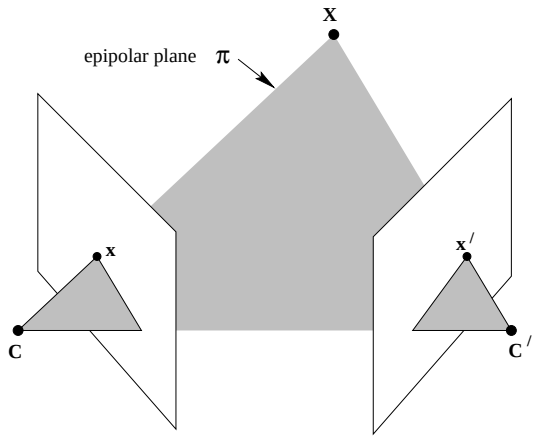



Figure 2.4: Point correspondence geometry. The two cameras are indicated by their centers $\mathbf{C}$ and $\mathbf{C}'$ and image planes. The camera centers, the point $\mathbf{X}$, and its images $\mathbf{x}$ and $\mathbf{x}'$ all lie on a common plane $\pi$.

Suppose now only $\mathbf{x}$ is known and we ask how the corresponding point $\mathbf{x}'$ is constrained. The plane $\pi$ is determined by the baseline and the ray defined by $\mathbf{x}$, as shown in Figure 2.5. The ray corresponding to the (unknown) point $\mathbf{x}'$ lies in $\pi$, hence the point $\mathbf{x}'$ lies on the line of intersection $\mathbf{l}'$ of $\pi$ with the second image plane. This line $\mathbf{l}'$ is the image in the second view of the ray back-projected from $\mathbf{x}$. It is the epipolar line corresponding to $\mathbf{x}'$. In terms of a stereo correspondence algorithm the benefit is that a search for a point corresponding to $\mathbf{x}$ need not cover the entire image plane, but can be restricted to the line $\mathbf{l}'$. It turns out that one can

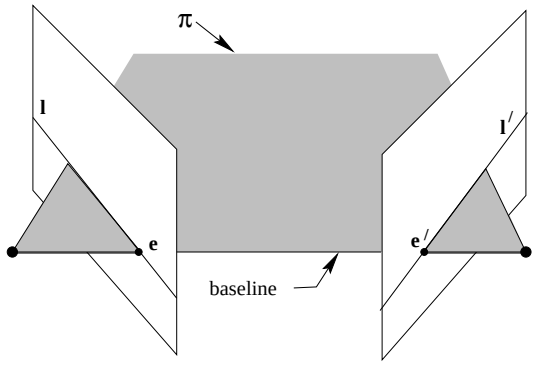



Figure 2.5: Epipolar geometry. The camera baseline intersects each image plane at the epipoles $\mathbf{e}$ and $\mathbf{e}'$. Any plane $\pi$ containing the baseline is an epipolar plane, and intersects the image planes on corresponding epipolar lines $\mathbf{l}$ and $\mathbf{l}'$.

further simplify the correspondence search using image rectification. This process is described next.

### 2.2.3 Rectification

Given a pair of stereo images, rectification [33] determines a geometric transformation of each image plane such that pairs of conjugate epipolar lines become collinear and parallel to one of the image axes (usually the horizontal axis). Effectively this means that we have to define a new rotation $R_n$ and a new intrinsic parameter matrix $K_n$. The rectified images can be thought of as acquired by a new stereo rig, obtained by rotating the original cameras. The important advantage of rectification is that computing stereo correspondences is made simpler because the search need only be done along the horizontal lines of the rectified images.

If $P_1$ and $P_2$ are the camera matrices corresponding to the two images, the new intrinsic parameters can be chosen to be

$$K_n = \frac{(K_1 + K_2)}{2}. \tag{2.7}$$

The objective is to have the two images coplanar, meaning that the $Z$ axis of the new orientation $R_n$ has to be perpendicular to the baseline (the line joining $\mathbf{C}$ and $\mathbf{C}'$). This does not ensure that the image scan lines are aligned. For the alignment, the scan lines need to be parallel to the baseline. The first unit vector $\mathbf{u}_1$ is chosen to be along the baseline. This unit vector corresponds to the $X$ axis of the new orientation and is expressed as

$$\mathbf{u}_1 = \frac{(\mathbf{C} - \mathbf{C}')}{||(\mathbf{C} - \mathbf{C}')||}. \tag{2.8}$$

In order to define the $Y$ axis the unit vector of the principal ray of the old image, $\mathbf{n}$ is used. The $Y$ axis has to be perpendicular to $\mathbf{u}_1$, therefore

$$\mathbf{u}_2 = \mathbf{n} \times \mathbf{u}_1. \tag{2.9}$$

The last unit vector, $\mathbf{u}_3$, corresponding to the $Z$ axis has to be perpendicular to both $\mathbf{u}_1$ and $\mathbf{u}_2$, so

$$\mathbf{u}_3 = \mathbf{u}_1 \times \mathbf{u}_2. \tag{2.10}$$

Now the new rotation can be written as

$$R_n = \begin{pmatrix} \mathbf{u}_1^T \\ \mathbf{u}_2^T \\ \mathbf{u}_3^T \end{pmatrix} \tag{2.11}$$

The images may be remapped by defining the transformations $S_1$ and $S_2$ as

$$S_1 = M_n M_1^{-1}, \quad S_2 = M_n M_2^{-1} \tag{2.12}$$

where $M_n = K_n R_n$ and $M_i = K_i R_i$ for $i = 1, 2$. The image points $\mathbf{x}$ and $\mathbf{x}'$ of the original images can now be mapped to the rectified image points $\mathbf{x}_{rect}$ and $\mathbf{x}'_{rect}$ as follows:

$$\mathbf{x}_{rect} = S_1 \mathbf{x}, \quad \mathbf{x}'_{rect} = S_2 \mathbf{x}. \tag{2.13}$$

The effect of this mapping can be illustrated in Figure 2.6, where the image scanlines are aligned.


Figure 2.6: Image rectification. The two images are calibrated and rectified. By rectifying the images the scanlines of both images are aligned which restricts the correspondence search.

### 2.2.4 Triangulation

At this point, enough has been described to be able to find the 3D coordinate of a point given undistorted and rectified stereo images. This is done by using triangulation. It is assumed that the image coordinates $\mathbf{x}$ and $\mathbf{x}'$ of the feature coordinate $\mathbf{X}$ in the left and right images respectively are known. Then $\mathbf{X}$ can be calculated by using similar triangles as follows: If $\mathbf{x} = (u, v)$, where $u$ and $v$ are the row and column of the image, and $\mathbf{x}' = (u', v')$ are the image coordinates of the left and right images respectively, then the disparity $d$ is defined as $d = v - v'$. Given the baseline $B$ and the focal length $f$ of the left camera, the depth $Z$ of $\mathbf{X}$ can be determined by using similar triangles as illustrated in Figure 2.7 as

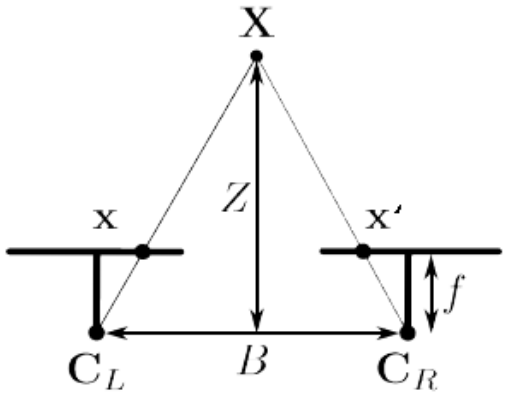



Figure 2.7: Triangulation of $\mathbf{X}$ using similar triangles. The depth $Z$ is triangulated using the disparity $d$ of the two image coordinates of $\mathbf{X}$.

$$\frac{B-d}{Z-f} = \frac{B}{Z}, \tag{2.14}$$

$$Z = \frac{fB}{d}.$$

The depth of a point is inversely proportional to the disparity. In order to visualize this relationship one can plot the disparity versus the depth as shown in Figure 2.8.

The figure illustrates that for small values of $d$, the estimates of the depth based on $d$ become more inaccurate. This point is important to note as the work progresses. Next, one needs to find the $X$ and $Y$ coordinates of the world point $\mathbf{X}$. Using the left image of the stereo camera pair, similar triangles can be used to find $X$ as

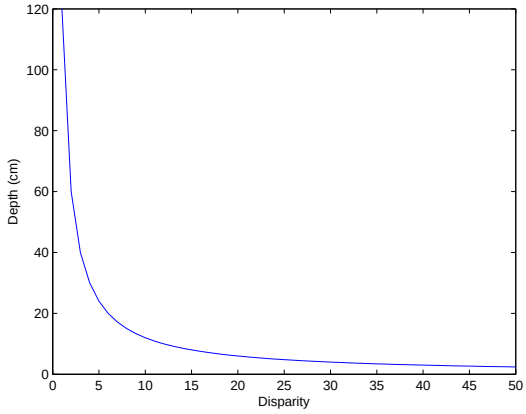



Figure 2.8: Depth vs. disparity for $f = 1000$ (measured in pixels) and $B = 12cm$. The image shows that for large disparities, the depth resolution is high and decreases as the disparities become smaller.

follows:

$$X = \frac{(u - p_u)}{f} Z \tag{2.15}$$

where $p_u$ is the vertical offset of the principal point. Equation 2.15 can be written in terms of $B$ and $d$ by substituting for $Z$ as follows:

$$X = \frac{(u - p_u)}{f} \frac{fB}{d} = \frac{(u - p_u)B}{d}. \tag{2.16}$$

Similarly,

$$Y = \frac{(v - p_v)B}{d}, \tag{2.17}$$

where $p_v$ is the horizontal offset of the principal point. Homogeneous coordinates can be used to neatly encapsulate the whole process of triangulation. The matrix $Q$ can be defined such that

$$\begin{pmatrix} WX \\ WY \\ WZ \\ W \end{pmatrix} = Q \begin{pmatrix} u \\ v \\ d \\ 1 \end{pmatrix}, \tag{2.18}$$

with

$$Q = \begin{pmatrix} 1 & 0 & 0 & -p_u \\ 0 & 1 & 0 & -p_v \\ 0 & 0 & 0 & f \\ 0 & 0 & \frac{1}{B} & 0 \end{pmatrix}. \tag{2.19}$$

Now one needs to find corresponding pixels in a stereo image pair. This is called the stereo correspondence problem and is described in the next chapter.

Chapter 3

# Literature review

Stereo vision is one of the most actively researched fields in computer vision. A large number of stereo correspondence algorithms have been developed. In order to gauge progress in the area, Schartein et al. [2] wrote a paper which provides an update on the state of the art in dense two frame stereo correspondence algorithms under known camera geometry. The algorithms produce dense disparity maps which are useful in a number of applications including view synthesis, robot navigation, image-based rendering and tele-presence. For researchers in stereo vision one of the most useful outputs of the paper is a quantitative testbed for stereo correspondence algorithms available, from `vision.middlebury.edu/stereo`. The testbed allows an on-line evaluation of developed stereo correspondence algorithms and rates the algorithms according to their depth map quality relative to the ground truth. Because of the quantitative on-line evaluations the testbed is widely used in the stereo vision community. This section reviews stereo correspondence algorithms mostly which have received attention in the Middlebury evaluation [1].

## 3.1 Review of stereo correspondence algorithms

Top-performing stereo correspondence algorithms make use of colour segmentation techniques. The mean-shift [35] algorithm is popular for performing colour segmentation. Currently, the best performing algorithm in the Middlebury evaluation page is based on inter-regional cooperative optimization [22]. The algorithm firstly segments the images into homogeneous regions. Secondly, a local-matching algorithm is used to compute the initial disparity estimate. Thirdly, a voting-based plane fitting technique is applied to obtain the parameters of a disparity plane corresponding to each image region. Finally, the disparity plane parameters of all regions are iteratively optimized by an inter-regional cooperative optimization procedure until a reasonable disparity map is obtained. The algorithm in [23] is similar to [22] but instead of using inter-regional optimization, it uses belief propagation to determine the optimal plane parameters. The iterative nature of this algorithm makes it very slow and thus not suitable for real-time implementations.

A slightly different approach to solving the stereo correspondence problem can be found in [36]. Pixels are classified as either stable, unstable or occluded. Occluded pixels are those which fail the left-right consistency check [37, 38]. The distinctiveness of the correlation measure peak is then used to classify the pixels which passed the left-right consistency check into stable and unstable pixels. Information from

the stable pixels is then propagated to the unstable and occluded pixels by using colour segmentation and plane fitting. Hierarchical belief propagation in a global energy minimization framework is then used to iteratively improve the plane fitting results. Instead of classifying pixels as in [36], [39] uses an outlier confidence model to measure how likely a pixel is occluded. Because there is no direct labelling, the model has more tolerance for errors produced in the occlusion detection process. The model then uses a reliable colour refinement scheme to locally infer the possible disparity values for the outlier pixels. Belief propagation is used to minimize the energy function.

Segmentation-based stereo correspondence algorithms work well in textureless regions because they assume that depth varies smoothly within regions of homogeneous colour, and that depth discontinuities coincide with depth boundaries. The algorithm in [40] takes into account the drawback of this assumption: depth discontinuities may not lie along colour segmentation boundaries, resulting in segments that span depth discontinuities. In order to overcome this effect, the algorithm in [40] jointly estimates image segmentation, depth, and matting/depth information for mixed pixels, which span two objects at different depths. An over-segmentation approach is used to represent the scene as a collection of fronto-parallel planar segments. The segments are then characterized by their depth, 2D shape and colour. These parameters are jointly estimated by alternating the update of segment shapes and depths. The segment shapes are updated using a generative model that accounts for mixed pixels at the segment boundary as well as the depth and shape probabilities. Segment depths are updated by defining a pairwise Markov random field and belief propagation is used to minimize the energy.

In [41], an explicit treatment of occlusions is carried out. The visibility constraint, which requires that an occluded pixel must have no match on the other image and a non-occluded pixel must have at least one match, is exploited. This results in a symmetric stereo model that can handle occlusions. The visibility constraint is embedded in an energy minimization framework. The energy is minimized with an iterative optimization algorithm that uses belief propagation. In [42], a near real-time stereo correspondence algorithm which explicitly deals with textureless regions is developed. Instead of having strong planarity constraints in the environment, which tend to force non-planar objects onto planes, the algorithm uses a compromise approach. Depth estimates are preferred while in textureless regions, the estimates can be replaced with planes. A near real-time colour segmentation algorithm is used and planes are fitted in textureless segments. The planes are refined using consistency constraints. Loopy belief propagation is used to correct local errors and improve the algorithm. The algorithm in [43] uses a window-based algorithm to obtain initial depth estimates. Instead of halting the process after the initial depth estimation, the process is taken one step further using a technique called disparity calibration. For disparity calibration, an appropriate calibration window is selected for each pixel using colour similarity and geometric proximity.

By using these calibration windows, a local disparity calibration method is designed to acquire accurate disparity estimates.

In [44], an extension of semi-global matching is used to solve stereo correspondence in a structured environment. In order to handle untextured areas, intensity consistent disparity selection is proposed. Holes caused by filters are filled in by discontinuity preserving interpolation. One of the main advantages of this algorithm is its low complexity and runtime. The algorithm in [45] is based on scanline optimization (SO). The algorithm embodies a function based on variable support. Low-textured regions are handled by the SO framework and the variable support helps to preserve accuracy along depth borders. A refinement step based on a technique that exploits symmetrically the relationship between occlusions and depth discontinuities on the disparity maps obtained, assuming alternatively as reference the left and the right image, allows for accurately locating borders.

A similarity measure called the Distinctive Similarity Measure (DSM) is proposed in [46]. The DSM resolves the point ambiguity based on the idea that the distinctiveness, not the interest, is the appropriate criterion under the point ambiguity. The distinctiveness of a point is related to the probability of a mismatch. Also, the dissimilarity between image points is used since it is related to the probability of a good match. The algorithm in [10] uses a local algorithm based on adaptive weights for cost aggregation. The algorithm includes information obtained from a segmentation process in order to improve aggregation. The algorithm in [47] combines the strength of the region-based approach and the 2D DP optimization framework. Instead of optimizing a global energy function defined on a 2D pixel-tree structure using DP, a region-tree built on over-segmented image regions is used. The resulting disparity maps do not contain any streaking problems as is common in SO algorithms because of the tree structure.

The algorithm in [48] is a local-matching algorithm which uses varying support weights for cost aggregation. The support weight in a given support window is based on colour similarity and geometric proximity to reduce the image ambiguity. The algorithm produces surprisingly good results for a local-matching algorithm. The algorithm in [49] solves the stereo correspondence problem by formulating the problem as a large scale linear programming problem. The match cost function is approximated by a piecewise linear convex function. The resulting problem is solved using an interior point method and the associated Newton steps involve matrices that reflect the structure of the underlying pixel grid. The algorithm in [50] aims to improve sub-pixel accuracy in low-texture regions. The algorithm preserves depth discontinuities and enforces smoothness on a sub-pixel level. A stereo constraint called the gravitational constraint is presented. The constraint assumes sorted disparity values in a vertical direction and guides global algorithms to reduce false matches, especially in low-texture regions.

The algorithm in [51] is a semi-global algorithm which uses mutual information for pixel-wise matching. The calculation of mutual information is performed hierarchically. A global cost calculation is approximated and can be performed at a time that is linear in the number of pixels. The algorithm in [52] uses fast converging hierarchical belief propagation to achieve a real-time stereo correspondence algorithm on a graphics processing unit (GPU). A novel approach is used to adaptively update pixel costs since belief propagation is linear in the number of iterations, making it unfeasible for practical applications. The algorithm in [53] is a two-step local stereo correspondence algorithm, initial matching and disparity estimation, which employs segmentation cues. The initial matching uses a raw matching cost with the contrast context histogram descriptor and two-pass cost aggregation with segmentation-based adaptive support weight. The disparity computation consists of two parts: narrow occlusion handling and multi-directional weighted least-squares fitting for the broad or large occlusion areas.

A near real-time local stereo correspondence algorithm is developed in [27] based on segmentation. The algorithm uses effective cost aggregation and finds a compromise between cost aggregation and computational time. The algorithm in [54] is a local stereo correspondence algorithm which uses the anisotropic local-polynomial approximation-intersection of confidence intervals technique in order to define an appropriate window size. The method performs the entire disparity estimation and refinement within the local high-confidence voting framework. The algorithm in [55] uses a combination of binocular and monocular cues for initial match candidates. The matching candidates are then embedded in disparity space, where perceptual organization takes place in 3D neighbourhoods. The assumption is that correct matches produce salient, coherent surfaces, while wrong ones do not. Matching candidates that are consistent with the surfaces are kept and grouped into smooth layers. The projections of the refined surfaces on both images are used to obtain disparity hypotheses for unmatched pixels. The final disparities are selected after a second tensor voting stage, during which information is propagated from more reliable pixels to less reliable ones.

The algorithm in [56] is a real-time DP based algorithm on a GPU. The algorithm reduces the typical streaking artifacts by aggregating the per-pixel matching cost in the vertical direction. The algorithm in [57] uses DP in a tree structure instead of on the individual scanlines. The nodes on the tree represent image pixels, but only the most important edges of the 4-connected neighbourhood system are included. The algorithm becomes a global optimization method because a disparity estimate at one pixel depends on the estimates at all other pixels.

Chapter 4

# Local-matching stereo algorithms

The aim of stereo correspondence is to find corresponding pixels between two images of the same scene taken from slightly different viewpoints. This means that for a pixel in the left image, every pixel in the right image can be scanned to find which one corresponds. This would be very computationally expensive. Recalling from Section 2.2.3, the search space for finding pixel correspondences can be significantly reduced by calibrating and rectifying the images. This simplifies the stereo correspondence problem significantly. The problem can now be phrased as: given two images of the same scene taken from slightly different viewpoints, for a pixel in the left image, find a corresponding pixel in the right image along the same scanline and the distance by which the pixel in the left image is translated relative to its corresponding pixel in the right image. Mathematically, if $\mathbf{x} = (u, v)$ is the left image pixel coordinate where $u$ is the image row/scanline and $v$ is the image column and $\mathbf{x}' = (u', v')$, is the right image pixel coordinate, then the correspondence problem can be written as:

$$u = u', \quad v = v' - d, \tag{4.1}$$

where $d$ is the disparity. If we compute $d$ for every pixel in the left image, a dense disparity map is obtained.

The earliest attempts into solving the stereo correspondence problem involve the use of local-matching algorithms [8, 9]. These algorithms generally use some kind of statistical correlation between colour or intensity patterns in local support windows. By using the local support windows, the image ambiguity is reduced efficiently while the discriminative power of the similarity measure is increased.

Generally, stereo correspondence algorithms can be broken down into components. These components allow us to separate the different design considerations of a particular algorithm. By separating the algorithms into their components, they can be evaluated thoroughly and the inner workings can be easily understood. The four steps generally performed by a stereo correspondence algorithm are:

- *Matching cost computation*
  A matching cost is defined and used to measure pixel similarity.

- *Cost (support) aggregation*
  A support region is defined to spatially aggregate the matching cost.

- *Disparity computation / optimization*
  The best disparity hypothesis for each pixel is defined to minimize a cost function.

- *Disparity refinement*
  The disparity estimates are post-processed to remove outliers and/or perform sub-pixel estimation.

A particular algorithm might perform all four steps or a subset of the steps. The sequence in which they are applied depends on the algorithm. Our interest lies in local-matching algorithms that perform all the above-mentioned steps.

In this chapter a basic local-matching stereo algorithm is described by decomposing it into the different steps described above. It is then shown that by breaking down the algorithm we can look at the different design considerations and improve it. The improvement is demonstrated by implementing a state-of-the-art variant of local-matching algorithms.

## 4.1 Simple local-matching algorithm

A simple local-matching algorithm is implemented. The algorithm is broken down into its components and the different design considerations for the algorithm are covered.

### 4.1.1 Matching Cost

In order to find corresponding pixels we need some way of quantifying how similar they are. To achieve this a matching cost $e(\mathbf{x}, \mathbf{x}')$ is defined, where $\mathbf{x}$ and $\mathbf{x}'$ are the pixel locations in the left and right images respectively. By using the intensity values at the specific image locations $I(.)$, a matching cost can be defined using the absolute differences of the pixel intensities as follows:

$$e(\mathbf{x}, \mathbf{x}') = |I_L(\mathbf{x}) - I_R(\mathbf{x}')| \tag{4.2}$$

where $I_L(.)$ and $I_R(.)$ represent the intensity at the specific image locations in the left and right images respectively.

Pixels that are similar will give a low score and ones that are different will give a high score. An extension of the above mentioned matching cost would be Birchfield and Tomasi's sampling insensitive matching cost [58]. The reader is referred to [59] for a comprehensive study on the performance of matching costs under different radiometric changes of the input images. Since now there is a way of determining the similarity of pixels, the only part left is the process of searching and choosing the corresponding pixels.

### 4.1.2 Ordering Constraint

Recalling that the images are calibrated and rectified, the search for corresponding pixels is limited to a single scanline. In order to find correspondences, a pixel in the left scanline compared to every pixel in the corresponding scanline of the right image. It turns out that the search for a corresponding pixel can be reduced by using an ordering constraint [21].

Considering the two-camera geometry, the ordering constraint can be phrased as follows: pixels $A$ and $B$ are on the left image and their matches on the right image are $A'$ and $B'$ respectively. If pixel $A$ is on the left of pixel $B$ in the left image, $A'$ has to be to the left of $B'$ in the right image. This constraint does not always hold: it is violated by thin objects which are close to the camera as demonstrated in Figure 4.1. However, it can be assumed that most objects in the scene are large. The ordering constraint restricts the search for correspondence of a pixel in the left image scanline to be to the left of the same pixel in the right image scanline.

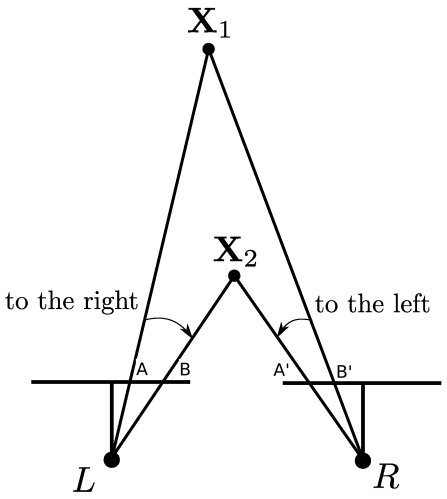



Figure 4.1: The ordering constraint violated if features are thin objects close to the camera.

### 4.1.3 Disparity computation

The ordering constraint allows to restrict the correspondence search to a certain direction. The search can be further reduced by choosing the disparity range to be $d \in [d_{min}, d_{max}]$. Using Equation 4.2 and the ordering constraint, the matching costs between the pixel of interest in the left image and the candidate pixels in the right image can be computed. The computation produces a correlation curve as illustrated in Figure 4.2.

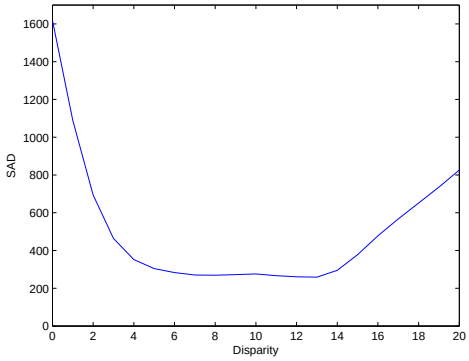



Figure 4.2: Correlation curve produced by calculating the pixel simmilarity between the pixel of interest and candidate matching pixels.

The disparity of the pixel is then selected using a Winner-Takes-All (WTA) method without considering global reasoning as

$$d = \text{argmin}(e(\mathbf{x}, \mathbf{x}'(d))) = \text{argmin}(|I_L(u, v) - I_R(u', v' - d)|). \tag{4.3}$$

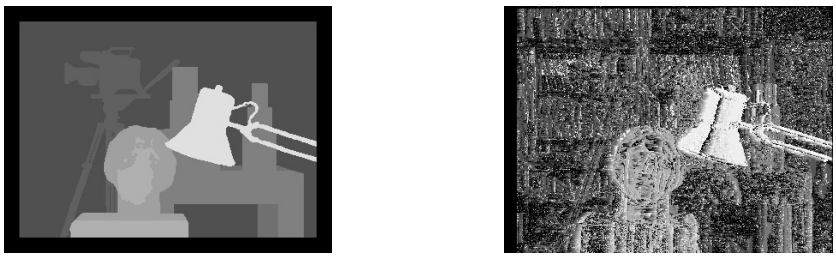
This algorithm gives a disparity map as shown in Figure 4.3.

Although the disparity map in Figure 4.3 is very detailed, it suffers from noise. This is due to the fact that the similarity measure is done pixelwise. To reduce this noise one can support the pixels of interest with their neighbouring pixels as shown in the next section.

### 4.1.4 Cost Aggregation

Using the raw matching cost gives noisy results as shown in Figure 4.3. In order to combat this problem, a support region is defined to spatially aggregate the matching cost. A number of aggregation strategies exist [60]. The matching cost can be aggregated by using an $n \times n$ window centered at the pixel of interest. Then Equation 4.2 becomes

$$\mathcal{E}(\mathbf{x}, \mathbf{x}') = \sum_{\mathbf{x}_a \in N_L, \mathbf{x}'_a \in N_R} |I_L(\mathbf{x} - \mathbf{x}_a) - I_R(\mathbf{x}' - \mathbf{x}'_a)|. \tag{4.4}$$



(a) Ground truth disparity map.

(b) Disparity map produced by using absolute differences as a matching cost.

Figure 4.3: Results of using the absolute differences as a matching cost for stereo correspondence.

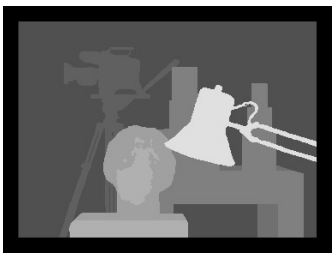
where $N_L$ and $N_R$ represent the support windows centered on pixels $\mathbf{x}$ and $\mathbf{x}'$ respectively. $\mathbf{x}_a$ and $\mathbf{x}'_a$ are the pixels within $N_L$ and $N_R$ respectively. Aggregation of this form assumes a constant disparity across the support window. Figure 4.4 shows the resultant disparity maps obtained while varying the support window size.

It is observed that increasing the window size reduces the noise in the disparity maps. Furthermore, the resolution of the disparity map is decreased. Because every pixel of interest has a support window associated with it, the computational time of the disparity map is increased. An artifact called the fattening-effect is introduced at the object boundaries. Choosing the optimal window size becomes an important consideration in the design of a stereo algorithm. There have been several studies into addressing the choice of window size [25].

Three observations can be made from the resulting disparity maps. Firstly, local-matching stereo algorithms fail in weakly-textured regions because there is not enough information to reliably match the pixels in that region. Secondly, if a pixel is occluded, visible in one view and not visible in the other view, the algorithm will give incorrect results because it does not penalize these occurrences. Thirdly, because the disparities are calculated as discrete values, the algorithm suffers from the discretization effect [2]. The next section describes how some of these problems can be addressed.

### 4.1.5 Disparity refinement

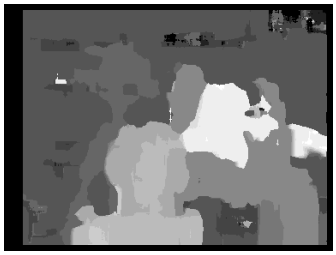
In this section, some of the post-processing steps used in stereo vision are discussed.



(a) Ground truth disparity map

(b) Disparity map produced by using the sum of absolute differences with a $3 \times 3$ window size.

(c) Disparity map produced by using the sum of absolute differences with a $9 \times 9$ window size.



(d) Disparity map produced by using the sum of absolute differences with a $15 \times 15$ window size.

Figure 4.4: Disparity maps obtained by varying the support window size.

#### 4.1.5.1 Left-right consistency check

The left-right consistency check [37, 38, 61, 62] is used to detect inconsistencies in the matching process. The check works by reversing the roles of the left and right images. Firstly, the matching pixel of the left image in the right image is determined. Then the roles of the images are reversed and there is a check if the right image pixel matches the left image pixel. This check is very useful especially when a pixel is occluded.

#### 4.1.5.2 Sub-pixel refinement

The disparity computation gives discrete disparity values. To remove the discretization effects, one can perform sub-pixel estimation based on quadratic polynomial interpolation [63]. $d_{sub}$ is approximated between three discrete variables, $f(i) = \mathcal{E}(\mathbf{x}, \mathbf{x}' - d)$, $f(i+1) = \mathcal{E}(\mathbf{x}, \mathbf{x}' - (d+1))$ and $f(i-1) = \mathcal{E}(\mathbf{x}, \mathbf{x}' - (d-1))$ as

$$d_{sub} = d - \left\{ \frac{f(i+1) - f(i-1)}{2\big(f(i+1) + f(i-1) - 2f(i)\big)} \right\}. \tag{4.5}$$

## 4.2 Pseudo-code

The different design considerations of a local-matching stereo algorithm have been covered. The algorithms can be summarized in pseudo-code shown in Algorithm 1.

**Algorithm 1** Stereo algorithm

*INPUT: Stereo images, window size, disparity range.*
*OUTPUT: Disparity map.*

```
for each pixel in the left frame do
    set support region around the pixel (left frame)
    set search window in the right frame
    for each pixel in the search window (right frame) do
        set correlation window around the pixel
        correlate support region with correlation window
    find best match
    calculate disparity
    refine disparity
```

## 4.3 Adaptive weights for cost aggregation

The algorithm implemented is found in [10]. For a matching cost the algorithm uses truncated absolute differences. The cost is expressed as

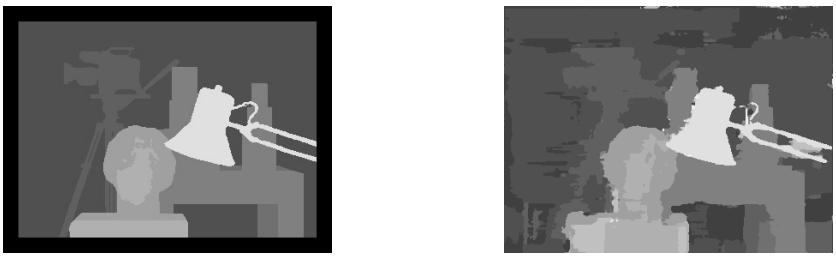
$$e(\mathbf{x}, \mathbf{x}') = \min(|I_L(\mathbf{x}) - I_R(\mathbf{x}')|, T), \tag{4.6}$$

where $T$ is the truncation value that controls the limit of the matching cost. The intensity values used in this algorithm are based on the *Lab* color space which closely represents the human visual system. The main contribution of the algorithm is in the aggregation step. Similar to the simple matching algorithm, a square window is used for aggregation but with some additions. The approach taken in this algorithm is based on the observation that pixels in a support region are not equally important for support aggregation. Every pixel within a window is given a support weight based on the gesalt grouping [64, 65]:

$$w(p, q) = f(\Delta c_{pq}, \Delta g_{pq}), \tag{4.7}$$

where pixel $p$ is the center pixel of the window which is at position $(u, v)$ according to the previous description. Pixel $q$ is a pixel neighbour of $p$ within the support window. $\Delta c_{pq}$ and $\Delta g_{pq}$ represent the colour difference and spatial difference between pixel $p$ and $q$. By regarding $\Delta c_{pq}$ and $\Delta g_{pq}$ as independent events, $f(\Delta c_{pq}, \Delta g_{pq})$ can be written as

$$f(\Delta c_{pq}, \Delta g_{pq}) = f_s(\Delta c_{pq}) f_p(\Delta g_{pq}), \tag{4.8}$$



(a) Ground truth disparity map.

(b) Disparity map produced using adaptive weights for cost aggregation.

Figure 4.5: Results produced by using adaptive weights for cost aggregation.

where $f_s(\Delta c_{pq})$ and $f_p(\Delta g_{pq})$ represent the strength of grouping by similarity and proximity. These groupings are modelled as

$$f_s(\Delta c_{pq}) = \exp\left(-\frac{\Delta c_{pq}}{\gamma_c}\right) \tag{4.9}$$

and

$$f_p(\Delta g_{pq}) = \exp\left(-\frac{\Delta g_{pq}}{\gamma_p}\right), \tag{4.10}$$

where $\Delta c_{pq}$ represents the Euclidean distance between two colors, $\mathbf{c}_p = [L_p, a_p, b_p]$ and $\mathbf{c}_q = [L_q, a_q, b_q]$. $\Delta g_{pq}$ is the Euclidean distance between $p$ and $q$ in the image domain. $\gamma_c$ is a factor used in the strength of grouping by similarity and $\gamma_p$ is the radius of the window size. The support weight based on the strength of the groupings then becomes

$$w(p,q) = \exp\left(-\left(\frac{\Delta c_{pq}}{\gamma_c} + \frac{\Delta g_{pq}}{\gamma_p}\right)\right). \tag{4.11}$$

Using Equation 4.11, the dissimilarity between pixels becomes

$$\mathcal{E}(\mathbf{x}, \mathbf{x}') = \frac{\sum_{q \in N_p, \bar{q}_d \in N_{\bar{p}_d}} w(p,q) w(\bar{p}, \bar{q}) e(q, \bar{q}_d)}{\sum_{q \in N_p, \bar{q}_d \in N_{\bar{p}_d}} w(p,q) w(\bar{p}, \bar{q})}. \tag{4.12}$$

The disparity computation used is the same as in Equation 4.3. The results of the extensions to the simple local-matching stereo algorithm are shown in Figure 4.5.

Chapter 5

# Temporal seeding in stereo image sequences

The previous chapter dealt with obtaining a dense disparity map from static images. In this chapter, the temporal domain of a stereo image sequence is explored. In a stereo image sequence, finding feature correspondences is normally done for every pair of frames without taking temporal information into account. Current imaging sensors can acquire images at high frequencies resulting in small movements between consecutive image frames. Since there are small inter-frame movements, the frame-to-frame disparity estimates do not change significantly. We explore using previously computed disparity estimates to seed the matching process of the current stereo image pair.

Most conventional stereo correspondence algorithms contain fixed disparity search ranges by assuming the depth range of the scene as in Equation 4.3. This range stays constant throughout a stereo image sequence. By decreasing the disparity search range the efficiency of the matching process can be improved. This can be seen in [66], where the probability of an incorrect stereo match is given by:

$$P^T \propto P^a + P^b + P^c,$$

where $P^a$ is the probability of mismatching a pair of features when neither feature has its correct match detected in another image, $P^b$ is the probability of mismatch when one feature has had its correct match detected, and $P^c$ is the probability of mismatch when both features have had their correct matches found in the other image. The probabilities, $P^a$, $P^b$ and $P^c$ are all proportional to the mean number of candidate matches and thus $P^T$ is proportional to the disparity search range of the stereo correspondence algorithm. Therefore by reducing the disparity search range, $P^T$ is reduced assuming that the correct match remains in the reduced range.

A method of using temporal information in stereo image sequences to decrease the disparity search range so as to decrease the probability of a mismatch is developed. A local-matching stereo correspondence algorithm is implemented on KLT (Kanade Lucas Tomasi) features and the disparity estimates obtained are used on the consecutive stereo image frame to seed the matching process. Local-matching stereo algorithms have a uniform structure, as demonstrated in Chapter 4. This allows the temporal seeding method to be used across different variations of these stereo algorithms. The method is expected to be at the least as accurate as the local-matching

algorithm at a lower computational expense when using a small number of frames.

Errors in both the local-matching algorithm and the temporal seeding algorithm are quantified. The algorithms are run on two successive stereo images and an error comparison is carried out. Furthermore, the algorithms are run on 87 frames of a different dataset and temporal seeding is evaluated.

This section is structured as follows. Section 5.1 covers the related literature. Section 5.2 defines the problem to be solved. Section 5.3 discusses the feature matching and detection process. The local-matching stereo correspondence algorithm implemented is discussed in Section 5.4. Section 5.5 discusses the temporal seeding process. Section 5.6 discusses the experiments and results.

## 5.1 Related work

There have been successful implementations of enforcing temporal constraints for depth estimation in successive stereo image frames. The work in this chapter is directly related to approaches that combine motion and stereo.

Algorithms such as those in [67, 68, 69] can be classified as pseudo-temporal stereo vision algorithms. They aim to solve the stereo correspondence problem for a wide baseline. The input to such algorithms is a monocular sequence of images produced by a camera undergoing controlled translating or rotating motion. Stereo image pairs are produced by selecting and pairing different images from the input sequence. The algorithms initially start by finding correspondences in a short baseline stereo pair and use these correspondences to bootstrap the stereo correspondences of a wider baseline stereo image pair. The propagation of disparity values from one set of frames to the next helps to improve computational efficiency and reliability of stereo matching.

The work in [70] and [71] also takes into account temporal information to solve for depth from triangulation. These methods extend support aggregation from 2D to 3D by adding the temporal domain. These algorithms can be viewed as exploiting temporal aggregation to increase matching robustness.

The work in [72] uses previous disparity estimates by analyzing their local neighbourhood to decrease the disparity search range of the current stereo image frame. The computational load and robustness of using temporal information are demonstrated. Although this algorithm performs well, it suffers from start-up problems. Research from [73] addressed the start-up problem and was successfully used on a wide baseline stereo sequence.

The work in this chapter is similar to that in [72] but instead of analyzing the local

neighbourhood of the previous estimate to decrease the search range, the search range is decreased according to the shape of the correlation curve in the successive image frame given an initial disparity estimate.

## 5.2 Problem statement

The problem to be solved can be defined as follows.

The input is a set of calibrated and rectified stereo image pairs, $\{\mathcal{L}_t, \mathcal{R}_t\}_{t=0}^{N}$, each pair acquired at time $t = 0, ..., N$. $\mathcal{L}_t$ and $\mathcal{R}_t$ denote the left and the right images taken at time $t$. An image coordinate, $\mathbf{x}_t = (u_t, v_t) \in \mathcal{F}$, represents a pixel location of a detected feature in the set of features $\mathcal{F}$ at row $u_t$ and column $v_t$ in the left image, while $\mathbf{x}'_t = (u'_t, v'_t)$ represents the corresponding pixel on the right image. If $d(\mathbf{a})$ is the disparity estimate of pixel $\mathbf{a}$, then our goal is to determine a disparity estimate $d(\mathbf{x}_{t+1})$ when given $d(\mathbf{x}_t)$ as a prior. The different aspects of the problem are discussed in the following sections.

## 5.3 Feature detection and matching

This section discusses the feature detector and tracker used in determining and tracking the features $\mathcal{F}$.

### 5.3.1 KLT (Kanade-Lucas-Tomasi)

The Kanade-Lucas-Tomasi (KLT) feature tracker is based on the early work by Lucas and Kanade (LK) on optical flow in [74]. The LK algorithm attempts to produce dense disparity estimates. The method is easily applied to a subset of points in the image and can be used as a sparse technique. Using the LK algorithm in a sparse context is allowed by the fact that it relies on local information extracted from some small window surrounding the point of interest.

LK works on three assumptions:

- *Brightness consistency*
  The appearance of a pixel does not change with time. This means that the intensity of a pixel denoted as $I(.)$ is assumed to remain constant between image frames: $I_t(\mathbf{x}_t) = I_{t+1}(\mathbf{x}_{t+1})$.

- *Temporal persistence*
  The movement of the image frames is small, so a surface patch changes slowly over time.

- *Spatial coherence*
  Neighbouring pixels that belong to the same surface patch have similar motion and project to nearby image points on the image plane.

The KLT tracker [75] is a frame-by-frame temporal tracker for a single video sequence which is applied at a set of corner points that change throughout the tracking. Good features [76] are selected by examining the minimum eigenvalue of each $2 \times 2$ gradient matrix. The features are then tracked using a Newton-Rhapson method of minimizing the differences between two image patches. Figure 5.1 shows an image with 313 detected KLT features.

Detecting the features solves for $\mathcal{F}$ and tracking the features on successive frames estimates $\mathbf{x}_t \leftrightarrow \mathbf{x}_{t+1}$.

Figure 5.1: Features detected in the image. 313 KLT features are detected on this image.

## 5.4 Stereo correspondence

Chapter 4 described in detail how to solve the stereo correspondence problem. Now the problem is viewed at in the context of temporal seeding. Given an image pair that is calibrated and rectified, $\{\mathcal{L}_t, \mathcal{R}_t\}$, a set of detected features $\mathcal{F}$ in the left image, and corresponding pixels $\mathbf{x}_t \leftrightarrow \mathbf{x}'_t$, the objective is to determine $d(\mathbf{x}_t)$.

### 5.4.1 Stereo algorithm

The stereo algorithm used in this section is similar to that in Section 4.1 with the main difference being the matching cost used. Birchfield and Tomasi's (BT's) sampling insensitive matching cost [58] is used. This matching cost $e(\mathbf{x_t}, \mathbf{x'_t})$ is formulated as follows:

$$I_l(\mathbf{x}'_t) = \frac{1}{2}(I_R(\mathbf{x}'_t) + I_R(u'_t, v'_t - 1))$$

is the linearly interpolated intensity to the left of pixel $\mathbf{x}'_t$ and, analogously,

$$I_r(\mathbf{x}'_t) = \frac{1}{2}(I_R(\mathbf{x}'_t) + I_R(u'_t, v'_t + 1))$$

is to the right of pixel $\mathbf{x}'_t$, then $I_{min}(\mathbf{x}'_t)$ and $I_{max}(\mathbf{x}'_t)$ are computed as follows:

$$I_{min}(\mathbf{x}'_t) = \min(I_l(\mathbf{x}'_t), I_r(\mathbf{x}'_t), I_R(\mathbf{x}'_t)),$$

$$I_{max}(\mathbf{x}'_t) = \max(I_l(\mathbf{x}'_t), I_r(\mathbf{x}'_t), I_R(\mathbf{x}'_t)).$$

The matching cost is then computed as

$$e(\mathbf{x}_t, \mathbf{x}'_t) = \max(0, I_L(\mathbf{x}_t) - I_{max}(\mathbf{x}'_t), I_{min}(\mathbf{x}'_t) - I_L(\mathbf{x}_t)). \tag{5.1}$$

To aggregate the cost a square window is used, as in Section 4.1.4, and computing the disparities is the same as in Section 4.1.3. Furthermore, sub-pixel interpolation is performed for disparity refinement as in Section 4.1.5.2. In order not to confuse this section's stereo algorithm with the previously described algorithms, from this point it will be called BT's stereo algorithm.

## 5.5 Temporal seeding

The main contribution of this work lies in the way the computed $d(\mathbf{x}_t)$ is reused as a prior to determine $d(\mathbf{x}_{t+1})$. In order to achieve the objective, some important assumptions have to be made and justified.

### 5.5.1 Assumptions

The fact that the local structure of successive stereo disparity maps does not change significantly is noted. This means that one can assume that the shape of the correlation curve of a tracked feature point at time $t + 1$ does not change by much compared with the feature point's correlation curve at time $t$. This assumption is made feasible because of the LK algorithm's three assumptions, as stated in Section 5.3.1. Figure 5.2 shows shows an example where the assumption holds. The shapes and positions of the two correlation curves are almost identical despite the frame-to-frame movement.

### 5.5.2 Seeding

The disparity estimate $d(\mathbf{x}_t)$ is used as an initial disparity estimate for $d(\mathbf{x}_{t+1})$. The new disparity estimate is found by the local minimum around the initial estimate. This local minimum is then assumed to be the global minimum of the correlation curve. A technique similar to gradient descent is used to determine the local minimum. The gradients to the left and to the right of $d(\mathbf{x}_t)$ are determined. If the

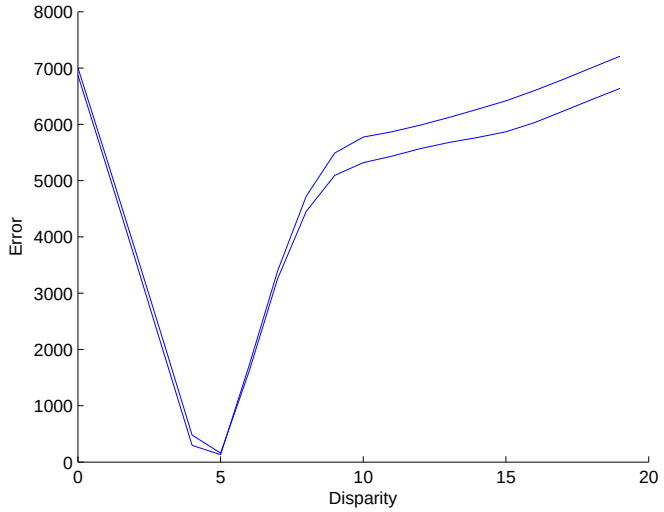



Figure 5.2: Correlation curves for a feature point in $\mathcal{F}$ at time $t$ and $t+1$.

gradients on both side of $d(\mathbf{x}_t)$ are negative, then we move a single step towards the descending slope until the gradient changes the sign from negative to positive. This signals the local minimum around the initial point. A similar process is carried out if the gradients on both side of $d(\mathbf{x}_t)$ are positive.

## 5.6 Experiments and results

This section discusses the experiments carried out to evaluate the temporal seeding algorithm.

### 5.6.1 Ground truth

Since features are tracked along the left images of a stereo sequence, one needs ground truth disparity estimates to evaluate the implemented stereo correspondence algorithm and the potential of temporal seeding. Ground truth is determined by first using the simple local-matching algorithm, as discussed in Section 4.1, on each feature point of a stereo pair. The disparity estimates are then refined to sub-pixel accuracy by using the LK optical flow method.

### 5.6.2 Experiments

Tests on BT's stereo algorithm and the temporal seeding algorithm are carried out on two successive stereo image pairs. Figure 5.3 shows the dataset used for the experiments. On the first stereo pair, feature points are detected and the disparity estimates of the features are calculated with BT's stereo algorithm. The disparity search range used is $[0, 19]$. Features are then tracked on the left images of the

sequence and temporal seeding is applied to determine the disparity estimates in the range of $[0, 19]$ on the successive stereo image pair.

(a) Left image of the first stereo image frame.

(b) Right image of the first stereo image frame.

(c) Left image of the second stereo image frame.

(d) Right image of the second stereo image frame.

Figure 5.3: Dataset used in the experiments.

Table 5.1: Table showing the Root Mean Square Error ($RMSE$) and computational times on 313 KLT features for BT's stereo algorithm and the temporal seeding algorithm using chosen stereo parameters.

| Frame number | Window size | Disparities | BT RMSE | Temporal RMSE | BT time (s) | Temporal time (s) |
|---|---|---|---|---|---|---|
| 1 | $13 \times 13$ | 19 | 2.35 | - | 0.603 | - |
| 2 | $13 \times 13$ | 19 | 1.69 | 1.69 | 0.5694 | 0.3903 |

Our approach is quantitatively evaluated by computing the root mean square error

($RMSE$) of the detected feature's disparity estimates as follows:

$$RMSE = 100 \times \sqrt{\frac{1}{N_f} \sum_{f \in \mathcal{F}} (d_f - d_g(f))^2}, \tag{5.2}$$

where $f$ is a detected feature point in the set $\mathcal{F}$, $N_f$ is the number of detected feature points, $d_f$ is the estimated disparity of a feature, and $d_g$ is the ground truth disparity of the detected feature.

The computational time of a local-matching algorithm is directly proportional to the window size as discussed in Section 4.1.4. We experimented on different window sizes on the temporal seeding algorithm and analyzed the error. Figure 5.4 shows the effect of different window sizes used for temporal seeding. The results show that the optimal window size is $13 \times 13$ because it gives the lowest $RMSE$. The window size is similar to BT's stereo algorithm's optimal window size.

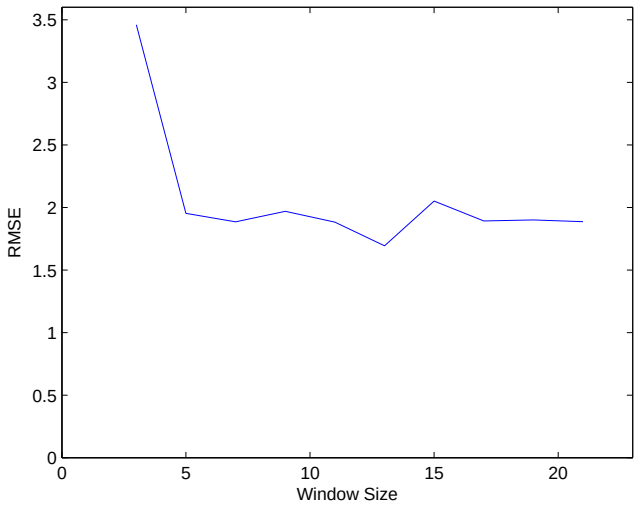



Figure 5.4: Plot of window size versus $RMSE$ for temporal seeding.

Table 5.1 shows the results obtained on the stereo frames in Figure 5.3 on 313 detected KLT features. The first column shows the frame number of the stereo sequence. The second and third columns show the window size and the number of disparities used in the experiment. The fourth column shows the RMSE for BT's stereo algorithm while the fifth column shows the RMSE when using temporal seeding. The sixth and seventh columns show the computational times for BT's stereo algorithm and the temporal seeding algorithm respectively. The results show that

our temporal seeding approach yields the same error as a fully exhaustive disparity search. The computational complexity of a stereo algorithm is directly proportional to the disparity search range, so decreasing the search range decreases the computational cost as shown in Table 5.1.

The quality of the startup disparity estimates are assumed to be accurate. This means if the initial disparity estimates are incorrect, then the temporal seeding algorithm might give inaccurate results. Therefore the algorithm suffers from error propagation. Nevertheless, the assumptions made in Section 5.5.1 give good results.

## 5.7 Further experiments

In this section a different dataset is used to further evaluate temporal seeding. The dataset used is the Microsoft i2i chairs dataset [77] which consists of a stereo video of an indoor scene with chairs and tables. Figure 5.5 shows a stereo image pair of the dataset.

(a) (b)

Figure 5.5: First stereo image pair of the Microsoft i2i chairs dataset.

The experiments carried out on this section are on 87 successive stereo image frames. Similar to Section 5.6.2, features are detected then tracked on the left images of the sequence. The disparity range selected for the stereo algorithm is $d = [0, 20]$. To quantitatively evaluate the results, the $RMSE$ is computed using Equation 5.2. The temporal seeding algorithm is initialised by the disparity estimates of BT's stereo algorithm on the first frame of the image sequence. Temporal seeding is then performed on the successive frames. Seed values for frame number $k$ are taken from frame number $k - 1$.

Figure 5.6 shows the number of features which were successfully tracked along the stereo sequence. Features which were not successfully tracked are replaced by new features. Although some features are replaced, features which lie close to the border of the image are removed. Further, features with a low reliability of being tracked are also removed. This causes the number of features to drop as the number of

frames increase.

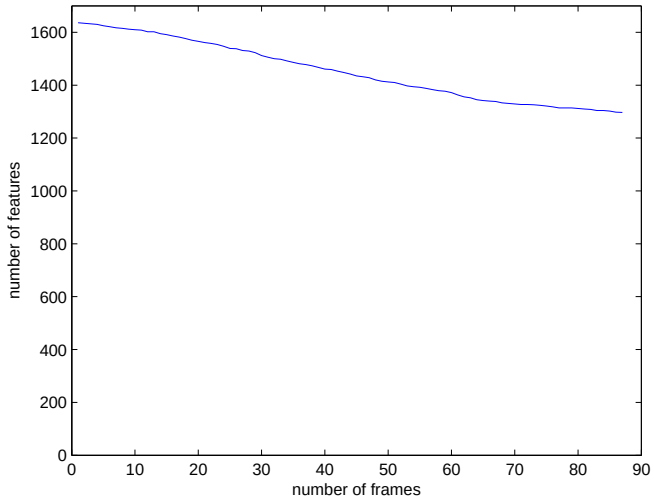



Figure 5.6: Number of frames versus the number of features in temporal seeding.

Figure 5.7 shows the $RMSE$ for every stereo frame in the sequence. The results show that the error increases as the number of frames increase. This means that temporal seeding suffers from error propagation. The results also show that the error propagation rate decreases as the number of frames increase. This might be caused by the fact that features which cannot be tracked are replaced by new features. As the sequence progresses, more features fall away because parts of the scene fall out of the field of view of the camera. Furthermore, the lighting of the scene changes because of the change in viewpoint. Some of the features which fall away are erroneous and are replaced with new features which have not suffered from error propagation. The computational time when using temporal seeding on the stereo sequence is approximately 20% less than that of BT's stereo algorithm. The improvement in speed is achieved because the developed temporal seeding approach does not always do the full disparity search. It only searches for a local minimum around the initial estimate.

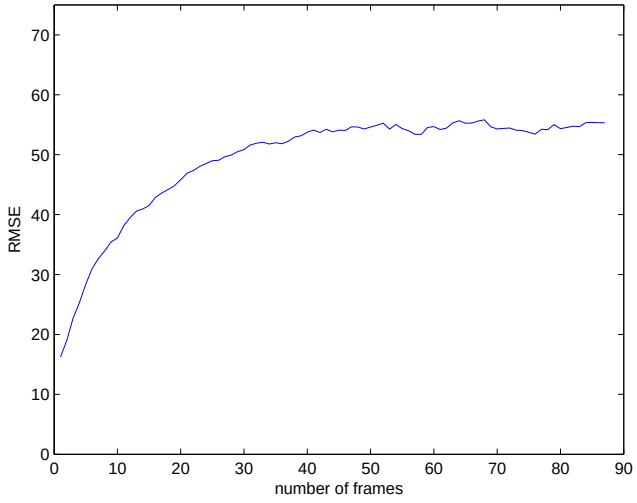



Figure 5.7: Number of frames versus RMSE in temporal seeding.

Chapter 6

# Confidence measure

A method of assigning a confidence to a disparity estimate for local-matching algorithms is presented. The confidence measure is expected to give low confidences to disparity estimates in textureless regions, where many local-matching algorithms fail. While the measure is similar to a number of previously developed confidence measures, in that the confidence of a disparity estimate is a by-product of the matching process, our analysis focuses on the basin of convergence (refer to Figure 6.2) of a disparity estimate.

To evaluate the confidence measure, a local-matching algorithm is implemented. The confidence measure is expected to be applicable across the different variations of these algorithms because of the uniform structure of the local-matching process. The algorithm is applied on the widely-used Middlebury dataset [1] in order to evaluate the performance of the confidence measure using the developed evaluation scheme.

The remainder of this section is structured as follows. Section 6.1 briefly covers the related literature. Section 6.2 discusses the local-matching algorithm implementation. Section 6.3 discusses the confidence measure and it is used for disparity refinement. Section 6.4 discusses the evaluation methodology and the results of experiments. In Section 6.5 the confidence measure is used in temporal seeding.

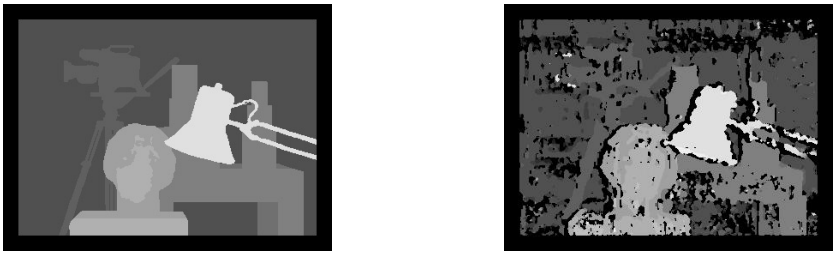
## 6.1 Related work

In stereo vision research, there have been several approaches to assign a confidence to a disparity estimate. The left-right consistency constraint [37, 38, 61, 62, 78] has traditionally been used to characterize pixel ambiguity. This constraint checks a left image disparity estimate and compares it to the inverse mapping of a right image disparity estimate. The approach is successful in detecting occluded regions. There have been approaches that analyze the matching score of the disparity estimate [79, 80]. The confidence of a pixel is based on the magnitude of the similarity value between the pixel in the left image and the matching pixel in the right image. Other approaches analyze the curvature of the correlation curve [28, 29] and assign low confidences to disparity estimates resulting from a flat correlation curve. Our approach is similar, as we also analyze the correlation curve. Approaches such as [81, 82] estimate the confidences of pixels with two similar match candidates. Research has also been conducted in determining pixel confidence based on image

entropy [83, 84]. Low confidence scores are assigned to low entropy points in the left image. Recently, a new approach has been developed that extrapolates confidence *a posteriori* from an initially given and possibly noisy disparity estimate [85].

## 6.2 Stereo algorithm

For our purposes the same stereo algorithm presented in Section 5.4 is used. The left-right consistency check is also performed in order to detect occluded regions and filter out the erroneous pixels. The dense disparity map produced by the stereo algorithm for the Tsukuba image pair with a $5 \times 5$ aggregation window and 15 disparities followed by a $5 \times 5$ median filter is shown in Figure 6.1. It should be noted that the algorithm implemented is to be used as a testbed for the confidence measure and is not meant to be compared with the state of the art.



(a) Ground truth disparity map

(b) Disparity map obtained using Birchfield and Tomasi's sampling insensitive cost with the left-right consistency check

Figure 6.1: Dense disparity map of the Tsukuba image pair using Birchfield and Tomasi's sampling insensitive cost.

## 6.3 Confidence measure for local-matching stereo algorithms

The confidence measure is calculated as a function of $(u, v, d)$, where $(u, v)$ are the image coordinates and $d$ is the disparity. A typical correlation curve is shown in Figure 6.2. Local-matching algorithms aim to find the disparity that minimizes the error represented by this curve. Given a disparity $d$, we propose computing the confidence of a disparity estimate as follows:

$$C_d = \frac{B(d)}{d_{max} - d_{min}}. \tag{6.1}$$

Here $C_d$ is the confidence for a given disparity, $B(d)$ is the basin of convergence (refer to Figure 6.2) of the disparity estimate $d$, and $d_{max} - d_{min}$ is the disparity range as in Section 4.1.3. It is expected that in textureless regions the correlation curve will have multiple local minima with small $B(d)$ values, and since $C_d$ is proportional to $B(d)$ we expect low confidences. A high confidence value would have few local minima in the correlation curve and a fully confident disparity estimate would arise where the local minimum is the global minimum of the correlation curve.

The value of $B(d)$ is determined by using an approach similar to gradient ascent. Given a disparity estimate $d$, the gradient to the right of $d$ is expected to be positive and the gradient to the left of $d$ is expected to be negative. The algorithm takes single steps on both sides of $d$ until the sign of the gradient changes on both sides. This represents the local maxima on the left and on the right of $d$. The disparity range covered by the two local maxima is defined as the basin of convergence $B(d)$.

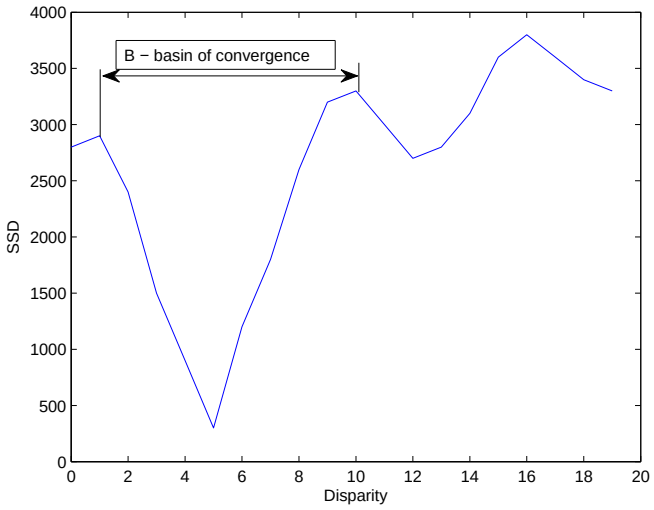



Figure 6.2: Correlation curve and the basin of convergence.

### 6.3.1 Disparity refinement

After computing confidences for all the disparity estimates, a threshold $T$ which gives the lowest errors compared to the ground truth is selected to create a mask of acceptable and unacceptable estimates. Acceptable disparity estimates are defined as those satisfying $C_d > T$. The refined disparity map for $T = \frac{2}{15}$ is shown in Figure 6.3. One can visually see that most of the noisy estimates arising from the

local-matching algorithm are successfully filtered out.

(a) Disparity map before refinement.

(b) Disparity map with disparity estimates of $C_d > \frac{2}{15}$

Figure 6.3: Disparity map with disparity estimates of $C_d > \frac{2}{15}$. Estimates which do not satisy the condition, $C_d > \frac{2}{15}$ are filtered out.

## 6.4 Experiments

The Middlebury stereo benchmark provides a testbed to quantitatively evaluate stereo algorithms. Although the testbed is widely used in the computer vision community, it requires a dense disparity map. Generally algorithms that perform disparity refinement would also include a hole-filling step. Our algorithm does not perform hole filling because of the errors it might introduce, which leaves a sparse disparity map. Evaluating the sparse disparity map on the Middlebury stereo benchmark would not be appropriate because most errors would arise from the filtered-out disparities. Thus our own evaluation scheme is used.

Pixels are classified as containing no information, unreliable information, or good information. Occluded pixels are defined as containing no information, pixels with $C_d \leq T$ as containing unreliable information, and the rest of the pixels as containing good information. In this evaluation only pixels containing good information are considered.

The root mean square error ($RMSE$) is calculated as follows:

$$RMSE = 100 \times \sqrt{\frac{1}{N_p} \sum_{(u,v) \in p} (d(u,v) - d_g(u,v))^2},$$

where $p$ is the set of all pixels containing good information, $N_p$ is the number of pixels in $p$, $d(u,v)$ is the estimated disparity at pixel $(u,v)$, and $d_g(u,v)$ is the ground

Table 6.1: $RMSE$ with a chosen window size, number of disparities, and threshold $T = 0$ for the Middlebury dataset.

| Image pair | Window size | Number of disparities | $T$ | $RMSE$ |
|---|---|---|---|---|
| Tsukuba | $5 \times 5$ | 15 | 0 | 7.56 |
| Venus | $5 \times 5$ | 19 | 0 | 29.84 |
| Teddy | $9 \times 9$ | 59 | 0 | 37.43 |
| Cones | $9 \times 9$ | 59 | 0 | 30.13 |

Table 6.2: $RMSE$ for a chosen window size, an empirically selected $T$ value and the percentage computational overhead for the Middlebury dataset.

| Image pair | Window size | T | RMSE | Computational overhead (%) |
|---|---|---|---|---|
| Tsukuba | $5 \times 5$ | $\frac{2}{15}$ | 6.56 | 2.33 |
| Venus | $5 \times 5$ | $\frac{4}{19}$ | 21.94 | 18.35 |
| Teddy | $9 \times 9$ | $\frac{2}{59}$ | 32.79 | 19.23 |
| Cones | $9 \times 9$ | $\frac{6}{59}$ | 23.38 | 17.60 |

truth disparity at pixel $(u, v)$.

The Middlebury dataset is used for evaluation. The results for this dataset on the different image pairs with $T = 0$ are shown in Table 6.1. This table shows the image pair used from the Middlebury dataset in the first column. The second and third columns show the window size and the disparity search range used. The last two columns show the $T$ values and the $RMSE$. Table 6.2 shows results where the value of $T$ is empirically selected based on a value giving the lowest $RMSE$. The table also shows the percentage computational overhead. This is the extra percentage of computational time required for a chosen value of $T$ compared to a value of $T = 0$.

In the experiments it is noted that the $RMSE$ starts increasing after a certain value of $T$ for a selected window size. This is due to the errors introduced by the window size. Local-matching stereo algorithms assume constant disparity throughout the aggregation window, so errors known as the "foreground fattening" effect [2] arise. Also, since the images have pixel resolution, a window size greater than a pixel affects the resolution of the disparity estimates. Errors are introduced where the image details are smaller than the window size. Since the algorithm does not filter out these errors, they are fixed with a changing value of $T$. The larger the value of $T$, the smaller the value of $N_p$ while the errors remain fixed. A plot showing the relationship between $T$ and $N_p$ with a $5 \times 5$ window size for the Tsukuba image pair is shown in Figure 6.4.

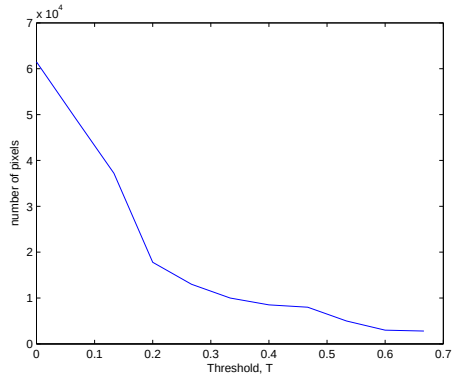



Figure 6.4: Number of pixels $N_p$ versus Threshold ($T$) with a $5 \times 5$ window size for the Tsukuba image pair.

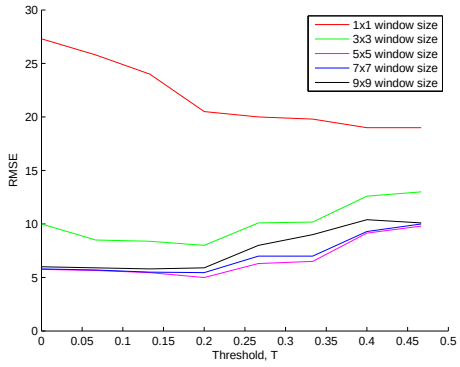



Figure 6.5: Threshold $T$ versus RMSE for varying window sizes.

To show the effect of the window size on the evaluation, Figure 6.5 contains a plot of $T$ versus $RMSE$ for varying window sizes. Different window sizes tend to shift the curve up or down. As the window size increases, the curve shifts downwards until a point where a larger window introduces more errors, causing the curve to shift upwards.

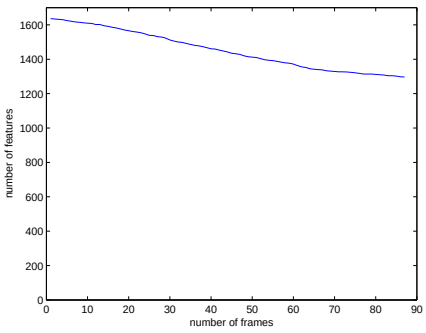
## 6.5 Confidence measure in temporal seeding

In Chapter 5 temporal seeding is applied on a stereo image sequence. The startup disparities for temporal seeding are calculated using a full disparity search on the KLT features of the left image in the first stereo image frame. In this section, the

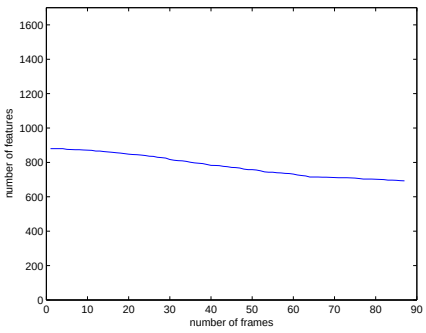
confidences of the detected features in the first stereo image frame are determined. The features are then selected according to their confidences and used as the startup features. Temporal seeding is then applied and the results are evaluated as in Chapter 5.

The results of the experiments are shown below. Figure 6.6 shows the number of features which satisfy a particular confidence. For the same reason as in Section 5.7, the number of tracked features decreases as the number of frames progresses. Also, the number of startup features are high for low confidence values and low for high confidence values.

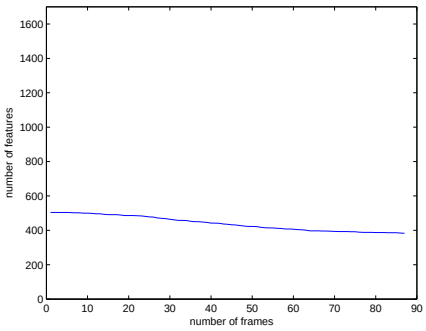
Figure 6.7 shows the $RMSE$ for every frame when using startup disparities with a particular confidence. The results show that the shapes of the Figures 6.7(a)-6.7(e) are similar meaning that the confidence measure is not removing enough erroneous points in temporal seeding. The peak error becomes higher as $C_d$ increases. This might be caused by the fact that features with a higher $C_d$ value are tracked for longer in the image sequence. This means the error is propagated longer in the sequence leading to a higher peak error. Figure 6.7(e) appears to have more noise compared to the other plots. This is due to the low number of features which have a confidence value of 1.



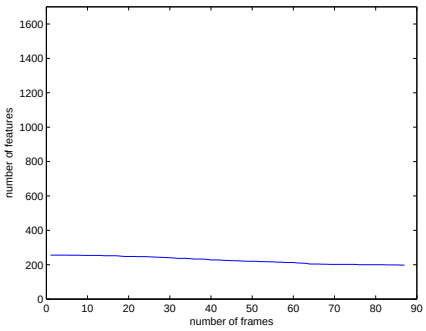

(a) Number of features with $C_d \geq \frac{0}{20}$ detected on the stereo image sequence



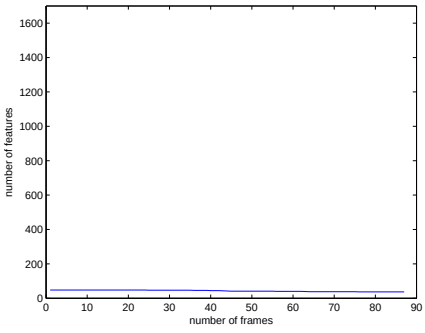

(b) Number of features with $C_d \geq \frac{5}{20}$ detected on the stereo image sequence




(c) Number of features with $C_d \geq \frac{10}{20}$ detected on the stereo image sequence

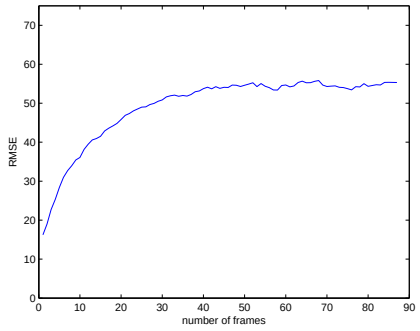



(d) Number of features with $C_d \geq \frac{15}{20}$ detected on the stereo image sequence

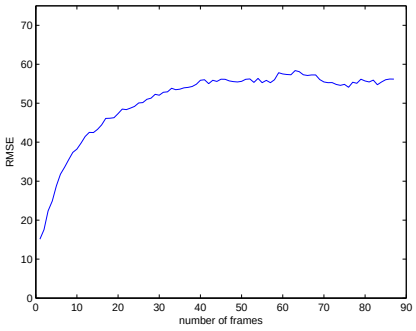



(e) Number of features with $C_d \geq \frac{20}{20}$ detected on the stereo image sequence

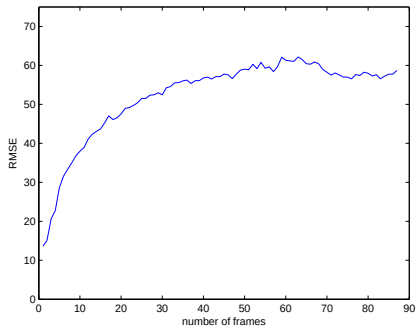
Figure 6.6: Number of features detected on the stereo image sequence which have a chosen confidence.



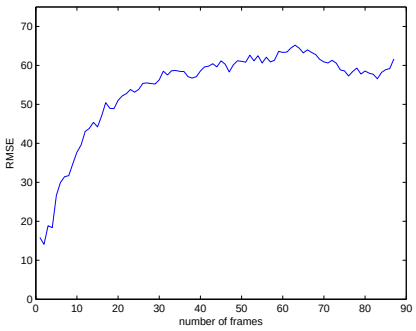

(a) RMSE for features with $C_d \geq \frac{0}{20}$ for the stereo image sequence



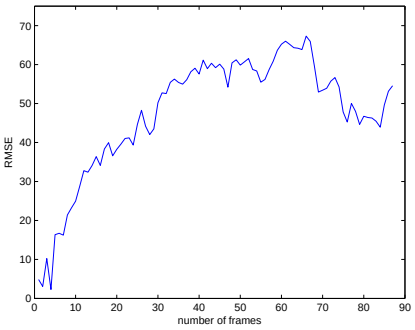

(b) RMSE for features with $C_d \geq \frac{5}{20}$ for the stereo image sequence




(c) RMSE for features with $C_d \geq \frac{10}{20}$ for the stereo image sequence




(d) RMSE for features with $C_d \geq \frac{15}{20}$ for the stereo image sequence




(e) RMSE for features with $C_d \geq \frac{20}{20}$ for the stereo image sequence

Figure 6.7: RMSE versus the number of frames for different confidence values.

Chapter 7

# Conclusions

Section 7.1 summarizes the main topics and the results of this work. Section 7.2 discusses possible future work.

## 7.1 Conclusions

There are two main objectives of this work. The first objective is to investigate temporal seeding in stereo image sequences. This is discussed in Chapter 5. The second objective is to develop a confidence measure for local-matching stereo algorithms discussed in Chapter 6.

The stereo vision problem is discussed sequentially. It is pointed out that the main problem with stereo vision is the stereo correspondence problem. Relevant literature for this problem is discussed and the solution chosen to be explored is the use of local-matching stereo algorithms. These algorithms have a uniform structure and are applicable in real-time systems. They can also be broken down into components which allows us to highlight the different design considerations. The usefulness of breaking down the algorithm into its components is discussed in Chapter 4. In order to meet the objectives, a central part in the stereo correspondence formulation in local-matching algorithms namely the correlation curve was investigated. The analysis of this curve provided interesting and useful information.

### 7.1.1 Temporal seeding

Chapter 5 presents a way of reusing computed disparity estimates of KLT features in a stereo image sequence. The temporal seeding approach developed makes use of the correlation curves of tracked KLT features. Two experiments are conducted. Firsly, only two stereo image frames are used. The results for this experiment show that temporal seeding produces the same error as the implemented stereo algorithm but does improve the computational overhead by approximately 30%.

The second experiment is done on 87 frames of a stereo image sequence. Features are tracked throughout the sequence and features which cannot be tracked are replaced by new ones. The temporal seeding is evaluated and the results show that the RMSE increases as the number of frames increase. This is caused by error

propagation. This means that the quality of the initial disparity estimates has to be high to avoid error propagation. The error propagation rate decreases as the number of frames increase. A possible cause for this is the fact that features which cannot be tracked are replaced with new features. The computational time of the temporal seeding algorithm is approximately 20% faster than Birchfield and Tomasi's stereo algorithm.

### 7.1.2 Confidence measure

Chapter 6 presents a confidence measure to detect textureless regions for local-matching algorithms. The confidence measure is formulated by analysing a property of the correlation curve called the basin of convergence. The effectiveness of this approach is demonstrated by implementing a local-matching algorithm and filtering out unreliable depth estimates. The quantitative evaluation demonstrated that the confidence measure decreases the disparity estimate errors at a small computational cost.

Further experiments involve using the confidence measure in temporal seeding. The detected KLT features are selected based on their confidence. These features are then used as start-up features for temporal seeding. The results show that the confidence measure does not succeed in removing features which produce high errors.

## 7.2 Future work

The main problem with temporal seeding is error propagation. This can be overcome by performing the temporal seeding process for a certain number of frames then using the stereo algorithm to determine new start-up disparities. Also, the temporal seeding approach is done on KLT features which means that a sparse disparity map is obtained. Ultimately, it would be very useful to have a denser disparity map. Instead of using the confidence measure on KLT features, one might consider using it as a feature detector. This would provide denser results for temporal seeding.

The temporal seeding approach developed is for local-matching stereo algorithms. Developing a temporal seeding approach for slower algorithms such as graph cuts will aid in decreasing the computational time in the hope of making the algorithm run in real-time.

Since the temporal seeding approach is used in a sparse context, it may be useful in applications such as motion estimation. Exploring such an application might result in a faster algorithm.

# Bibliography


[1] D. Scharstein and R. Szeliski, "Middlebury stereo vision page," August 2009, `vision.middlebury.edu/stereo/`. vii, 2, 21, 45

[2] D. Scharstein, R. Szeliski, and R. Zabih, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," in *Proceedings of the IEEE Workshop on Stereo and Multi-Baseline Vision*. Washington, DC, USA: IEEE Computer Society, 2001, p. 131. 1, 3, 6, 7, 21, 29, 49

[3] M. Z. Brown, D. Burschka, and G. D. Hager, "Advances in computational stereo," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 993–1008, August 2003. 1

[4] L. Linhui, Z. Mingheng, G. Lie, and Z. Yibing, "Stereo vision based obstacle avoidance path-planning for cross-country intelligent vehicle," in *Proceedings of the 6th international conference on Fuzzy systems and knowledge discovery*, ser. FSKD'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 463–467. 1

[5] T. Lemaire, C. Berger, I.-K. Jung, and S. Lacroix, "Vision-based slam: Stereo and monocular approaches," *International Journal on Computer Vision*, vol. 74, pp. 343–364, September 2007. 1

[6] M. Kanbara, H. Fujii, H. Takemura, and N. Yokoya, "A stereo vision-based augmented reality system with an inertial sensor," *International Symposium on Augmented Reality*, vol. 0, p. 97, 2000. 1

[7] V. I. Pavlovic, R. Sharma, and T. S. Huang, "Visual interpretation of hand gestures for human-computer interaction: A review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 677–695, July 1997. 2

[8] L. Stefano, "A fast area-based stereo matching algorithm," *Image and Vision Computing*, vol. 22, no. 12, pp. 983–1005, 2004. 3, 25

[9] L. Di Stefano, M. Marchionni, and S. Mattoccia, "A pc-based real-time stereo vision system," *International Journal on Machine Graphics and Vision*, vol. 13, pp. 197–220, January 2004. 3, 25

[10] F. Tombari, S. Mattoccia, and L. Di Stefano, "Segmentation-based adaptive support for accurate stereo correspondence," in *Proceedings of the 2nd Pacific Rim conference on Advances in image and video technology*, ser. PSIVT'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 427–438. 4, 23, 31

[11] R. Yang, M. Pollefeys, and S. Li, "Improved real-time stereo on commodity graphics hardware," in *Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop*, vol. 3. Washington, DC, USA: IEEE Computer Society, 2004, pp. 36–. 4

[12] J. Woodfill and B. Von Herzen, "Real-time stereo vision on the parts reconfigurable computer," in *Proceedings of the 5th IEEE Symposium on FPGA-Based Custom Computing Machines*. Washington, DC, USA: IEEE Computer Society, 1997, pp. 201–. 4

[13] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 23, pp. 1222–1239, November 2001. 4

[14] S. Roy and I. J. Cox, "A maximum-flow formulation of the n-camera stereo correspondence problem," in *Proceedings of the Sixth International Conference on Computer Vision*, ser. ICCV '98. Washington, DC, USA: IEEE Computer Society, 1998, pp. 492–. 4

[15] V. Kolmogorov and R. Zabih, "Computing visual correspondence with occlusions using graph cuts," *IEEE International Conference on Computer Vision*, vol. 2, p. 508, 2001. 4

[16] O. Veksler, "Efficient graph-based energy minimization methods in computer vision," Ph.D. dissertation, Ithaca, NY, USA, 1999, aAI9939932. 4

[17] R. B. Potts, "Some Generalized Order-Disorder Transformation," in *Transformations, Proceedings of the Cambridge Philosophical Society*, vol. 48, 1952, pp. 106–109. 4

[18] A. F. Bobick and S. S. Intille, "Large occlusion stereo," *International Journal of Computer Vision*, vol. 33, pp. 181–200, September 1999. 5

[19] I. J. Cox, S. L. Hingorani, S. B. Rao, and B. M. Maggs, "A maximum likelihood stereo algorithm," *Computer Vision and Image Understanding*, vol. 63, pp. 542–567, May 1996. 5

[20] D. Geiger, B. Ladendorf, and A. Yuille, "Occlusions and binocular stereo," *International Journal of Computer Vision*, vol. 14, pp. 211–226, April 1995. 5

[21] A. L.Yuille and T. Poggio, "A generalized ordering constraint for stereo correspondence," MIT, A.I.Memo 777, 1984, aI Lab. 7, 27

[22] Z.-F. Wang and Z.-G. Zheng, "A region based stereo matching algorithm using cooperative optimization," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 0, pp. 1–8, 2008. 7, 21

[23] A. Klaus, M. Sormann, and K. Karner, "Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure," in *Proceedings of the 18th International Conference on Pattern Recognition*, ser. ICPR '06, vol. 3. Washington, DC, USA: IEEE Computer Society, 2006, pp. 15–18. 7, 21

[24] H. Hirschmuller, P. R. Innocent, and J. Garibaldi, "Real-time correlation-based stereo vision with reduced border errors," *International Journal of Computer Vision*, vol. 47, pp. 229–246, April 2002. 8

[25] T. Kanade and M. Okutomi, "A stereo matching algorithm with an adaptive window: Theory and experiment." *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 8, 29

[26] J. C. Kim, K. M. Lee, B. T. Choi, and S. U. Lee, "A dense stereo matching using two-pass dynamic programming with generalized ground control points," in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, ser. CVPR '05, vol. 2. Washington, DC, USA: IEEE Computer Society, 2005, pp. 1075–1082. 8

[27] F. Tombari, S. Mattoccia, L. di Stefano, and E. Addimanda, "Near real-time stereo based on effective cost aggregation," in *International Conference on Pattern Recognition*, 2008, pp. 1–4. 10, 24

[28] P. Anandan, "Computing dense displacement fields with confidence measures in scenes containing occlusion," *Image Understanding Workshop*, vol. 84, pp. 236–246, 1984. 10, 45

[29] ——, "A computational framework and an algorithm for the measurement of visual," Amherst, MA, USA, Tech. Rep., 1987. 10, 45

[30] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004. 12, 16

[31] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 1330–1334, November 2000. 13

[32] D. G. R. Bradski and A. Kaehler, *Learning opencv*, 1st ed. O'Reilly Media, Inc., 2008. 13

[33] J. Y. Bouguet, "Camera Calibration Toolbox for Matlab," 2008. [Online]. Available: http://www.vision.caltech.edu/bouguetj/calib_doc/. 13, 14, 17

[34] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proceedings of the 4th Alvey Vision Conference*, 1988, pp. 147–151. 13

[35] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 603–619, May 2002. 21

[36] Q. Yang, L. Wang, R. Yang, H. Stewénius, and D. Nistér, "Stereo matching with color-weighted correlation, hierarchical belief propagation, and occlusion handling," *IEEE Transactions of Pattern Analysis and Machine Intelligence*, vol. 31, pp. 492–504, March 2009. 21, 22

[37] G. Egnal and R. P. Wildes, "Detecting binocular half-occlusions: Empirical comparisons of five approaches," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 8, pp. 1127–1133, 2002. 21, 30, 45

[38] K. Konolige, "Small vision systems: hardware and implementation," in *Eighth International Symposium on Robotics Research*, 1997, p. 111–116. 21, 30, 45

[39] L. Xu and J. Jia, "Stereo matching: An outlier confidence approach," in *European Conference on Computer Vision*. Springer, 2008, pp. 775–787. 22

[40] Y. Taguchi, B. Wilburn, and C. L. Zitnick, "Stereo reconstruction with mixed pixels using adaptive over-segmentation," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2008. 22

[41] J. Sun, Y. Li, S. B. Kang, and H.-Y. Shum, "Symmetric stereo matching for occlusion handling," in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, ser. CVPR '05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 399–406. 22

[42] Q. Yang, C. Engels, and A. Akbarzadeh, "Near real-time stereo for weakly-textured scenes," in *Proceedings of the British Machine Vision Conference*, 2008, pp. 80–87. 22

[43] Z. Gu, X. Su, Y. Liu, and Q. Zhang, "Local stereo matching with adaptive support-weight, rank transform and disparity calibration," *Pattern Recognition Letters*, vol. 29, pp. 1230–1235, July 2008. 22

[44] H. Hirschmuller, "Stereo vision in structured environments by consistent semi-global matching," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 2386–2393, 2006. 23

[45] S. Mattoccia, F. Tombari, and L. Di Stefano, "Stereo vision enabling precise border localization within a scanline optimization framework," in *Proceedings of the 8th Asian conference on Computer vision*, ser. ACCV'07, vol. 2. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 517–527. 23

[46] K.-J. Yoon and I. S. Kweon, "Stereo matching with the distinctive similarity measure," *IEEE International Conference on Computer Vision*, pp. 1–7, 2007. 23

[47] C. Lei, J. Selzer, and Y.-H. Yang, "Region-tree based stereo using dynamic programming optimization," in *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, ser. CVPR '06, vol. 2. Washington, DC, USA: IEEE Computer Society, 2006, pp. 2378–2385. 23

[48] K.-J. Yoon and I. S. Kweon, "Adaptive support-weight approach for correspondence search," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, pp. 650–, April 2006. 23

[49] A. Bhusnurmath and C. J. Taylor, "Solving stereo matching problems using interior point methods," in *Fourth International Symposium on 3D Data Processing, Visualization and Transmission*, O. S. Stefan Gumhold, Jana Kosecka, Ed., June 2008, pp. 321–329. 23

[50] S. K. Gehrig and U. Franke, "Improving stereo sub-pixel accuracy for long range stereo," *IEEE International Conference on Computer Vision*, pp. 1–7, 2007. 23

[51] H. Hirschmuller, "Accurate and efficient stereo processing by semi-global matching and mutual information," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 807–814, 2005. 24

[52] Q. Yang, L. Wang, R. Yang, S. Wang, M. Liao, and D. Nistér, "Real-time global stereo matching using hierarchical belief propagation," in *Proccedings of the British Machine Vision Conference*. British Machine Vision Association, 2006, pp. 989–998. 24

[53] T. Liu, P. Zhang, and L. Luo, "Dense stereo correspondence with contrast context histogram, segmentation-based two-pass aggregation and occlusion handling," in *Proceedings of the 3rd Pacific Rim Symposium on Advances in Image and Video Technology*, ser. PSIVT '09. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 449–461. 24

[54] J. Lu, G. Lafruit, and F. Catthoor, "Anisotropic local high-confidence voting for accurate stereo correspondence." in *Image Processing: Algorithms and Systems*, 2008, p. 1. 24

[55] P. Mordohai and G. Medioni, "Stereo using monocular cues within the tensor voting framework," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, pp. 968–982, June 2006. 24

[56] L. Wang, M. Liao, M. Gong, R. Yang, and D. Nister, "High-quality real-time stereo using adaptive cost aggregation and dynamic programming," in *Proceedings of the Third International Symposium on 3D Data Processing, Visualization, and Transmission*, ser. 3DPVT '06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 798–805. 24

[57] O. Veksler, "Stereo correspondence by dynamic programming on a tree," in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, ser. CVPR '05, vol. 2. Washington, DC, USA: IEEE Computer Society, 2005, pp. 384–390. 24

[58] S. Birchfield and C. Tomasi, "A pixel dissimilarity measure that is insensitive to image sampling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 4, pp. 401–406, 1998. 26, 36

[59] H. Hirschmuller and D. Scharstein, "Evaluation of stereo matching costs on images with radiometric differences," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 9, pp. 1582–1599, 2009. 26

[60] L. Wang, M. Gong, M. Gong, and R. Yang, "How far can we go with local optimization in real-time stereo matching," in *Proceedings of the Third International Symposium on 3D Data Processing, Visualization, and Transmission*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 129–136. 28

[61] J. J. Little and W. E. Gillett, "Direct evidence for occlusion in stereo and motion," in *Proceedings of the First European Conference on Computer Vision*. London, UK: Springer-Verlag, 1990, pp. 336–340. 30, 45

[62] A. Luo and H. Burkhardt, "An intensity-based cooperative bidirectional stereo matching with simultaneous detection of discontinuities and occlusions," *International Journal of Computer Vision*, vol. 15, no. 3, pp. 171–188, 1995. 30, 45

[63] Q. Yang, R. Yang, J. Davis, and D. Nister, "Spatial-depth super resolution for range images," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2007. 30

[64] S. W. Zucker and R. A. Hummel, "Toward a Low-level Description of Dot Clusters: Labeling Edge, Interior, and Noise Points," *Computer Vision, Graphics, and Image Processing*, vol. 9, no. 3, pp. 213–233, March 1979. 31

[65] N. Ahuja and M. Tuceryan, "Extraction of early perceptual structure in dot patterns: integrating region, boundary, and component gestalt," *Computer Vision, Graphics, and Image Processing*, vol. 48, pp. 304–356, December 1989. 31

[66] N. Thacker and P. Courtney, "Statistical analysis of a stereo matching algorithm," in *Proceedings of the British Machine Vision Conference*, 1992, pp. 316–326. 33

[67] L. Matthies and M. Okutomi, "Bootstrap algorithms for dynamic stereo vision," in *Proceedings of the 6th Multidimensional Signal Processing Workshop*, 1989, p. 12. 34

[68] A. Dalmia and M. Trivedi, "High speed extraction of 3d structure of selectable quality using a translating camera," *Computer Vision and Image Understanding*, vol. 64, no. 1, pp. 97–110, 1996. 34

[69] S. T. G. Xu and M. Asada, "A motion stereo method based on coarse to fine control strategy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, no. 2, pp. 332–336, 1987. 34

[70] B. C. L. Zhang and S. Seitz, "Spacetime stereo: Shape recovery for dynamic scenes," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2003, pp. 367–374. 34

[71] R. R. S. R. J. Davis, D. Nehab, "Spacetime stereo : A unifying framework for depth from triangulation," *IEEE Transaction On Pattern Analysis and Machine Intelligence*, vol. 27, no. 2, 2005. 34

[72] N. T. S. Crossley and N. Seed, "Benchmarking of bootstrap temporal stereo using statistical and physical scene modelling," in *Proceedings of the British Machine Vision Conference*, 1998, pp. 346–355. 34

[73] N. T. S. Crossley, A.J. Lacey and N. Seed, "Robust stereo via temporal consistency," in *Proceedings of the British Machine Vision Conference*, 1997, pp. 659–668. 34

[74] D. B. L. Kanade and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *International Joint Conference on Artificial Intelligence*, 1981, pp. 674–679. 35

[75] C. Tomasi and T. Kanade, "Detection and tracking of point features," Tech. Rep., 1991. 36

[76] J. Shi and C. Tomasi, "Good features to track," in *IEEE Conference on Computer Vision and Pattern Recognition*, 1994, pp. 593–600. 36

[77] "Microsoft i2i dataset," `http://research.microsoft.com/en-us/projects/i2i`. 41

[78] R. Trapp, S. Drüe, and G. Hartmann, "Stereo matching with implicit detection of occlusions," in *Proceedings of the 5th European Conference on Computer Vision*, vol. 2. London, UK: Springer-Verlag, 1998, pp. 17–33. 45

[79] M. J. Hannah, "Computer matching of areas in stereo images." Ph.D. dissertation, Stanford, CA, USA, 1974. 45

[80] D. Smitley and R. Bajcsy, "Stereo processing of aerial, urban images," *International Conference on Pattern Recognition*, vol. 84, pp. 433–435, 1984. 45

[81] J. J. Little and W. E. Gillett, "Direct evidence for occlusion in stereo and motion," in *Proceedings of the First European Conference on Computer Vision*. London, UK: Springer-Verlag, 1990, pp. 336–340. 45

[82] D. Scharstein, "View synthesis using stereo vision," Ph.D. dissertation, Ithaca, NY, USA, 1997. 45

[83] Y. G. Leclerc, "Constructing Simple Stable Descriptions for Image Partitioning," *International Journal of Computer Vision*, vol. 3, no. 1, pp. 73–102, 1989. 46

[84] D. Samaras, D. Metaxas, P. Ascalfua, and Y. G. Leclerc, "Variable albedo surface reconstruction from stereo and shape from shading," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2000, pp. 480–487. 46

[85] R. Gherardi, "Confidence-based cost modulation for stereo matching," in *International Conference on Pattern Recognition*, 2008, pp. 1–4. 46