Automatic Volume Estimation of Timber from Multi-View Stereo 3D Reconstruction

Emil Rundgren
Master of Science Thesis in Electrical Engineering

Automatic Volume Estimation of Timber from Multi-View Stereo 3D Reconstruction

Emil Rundgren
LiTH-ISY-EX--17/5093--SE

Supervisor:  
Hannes Ovrén  
ISY, Linköpings universitet
Fredrik Lundell
Saab Dynamics AB

 Examiner:  
Per-Erik Forssén  
ISY, Linköpings universitet

Computer Vision Laboratory  
Department of Electrical Engineering  
Linköping University  
SE-581 83 Linköping, Sweden

Copyright © 2017 Emil Rundgren
Abstract

The ability to automatically estimate the volume of timber is becoming increasingly important within the timber industry. The large number of timber trucks arriving each day at Swedish timber terminals fortifies the need for a volume estimation performed in real-time and on-the-go as the trucks arrive.

This thesis investigates if a volumetric integration of disparity maps acquired from a Multi-View Stereo (MVS) system is a suitable approach for automatic volume estimation of timber loads. As real-time execution is preferred, efforts were made to provide a scalable method. The proposed method was quantitatively evaluated on datasets containing two geometric objects of known volume. A qualitative comparison to manual volume estimates of timber loads was also made on datasets recorded at a Swedish timber terminal.

The proposed method is shown to be both accurate and precise under specific circumstances. However, robustness is poor to varying weather conditions, although a more thorough evaluation of this aspect needs to be performed. The method is also parallelizable, which means that future efforts can be made to significantly decrease execution time.
Acknowledgments

First of all, I would like to thank Hans Holmgren for giving me the opportunity to do my thesis work at Saab Dynamics AB. I would also like to thank my supervisor Hannes Ovrén and examiner Per-Erik Forssén at ISY for valuable critique throughout the work.

Finally, I thank everyone at Saab Dynamics AB who have contributed with constructive comments or helped me in one way or another. Special thanks to my supervisor, Fredrik Lundell, whose support and insatiable desire to help during the thesis work have been invaluable.

Linköping, July 2017
Emil Rundgren
# Contents

1 Introduction .......................... 1
   1.1 Motivation ............................... 1
   1.2 Aims .................................. 2
   1.3 Goals .................................. 3
   1.4 Questions subject to investigation ......................... 3
   1.5 Limitations ............................. 3
   1.6 Thesis outline .......................... 3

2 Theory .................................. 5
   2.1 Related work ............................ 5
   2.2 Multi-View Stereo Reconstruction ......................... 6
      2.2.1 Camera calibration and the pinhole camera model ....... 7
      2.2.2 Feature detection and matching .......................... 9
      2.2.3 Epipolar geometry ............................. 10
      2.2.4 Stereo rectification .............................. 11
      2.2.5 Depth from disparity ............................. 12
   2.3 Implicit volume representation ........................... 15
      2.3.1 Volumetric data ............................ 15
      2.3.2 Signed Distance Function ......................... 17
   2.4 Isosurface extraction .......................... 18
      2.4.1 Marching Cubes ............................. 20

3 Method .................................. 23
   3.1 Datasets ................................ 23
      3.1.1 Experimental setup ............................ 23
      3.1.2 Real data .................................. 24
      3.1.3 Test rig data ................................ 25
   3.2 Proposed method ............................ 27
      3.2.1 Volumetric integration .......................... 27
      3.2.2 Volume estimation ............................. 39
   3.3 Evaluation ................................ 41
      3.3.1 Accuracy ............................... 41
      3.3.2 Precision ............................... 41
3.3.3 Robustness .................................................. 41

4 Results ......................................................... 43
  4.1 Test rig data .................................................. 43
     4.1.1 Box datasets ............................................. 44
     4.1.2 Pyramid datasets ........................................ 46
  4.2 Real data .................................................... 48

5 Discussion .................................................... 53
  5.1 Results ....................................................... 53
     5.1.1 Test rig data ............................................. 53
     5.1.2 Real data ................................................ 54
  5.2 Method ....................................................... 55
     5.2.1 Ambiguities of multiple stereo pairs .................. 56

6 Conclusions .................................................. 61
  6.1 Conclusions .................................................. 61
  6.2 Future work .................................................. 62

Bibliography ...................................................... 63
1

Introduction

1.1 Motivation

The problem of 3D reconstruction is well studied in computer vision and many new applications emerge as the field progresses. One such possible extension is to automatically estimate the volume of an object. By first creating a surface representation from the reconstructed 3D points, e.g. a mesh, the volume can be calculated by some means of volumetric integration. This is an application that the company Saab Dynamics AB has recently proclaimed interest for in a 3D reconstruction project involving timber trucks.

Saab Dynamics AB has an operating Multi-View Stereo (MVS) reconstruction system for 3D reconstruction of timber trucks in an outdoor environment. The system consists of three stereo pairs, one above and two on the sides, and outputs for each time instance camera poses and corresponding disparity maps of the truck and its load. A disparity map is an image where each pixel value is inversely proportional to the depth of a 3D point seen by the cameras in a stereo pair. From this information, it is a trivial task to triangulate 3D points representing discrete samples of the sought after 3D model.

To date, the volume estimation is performed by rectifying a candidate image from a side view of the truck (see figure 1.1) and measuring manually in the image. Although measuring can be performed with sub-millimeter precision, estimating a volume from a single 2D image is troublesome. Furthermore, measuring by hand is both time consuming and introduces the risk of operator bias. Naturally, a clear ambition is to automate this process and at the same time provide a volume estimate that is in line with, or better than, existing manual volume estimation performance.
Figure 1.1: Rectified side view image of a truck and its load. To date, the
volume estimation is performed by measuring manually in the image.

1.2 Aims

The purpose of this thesis was to implement and evaluate a suitable method for
automatic estimation of the gross volume of a timber load on a timber truck in
motion. More specifically, the method for volume estimation was to be performed
on data acquired from the existing MVS reconstruction system at Saab Dynamics
AB. As the reconstruction system is already industrially deployed, the demands
from Saab Dynamics AB on accuracy, precision and robustness are high. There-
fore, these demands have served as aims in this thesis, toward which the course
of the thesis work was steered. The aims are concretized in the following list.

1. Provide a method for automatic volume estimation of timber loads, that
produces a volume estimate with a systematic error less than or equal to
the existing demands on precision. The demands on precision are:

   - standard deviation $\leq 4.5\%$ for loads with logs of differing lengths,
   - standard deviation $\leq 3\%$ for loads with logs of the same length.

2. The method is robust to different weather/lighting conditions, i.e. the per-
formance thresholds in 1 are not exceeded.

3. The method is robust to all types of timber passing through Swedish timber
terminals, i.e. the performance thresholds in 1 are not exceeded.
1.3 Goals

To make the distinction between preferred and required functionality of the method, a set of goals have also been defined. The following list states these goals, which collectively should be seen as a description of the minimum level of required functionality.

1. Provide a method that creates a volumetric representation of an object reconstructed through sensor data from a passive MVS reconstruction system.

2. The above mentioned method can automatically estimate the gross volume of the reconstructed object.

3. Provide a visualization of the reconstructed geometric model.

A less tangible goal was that the volume estimation is carried out within a reasonable time. An existing demand is that the whole volume measurement process is finished within 30 seconds, which still is not satisfied to date. It remained an ambition for the thesis, however, that this threshold was not grossly exceeded. As a result of this, in order for Saab Dynamics AB to continue the work of this thesis, there was an impending desire that the chosen method is scalable. In other words, the method had to be susceptible to parallelization and enable further improvements of performance.

1.4 Questions subject to investigation

Given the prerequisites stated in section 1.2 and 1.3, the thesis also aimed at answering the following questions.

How should a suitable method for automatic volume estimation be designed and implemented, so that all stated aims and goals are fulfilled?

With the above mentioned method, what can be achieved in terms of scalability?

1.5 Limitations

Scalability is a rather ambiguous goal and is therefore a difficult property to evaluate. Hence, this goal has merely been treated as a guideline in the choice of method rather than a feature that has permeated every step of the implementation. This property is handed a brief discussion in chapter 5.

1.6 Thesis outline

The report consists of six chapters which describe different parts of how the thesis work has been carried out. In chapter 2, theoretical aspects are covered, present-
ing a short review of MVS reconstruction, the concepts of implicit volume representation and isosurface extraction. Chapter 3 describes the proposed method and how it was designed, implemented and evaluated. In chapter 4, the achieved results are presented, which are followed by a discussion in chapter 5. Finally, in chapter 6, conclusions are made in relation to the aims and goals and some remarks about future work are made.
In this chapter, theoretical aspects of the proposed method are described and a short summary of related work is given.

2.1 Related work

Volume estimation is largely associated with surface reconstruction, hence methods for surface reconstruction constitute the main area of interest in this thesis. As discussed in the state-of-the-art report of Berger et al [2], the problem of surface reconstruction can often be characterized in terms of acquisition method and shape of the object being reconstructed. These characteristics determine what assumptions can be made to aid the reconstruction, such as sampling density, level of noise, amount of missing data and misalignment. MVS systems, for example, tend to produce outliers as a result of view-dependent specularities [2].

One of the most common techniques for surface reconstruction is Poisson surface reconstruction [9]. In this method, the surface is approximated by an indicator function calculated through viewing oriented points as part of a Poisson problem. This creates a smooth surface that is robust to noise, but entails a global optimization of the point cloud that is computationally expensive and therefore compromises speed. Furthermore, as addressed in [9], the lack of acquisition modality information may result in inferior reconstructions compared to visibility based methods. This is typically true in cases of hollow regions, where sparse samples are connected to create a false surface.

The summary of different state-of-the-art techniques in [2] highlights several reasonably robust visibility based methods. One of them is the well-known work of Curless and Levoy [6], where depth images are incrementally fused to create an
implicit representation of the surface in the form of a signed distance function (SDF). In this volumetric method, the SDF is stored in a three-dimensional grid from which the surface then is extracted as the zero-level set. Although proven to be a fast and relatively memory efficient method [9], it lacks robustness to outliers and does not scale particularly well [2, 4]. Zach and Pock [16] extend the approach of [6] and increase robustness by a global optimization method. However, the increased computational complexity further solidifies the problem of poor scalability.

Scalability is generally the major challenge for volumetric methods due to the large memory footprint of the regular voxel grid. Empty space and surfaces are both represented densely, meaning larger scenes will not be reconstructed without compromising quality. Several methods therefore focus on improving the efficiency of the volumetric data structure. Chen et al. [4] use a hierarchical data structure for real-time reconstruction at scale with promising results. Niessner et al. [11] propose a spatial hashing technique which improves both memory efficiency and reconstruction quality.

### 2.2 Multi-View Stereo Reconstruction

The theory in this section, section 2.2.5 excluded, is explained with the help of [8]. The theory in section 2.2.5 is explained with the help of [14].

Reconstructing a 3D model from multiple 2D images is a classic computer vision problem. An ordinary image is a projection of the 3D scene it depicts onto a two-dimensional surface. In this process of projection, one dimension is lost to describe the scene, namely depth. From a single image it is therefore impossible to step back into the 3D world, and accurately reconstruct a model of the scene, without having any information about depth.

Most 3D reconstruction algorithms solve this issue by matching features in two images with known camera positions and estimate the corresponding 3D point by means of triangulation. This simple principle is illustrated in figure 2.1. If $y_1$ and $y_2$ are the pixel coordinates of the corresponding 3D point $x$ in two images, their projection lines will intersect at $x$. However, this is a slightly simplified example. The two images are here assumed to be generated by pinhole cameras and are also, for convenience, represented by virtual image planes placed in front of their camera centers. This is also the representation chosen for all figures containing stereo images for the rest of this thesis. In reality, several factors, including lens distortion and camera position uncertainty, will affect the estimation of $x$ and need to be accounted for.
2.2 Multi-View Stereo Reconstruction

In multi-view stereo, the stereo part refers to a configuration where two or more cameras are arranged so that their field-of-view overlap. At each imaging instance, the cameras will take their images synchronously, meaning multiple views of the scene exist for each point in time. The cameras are also typically part of the same fixed rig, so that the relative transformation between the cameras remains constant and known. This is a desirable feature when matching 2D points in the images, as will be described in section 2.2.2.

2.2.1 Camera calibration and the pinhole camera model

In order to estimate 3D points by triangulation, information about the geometry of the two cameras is needed. In other words, the intrinsic and extrinsic parameters of the cameras need to be determined, a task referred to as camera calibration. Before we get into the specifics of camera calibration, a short review of the pinhole camera model is suitable.

In computer vision, the pinhole camera serves as an approximative model to describe the camera’s mapping of 3D space to image coordinates. As mentioned in the previous section, the model does not account for the effect of for example lens distortion, or other optical aberrations, but suffices to explain the concept of projection. The pinhole camera can be viewed as a box, where light from a scene enters through a small hole (the pinhole) in the front and produces an image on the image plane of the back wall that is rotated 180 degrees (see figure 2.2). The pinhole is referred to as the camera center, and often the image is represented by a virtual image plane in front of the camera center in order to avoid the rotation effect (as is done in this thesis).

Figure 2.1: A 3D point $x$ can be estimated from two images by triangulation if its projection coordinates $y_1$ and $y_2$ in are known.
Since projection is forming a 2D representation of 3D space, some form of transformation between coordinate systems is inevitable. In the pinhole camera, the camera center is the origin in what we refer to as the camera centered coordinate system. The projection of a 3D point is then the transformation of camera centered coordinates to image coordinates, and is determined by the intrinsic camera parameters.

Consider the pinhole camera shown in figure 2.3. The projection of a 3D point \( \mathbf{x} = [x, y, z]^\top \) in camera centered coordinates to image plane coordinates \( \mathbf{y} = [u, v]^\top \) depends on the distance between the camera center \( \mathbf{n} \) and the principal point \( \mathbf{p} \), i.e. the focal length \( f \). More specifically, the projection coordinates are given by

\[
\mathbf{y} = [u, v]^\top = -\frac{f}{z} [x, y]^\top.
\]  

(2.1)

Equation 2.1 implies that the size of the resulting image is proportional to the focal length of the camera. Furthermore, the minus sign states that the image is rotated 180 degrees. If the minus sign is removed, we instead get the projection coordinates for the virtual image plane in front of the camera center.

Image coordinates are typically defined in terms of an origin placed in the upper left corner, and not in the principal point. The image plane coordinates therefore need to be translated accordingly, which is given by the principal point coordinates \( u_0 \) and \( v_0 \). For an image of size \( m \times n \), we usually have \([u_0, v_0] = [m/2, n/2]\).

The focal length and the principal point coordinates constitute the key components of the intrinsic camera parameters. In reality, several additional parameters may be included to accurately describe the camera geometry, but are disregarded here. In general, as long as the settings of the camera are unchanged (e.g. zoom), these parameters are considered to be fixed. Intrinsic camera calibration is therefore a task performed relatively seldom, but a common method for this is Zhang’s
2.2 Multi-View Stereo Reconstruction

Figure 2.3: The projection of a 3D point \( x \) to image plane coordinates \( y \) depends on the focal length \( f \).

method [17]. To describe the transformation of camera centered coordinates to image coordinates, we can form an intrinsic camera matrix. Let \( K \) be the intrinsic camera matrix, we then have

\[
K = \begin{bmatrix}
f & 0 & u_0 \\
0 & f & v_0 \\
0 & 0 & 1
\end{bmatrix}
\]

(2.2)

Note that, since \( f \) is positive, this matrix describes a transformation to the virtual image plane.

Normally, 3D points are defined in terms of some global coordinate system, which means that an additional transformation is needed, determined by the extrinsic camera parameters. The extrinsic camera parameters represent a rigid transformation from world coordinates to camera centered coordinates. The rotation \( R \) and translation \( t \) make up a \( 3 \times 4 \) transformation matrix which can be combined with \( K \) to describe the complete mapping of a 3D point to image coordinates. This combined matrix is referred to as the camera matrix. Let \( C \) be the camera matrix, we then have

\[
C = K[R | t].
\]

(2.3)

The complete mapping of a 3D point \( x_g \) in global coordinates to image coordinates \( y_h \) is then described by

\[
y_h \sim Cx_g,
\]

(2.4)

where \( \sim \) denotes the projective equivalence relation.

2.2.2 Feature detection and matching

Finding corresponding points between two images is key to an accurate 3D reconstruction. This is usually done by finding a number of candidate points, and establishing some form of correspondence measure between them. The points
with a correspondence measure greater than some threshold are then the points deemed most likely to be true correspondences.

Points, or features, with a unique local environment are by definition easier to match, and are commonly referred to as interest points. Corners are good examples of interest points, as their local neighbourhood constitute a characteristic change in intensity in multiple directions. Several methods for finding interest points exist, one of the most common being the Harris operator \[7\]. The Harris operator generates a measure based on the structure tensor, a matrix constituting the gradient of a local neighbourhood. Let \( T \) be the structure tensor for the image \( I \) of a local neighbourhood \( \Omega \), we then have

\[
T = \int_{\Omega} w(x) \begin{bmatrix} I_u(x)^2 & I_u(x)I_v(x) \\ I_v(x)I_u(x) & I_v(x)^2 \end{bmatrix} dx,
\]

where \( x = [u,v]^T \) are image coordinates, \( w(x) \) is some low-pass filter and \( I_u \) and \( I_v \) are the partial derivatives of \( I \) in the u- and v-direction respectively. The Harris measure \( C_H \) is then given by

\[
C_H = \det(T) - \kappa(\text{trace}(T)^2),
\]

where \( \kappa \) is a sensitivity parameter. The higher the Harris measure, the more likely is the point \( x \) a corner point. The amount of found interest points therefore depends on a set threshold of the Harris measure.

When interest points are found, the next task is to establish which points that are most likely to be in correspondence. The common approach for this is to generate a descriptor for each point’s local environment, and find two descriptors with the smallest difference between them. A descriptor encodes useful information about a region, like color and shape, so that the similarity between two descriptors can be quantified. A good descriptor is invariant to changes in scale, orientation and contrast, which means that similarities can be found even during non-optimal conditions (noisy images for example). An example of such a descriptor is the scale-invariant feature transform (SIFT) \[10\], which is popular due to its robustness in this aspect.

### 2.2.3 Epipolar geometry

The problem of establishing whether two 2D points are in correspondence or not has still not been properly reviewed. This correspondence problem is determined by the epipolar geometry between the two cameras, i.e. their geometric relation. More precisely, in order for two image points \( y_1 \) and \( y_2 \) to be in correspondence, the epipolar constraint must be satisfied. The epipolar constraint is defined by the 3x3 fundamental matrix \( F \).

Again, let \( x \) be a 3D point and \( y_1 \) and \( y_2 \) the corresponding image points in cameras \( C_1 \) and \( C_2 \) (see figure 2.4). The camera centers \( n_1 \) and \( n_2 \), in figure 2.4
depicted behind the virtual image planes, form a baseline which intersects the image planes in the *epipoles* $e_1$ and $e_2$. These points can be seen as projections of the other camera center, and are given by

$$e_1 \sim C_1 n_2,$$

$$e_2 \sim C_2 n_1.$$  \hspace{1cm} (2.7)

The epipoles and the image projections together form *epipolar lines*. More precisely, for a given projection $y_1$ of $x$, the corresponding point $y_2$ will lie somewhere on the epipolar line in $C_2$. This is the geometric interpretation of the epipolar constraint. If the point $y_2$ does not lie on the epipolar line, the epipolar constraint is not fulfilled, and $y_2$ is not a corresponding point. Algebraically, this can be tested with the fundamental matrix. If $y_1$ and $y_2$ are the image coordinates in image 1 and 2, they are in correspondence if

$$y_2^T F y_1 = 0,$$  \hspace{1cm} (2.9)

where $F$ is the fundamental matrix.

![Figure 2.4: For a given 3D point $x$, its epipolar line in each image is the line between the projection coordinates and the epipole.](image)

### 2.2.4 Stereo rectification

One major advantage with MVS systems is the ability to significantly reduce the search space in the correspondence problem through *stereo rectification*. In other cases, when the relative transformation between the two cameras is unknown, the search space for corresponding points is two-dimensional (see figure 2.5a). By rectifying the images (see figure 2.5b), the epipolar lines become parallel and horizontal, meaning the search for a corresponding point only has to be carried out in the one-dimensional space of the epipolar line.
2.2.5 Depth from disparity

From rectified stereo images, it is possible to calculate a disparity map. Disparity means distance and, in the context of MVS, it refers to the distance between corresponding image points in rectified stereo images. This distance is inversely proportional to the depth of the 3D point, i.e. a larger disparity corresponds to a closer 3D point. With the help of figure 2.5a and 2.5b, we can derive the mathematical definition of disparity.

Let $x$ be a 3D point and $y_1 = [u_1, v_1]^T$, $y_2 = [u_2, v_2]^T$ its projection coordinates in the rectified cameras $C_1$ and $C_2$ (see figure 2.5a). Furthermore, let $b$ be the baseline distance between the camera centers $n_1$ and $n_2$, $f$ the cameras’ focal length and $Z$ the depth from the baseline to $x$. Through similarity (see figure 2.5b), we can derive the following equation
\[ \frac{b}{Z} = \frac{(b + u_2) - u_1}{Z - f}. \]  
(2.10)

Since disparity refers to the distance between the image points, we can rearrange equation 2.10 and define the disparity \( d \) as
\[ d = u_1 - u_2 = \frac{fb}{Z}. \]  
(2.11)

As stated by equation 2.11, the disparity is inversely proportional to the depth. Therefore, by knowing the focal length and baseline length, the depth can be estimated by calculating the disparity. More specifically, the depth is given by
\[ Z = \frac{fb}{d}. \]  
(2.12)

As previously mentioned, the focal length is determined in the camera calibration process. Since the two cameras’ extrinsic parameters (i.e., translation and rotation) are also determined in the calibration, and the relative transformation between the cameras is fixed, the baseline length can also be determined.

**The disparity map**

Calculating a disparity map means calculating the disparity for each pixel in the stereo images. The result is an image, usually grey-scale, where the disparity has been mapped to an intensity. Figure 2.6 shows an example of this, with rectified stereo images of a bowling ball and pins and the corresponding disparity map calculated for both cameras. In the disparity maps, the wall in the background has an intensity close to zero, while objects in front of it have an ascending intensity the closer they are to the camera.
Figure 2.6: Left and right stereo images and their corresponding disparity maps. Image derived from The Middlebury Computer Vision Pages [12], and used with permission.

Since the disparity is only a relative distance measure between the projections in each image, the disparity map can either be represented by the left or the right view. The left disparity map in figure 2.6, for example, is represented by the left view, which is also the general representation chosen in this thesis.

The disparity map gives us all the available 3D information about a scene from a single view instance. This is the key property of the disparity map; to store observed 3D geometry in a 2D representation. If we wish to step back into the 3D world, triangulating 3D points from a disparity map is a trivial task. Consider, again, the setting shown in figure 2.5a. The 3D coordinates \([x, y, z]^{\top}\) of the 3D point \(x\) are then given by

\[
[x, y, z]^{\top} = \left[ \frac{u_1 Z}{f}, \frac{v_1 Z}{f}, Z \right],
\]

where \(Z\) is the depth given by equation 2.12. If we insert equation 2.12 into 2.13, we get

\[
[x, y, z]^{\top} = \left[ \frac{u_1 b}{d}, \frac{v_1 b}{d}, \frac{f b}{d} \right].
\]

Note that these 3D coordinates are relative to the left camera as we have defined the disparity map in terms of the left camera.
2.3 Implicit volume representation

The theory in this section is explained with the help of [13].

The way a 3D model is represented during 3D reconstruction can vary considerably. Usually the choice of representation is not arbitrary, but stems from the chosen reconstruction method. For example, if the intention is to produce a point cloud, it makes sense to *explicitly* store the coordinates of the 3D points that have been reconstructed. As described earlier, the 3D model can also be described *implicitly*, where the surface is approximated by some mathematical function. In this case, the implicit function is often represented discretely in a three-dimensional grid which encloses the 3D model. An example of such an implementation is an SDF, where each grid vertex then stores a signed distance to the surface of the 3D model. This type of volumetric representation is suitably combined with visibility based reconstruction methods, as it is particularly susceptible to incremental update. The volume is then allowed to be updated in a manner where observation certainty can be accounted for.

This section provides a closer look at the implicit representation of the 3D model chosen in this thesis, and more specifically into the SDF.

2.3.1 Volumetric data

Volumetric data is the result of a discrete sampling of a three-dimensional function, also sometimes referred to as a scalar field. Volumetric data is the common output of any imaging technique performed in three dimensions. In medical imaging, for example, such techniques include Magnetic Resonance Imaging (MRI) and Computed Tomography (CT). The data retrieved with these techniques represents spatial samples of the signal intensity function generated by the imaging device. However, the z-dimension (along the patient) usually suffers from a lower sampling rate [1]. The best way to represent this kind of volumetric data is then arguably as distinct 2D images, which can be readily accessed for analysis.

In the case of a uniform sampling, the volumetric data can be represented in a regular 3D grid. The scalar data is then stored in equally sized voxels, which is the smallest comprising element of the grid (see figure 2.7a). As seen in figure 2.7a, the voxel $v_{i,j,k}$ represents a three-dimensional space that has been mapped to a scalar. Therefore, a common way to represent scalar data in a 3D grid is to view each voxel as a point (see figure 2.7b). The spacing between two points is then the sampling step, which determines the resolution of the discretized function.
Figure 2.7: A 3D grid of voxels can be viewed as both representations of space and scalars.

In practice, a common way of storing values in a 3D grid is to decompose it into a one-dimensional array. Voxels are then accessed from the array with the help of a memory offset. The memory offset \( m_o \) is given by

\[
m_o = x + y \cdot N_x + z \cdot N_x \cdot N_y,
\]

where \( x, y \) and \( z \) are the 3D grid coordinates and \( N_x \) and \( N_y \) are the number of voxels in the \( x \)- and \( y \)-dimension.
2.3.2 Signed Distance Function

The theory in this section is explained with the help of [6].

A signed distance function describes, for every point in a metric space, the closest distance from that point to the boundary of a given subset of the same metric space. The distance is signed as points inside the subset have positive values, and points outside the subset have negative values. Analogously, points lying on the boundary are zero valued. Let $V \subseteq W$, where $W \in \mathbb{R}^n$. The signed distance function $\Psi(w)$ for all $w \in W$ is then defined as

$$
\Psi(w) = \begin{cases} 
  d(w, \partial V) & \text{if } w \in V, \\
  -d(w, \partial V) & \text{if } w \in V^c,
\end{cases}
$$

where $d$ is the Euclidean distance, $V^c$ is the complement of $V$ and $\partial V$ denotes the boundary of $V$.

As an example, consider the two-dimensional disc $D \subseteq X$ with the radius $r$, and $X \in \mathbb{R}^2$. The signed distance function $\Psi(x)$ of the boundary $\partial D$ for all $x \in X$ is then

$$
\Psi(x) = r - \|x\|,
$$

where $x = [x y]$ and $\|x\| = \sqrt{x^2 + y^2}$. As stated by equation 2.17, every point inside the boundary will have a positive signed distance to the boundary, and every point outside the boundary will have a negative signed distance to the boundary. All points lying on the boundary, i.e. $\|x\| = r$, will have a signed distance equal to zero.

To illustrate the behaviour of $\Psi(x)$, we can plot its evaluations for $r = 2$ in a color representation. This plot is shown in figure 2.8, where magenta represents positive values and turquoise represents negative values. Increased brightness of the color corresponds to an increased signed distance value. Note that black represents the zero-valued boundary of $D$. 
2.4 Isosurface extraction

From an SDF we can extract a level set, which is a set whose evaluations in the SDF are equal to some constant $C$. For an SDF $\Psi(x)$, $x \in \{x_1, ..., x_n\}$, the level set $L_C(\Psi)$ for the constant $C$ is then formally defined as

$$L_C(\Psi) = \{ x \mid \Psi(x) = C \}.$$  \hspace{1cm} (2.18)

In two dimensions, the level set is often referred to as the isoline, while the three-dimensional equivalent is the isosurface. The constant which defines the level set is then referred to as the isovalue.

For a continuous function, extracting the level set is straightforward. As an example, we return to the SDF $\Psi(x)$ defined in equation 2.17 for $r = 2$. The level set, or isoline, in this example is defined by the value of $\Psi$, i.e. the isovalue. Figure 2.9 shows a plot of the two isolines $L_{-1}(\Psi)$ and $L_1(\Psi)$ with an isovalue of -1 and 1. To help the visualization of the isolines, the SDF is here plotted in a gray color space, where white corresponds to a large signed distance and black is the zero distance.
In the discrete case, we usually have to extract the level set by some form of interpolation. This is unsurprising as we are looking to approximate a continuous function from a finite set of samples. Consider the two-dimensional scalar field shown in figure 2.10, where each scalar is represented by a vertex in a grid. For future reference, we here also define a cell as a set of adjacent grid vertices. A cell consists of four vertices in the two-dimensional case, and eight in the three-dimensional case. To generate an isoline with a specific isovalue, we can first linearly interpolate along each edge to get point representations of the isoline. These points are then the best representations of the isoline. The next step is to connect the points in an appropriate manner. For this, a few different approaches exist.

One common approach is the Marching squares algorithm, where each cell is processed independently along with a look-up table to decide the geometry for the isoline. In three dimensions, the equivalent of Marching squares is the Marching cubes algorithm. The next section covers the principles behind this method and, more specifically, how an isosurface can be extracted from a three-dimensional scalar field.
2.4.1 Marching Cubes

To help illustrate the principles behind the Marching cubes algorithm, we hold on to the two-dimensional equivalent Marching squares for now. For the sake of simplicity, we also continue with the working example of the two-dimensional scalar field in figure 2.10. Unlike the procedure mentioned in section 2.4, the linear interpolation operations is here saved for last.

The algorithm rests on the general assumption that the sought after isoline can only pass through each cell in a finite number of ways. Therefore, each possible topological case is stored in a look-up table for reference. The topological case for a given cell is decided by the number of vertices and the number of ways these vertices can form an inside/outside relation with the isoline. A vertex is considered to be outside the isoline if its scalar value is smaller than the isovalue, and inside if its scalar value is larger than the isovalue. For a two-dimensional grid, with four vertices in each cell, the number of possible topological cases is then $2^4 = 16$ for each cell.

As the name implies, each cell is processed independently by incrementally traversing the grid. To illustrate the algorithmic procedure for one cell, consider the cell marked in red in figure 2.11a. The first step is to determine whether each vertex is inside or outside the isoline, by comparing its scalar to the isovalue (see figure 2.11b). When all vertices have an assigned state, we combine these to form an index matching one of the topological cases stored in the look-up table. For example, if we represent inside vertices as ones and outside vertices as zeros, and we form the index from the upper left vertex in a clockwise manner, the index for the cell in figure 2.11b would be 0111. We then use this index to find the corresponding topological state in the look-up table.
2.4 Isosurface extraction

(a) The cell marked in red consists of the scalar values 3, 6, 7 and 8.

(b) For an isovalue = 5, the upper left vertex is considered outside the isoline and marked as black. The other vertices are considered to be inside the isoline and marked as white.

(c) When the topological state of the cell is determined, the intersection points of the isoline are found through linear interpolation along the edges of inside/outside vertices.

Figure 2.11: The algorithmic procedure of Marching squares for one cell.

When the topological state of the cell has been determined, we use linear interpolation to find the intersection points of the isoline. For an isovalue = 5, the corresponding isoline when the marked cell has been processed is shown in figure 2.11c. If we process all the cells in the grid, the resulting isoline is shown in figure 2.12.

Figure 2.12: The resulting isoline for an isovalue = 5 when all the cells have been processed by the Marching squares algorithm.
The three-dimensional Marching cubes follows the same procedure, but presents a significantly higher number of possible topological cases. In fact, as each cell now consists of eight vertices, the number of possible topological cases is $2^8 = 256$. Fortunately, this number can be reduced to 15 by using symmetry arguments. On the other hand, several ambiguous cases emerge as there are circumstances for a cell where the isosurface can be constructed in more than one way. A simple solution to this problem is to add a few complementary cases, which are cases where the signs of the vertex scalars have been flipped. The main purpose of these cases is to prevent the creation of holes in the isosurface.
In this chapter, the proposed method is presented. Section 3.1 first describes the datasets used and the experimental setup, while section 3.2 covers details of the proposed method. Lastly, the evaluation methodology is covered in section 3.3.

### 3.1 Datasets

This section describes the experimental setup for the 3D reconstruction as well as details of how the two types of datasets used in this thesis were generated. The two dataset types are referred to as test rig data and real data, and are denoted $H_t$ and $H_r$ throughout this thesis. Each dataset consists of a measurement set $M^t$ such that $H = \{M^t\}$, for each time instance $t \in \{1, \ldots, T\}$. Each measurement set consists of a disparity map $d^s(u, v)$ and camera matrix $C^s$ such that $M = \{d^s(u, v), C^s\}$, for each stereo pair $s \in \{1, 2, 3\}$.

Both dataset types have principally the same experimental setup, but the test rig data setup is a down-scaled version of the real data setup. Furthermore, as opposed to the real data setup, the test rig data setup is located indoors.

#### 3.1.1 Experimental setup

As mentioned above, the MVS reconstruction system consists of three stereo pairs. The stereo pairs are mounted to a fixed rig to the left, right and above the reconstruction object. Figure 3.1 shows an illustration of this arrangement. For the sake of simplicity, each stereo pair is illustrated with a single camera for the rest of this thesis.
Figure 3.1: The general experimental setup for the 3D reconstruction. Three stereo pairs are mounted to a fixed rig to the left, right and above the object to be reconstructed. Each stereo pair is illustrated with a single camera.

During the reconstruction procedure, the object to be reconstructed is moved through the rig. Although the stereo pairs are physically static, the object movement is registered in terms of camera matrix transformations. Each camera pose is then relative to the initial position of the left stereo pair at the first time instance of the imaging sequence.

3.1.2 Real data

Saab Dynamics AB has a profuse collection of recorded data from the operating MVS system. The system is deployed at a Swedish timber terminal, where it performs and records 3D reconstructions of each timber truck that passes through the rig. The system has been in use for over half a year, meaning data from different weather and lighting conditions exists.

Since the data is currently used for manual volume estimation, each dataset is accompanied by a volume estimate performed by a human operator. As mentioned in section 1.1, the volume estimation is performed by manually measuring the width and height of each timber load in a rectified side image. Naturally, as estimating a volume from a single image is highly problematic, these volume estimates could not be used as ground truths in the evaluation. Instead, the real datasets were used for a qualitative evaluation of the surface reconstruction quality achieved by the proposed method. However, the proposed method’s volume
estimate was still compared to the manual volume estimate to investigate a possible correlation between the two.

In total, 5 real datasets were used. These datasets were chosen to contain as much variation in weather and lighting conditions as possible. Each real dataset $H^k_r$, $k \in \{1, \ldots, 5\}$, consists of measurement sets from one full 3D reconstruction of a timber truck with three separate loads. A manual volume estimate is provided for each of the three loads.

Figure 3.2 shows an example of one full 3D reconstruction of a real dataset in a point cloud representation.

![Figure 3.2: Point cloud representation of a truck from one full 3D reconstruction. The image is a screenshot of Saab Dynamics AB’s 3D reconstruction software.](image)

### 3.1.3 Test rig data

In addition to the real datasets, Saab Dynamics AB has built a test rig that is a down-scaled version of the operating MVS system. The test rig is located indoors with constant lighting conditions and new data can be acquired at any time. Figure 3.3 shows an image of the test rig setup.
Figure 3.3: The test rig data setup. In this image, stereo pair 1 is to the right and stereo pair 2 is to the left.

In total, 30 test rig datasets were used. These datasets were divided into two groups of 15, where each group corresponds to reconstructions of a specific geometric object. The geometric objects used were a rectangular box and a pyramid, with the corresponding datasets denoted $H_{t,\text{box}}$ and $H_{t,\text{pyr}}$. The box and the pyramid had different bark textures applied to each side to increase the number of features for the 3D reconstruction. Figure 3.4 shows images of the two geometric objects used for the test rig data.

(a) Rectangular box.  (b) Pyramid.

Figure 3.4: The two geometric objects used for the test rig datasets.

The reason why the test rig data was primarily used for evaluation is that a ground truth could be provided for each geometric object. Furthermore, being a controlled environment, the reconstruction procedure could be performed so
that the reconstruction quality is maximized. This is useful as we want to be able to evaluate the proposed method independently of the reconstruction performance.

3.2 Proposed method

The proposed method is a volumetric integration of depth maps, where each depth map is incrementally fused into an SDF representing the reconstructed object. The pipeline consists of mainly two parts: volumetric integration and volume estimation. An illustration of this pipeline is shown in figure 3.5.

![Figure 3.5: The main pipeline of the proposed method.](image)

The first part of the pipeline takes a dataset $H$ and integrates into an SDF $\Psi$. The second part takes the SDF $\Psi$ and produces a polygonized model for visualization, as well as a volume estimate $V$. The following sections describe each part in more detail.

3.2.1 Volumetric integration

The volumetric integration is the main part of the pipeline and where the SDF is constructed. This part closely resembles the method proposed by Bylow et al [3], with the main difference that we here use disparity maps acquired through MVS instead of raw depth images from a range camera. Furthermore, like most comparable methods, Bylow et al use depth data from a single camera during reconstruction, while we here use disparity maps from up to three stereo pairs at each time instance. The key contribution with the proposed method is therefore perhaps not in the reconstruction technique, but rather lessons on how to handle the geometric ambiguities that arise from using multiple stereo pairs simultaneously.

The volumetric integration process can itself be subdivided into five parts: perspective voxel projection, triangulation, normal vector estimation, calculation of point-to-plane distance and SDF update. These parts occur in the named order, as can be seen in figure 3.6. Each part will be described in more detail later on in their respective section. For now, we focus on the general attributes of the algorithm.
The input is a measurement set $M^t$ containing a disparity map $d^{t,s}(u,v)$ and corresponding camera matrix $C^{t,s}$ for each time instance $t \in \{1, \ldots, T\}$ and stereo pair $s \in \{1, 2, 3\}$. Each disparity map has been filtered in a prior step such that it only contains values representing the object $\chi$, and is zero otherwise. Output is a discrete SDF $\Psi[x]$ represented by a 3D grid of resolution $N = N_x \times N_y \times N_z$, where $x = [x,y,z]^T \in \mathbb{N}^3$ are the grid coordinates and $N_x, N_y, N_z$ are the number of voxels in the x-, y- and z-dimensions respectively.

As a first step, the 3D grid is allocated and initialized so that it encloses the object $\chi$ that is to be reconstructed (see figure 3.7). The spatial resolution $\gamma$ is also set, which corresponds to the side length of each voxel. Since the 3D grid represents the SDF, the task is to calculate, for each voxel, a signed distance to the closest 3D point of $\chi$. In addition, we also introduce a weight function $\Gamma[x]$ that describes the certainty associated with each signed distance. Each voxel therefore carries two values; the signed distance $\Psi$ and the associated weight $\Gamma$. The signed distance and the weight are then continuously updated for each new measurement set $M^t$ that is integrated.

In order to minimize the influence of poor signed distance estimates, we use here a truncated SDF, inspired by Curless and Levoy [6]. The idea behind the truncated SDF is that large distance estimates are more likely to be erroneous, so we truncate the SDF at some maximum distance $D_{\text{max}}$. Every estimated distance that is larger than $D_{\text{max}}$ is then set to $D_{\text{max}}$. Similarly, we set negative distances that are smaller than some minimum distance $D_{\text{min}}$ to $D_{\text{min}}$. In this thesis, we have $D_{\text{min}} = -D_{\text{max}}$, which is why $D_{\text{min}}$ is seldom referred to. The result of the truncation is an SDF where only voxels close to the surface are trusted to represent it. From here on, any reasoning about the SDF refers to this truncated version.

The weight function $\Gamma[x]$ can be represented in many ways, tentatively as a dependant on viewing angle. In this thesis, a linear distance weight has been used. The linear weight function $w_l(d), d \in \mathbb{R}$, is equivalent to the one proposed by Curless and Levoy [6], which assigns a lower certainty to voxels behind the surface of $\chi$. More specifically, at some distance $\eta$ behind the surface, the weight linearly decreases.
3.2 Proposed method

Figure 3.7: A 3D grid (here visualized in 2D) is initialized so that the reconstruction object $\chi$ is enclosed. The spatial resolution $\gamma$ corresponds to the side length of each voxel.

decreases to zero at the truncation distance $D_{\text{max}}$. The linear weight is given by

$$w_1(D) = \begin{cases} 1 & \text{if } D < \eta, \\ \frac{D_{\text{max}} - D}{D_{\text{max}} - \eta} & \text{if } \eta \leq D \leq D_{\text{max}}, \\ 0 & \text{if } D > D_{\text{max}}, \end{cases}$$

where $D$ is the signed distance.

Before we get into the details of each part of the volumetric integration, we make some final remarks on the initialization of both $\Psi[x]$ and $\Gamma[x]$. As we have seen from the definition of the SDF in section 2.3.2, voxels residing inside $\chi$ will have a positive signed distance, while voxels outside have a negative signed distance. This means that each voxel, including those carrying a truncated signed distance, represents a state of space. More specifically, voxels that carry a positive signed distance represent an unseen space, while negative signed distance voxels represent empty space. This notation was proposed by Curless and Levoy and plays a significant role in the volume estimation part of this thesis, as will be described in section 3.2.2. For the initialization part, however, we here do the opposite of Curless and Levoy, and set each voxel distance to $-D_{\text{max}}$ so that it represents empty space. The weight $\Gamma$ is set to 0, which will make sense when we review the update procedure.
Perspective voxel projection

For each new disparity map that we want to integrate into the SDF, we need to update the voxels to the newly available 3D information accordingly. A common method for this is ray casting, where a ray is formed from the camera center through each pixel of the disparity map, updating each voxel along the line of sight. In this thesis, we instead use voxel projection, where each voxel is projected onto the image plane of the disparity map. Motivated by Bylow et al [3], the key reasons for this choice of method are to ensure that each voxel is only visited once, and to keep computations parallelizable. In other words, it is a choice made with the demand for scalability kept in mind.

Consequently, each calculation described from here on is performed on a per voxel basis. For simplicity, we also refer to a single disparity map in each calculation, although several disparity maps may be present for each voxel. Handling several disparity maps is primarily an update problem, so this will be covered in the section regarding update of the SDF.

As we saw in section 2.2, the mapping of a 3D point to an image plane is given by equation 2.4. For a given disparity map \( d(u, v) \) and the corresponding camera matrix \( C \), we project the voxel center in global coordinates \( x_g = [x_g, y_g, z_g]^\top \) onto the image plane of the disparity map using equation 2.4 to obtain the projection coordinates \( y_p = [u_p, v_p]^\top \). An illustrative example of this projection is shown in figure 3.8.
3.2 Proposed method

Figure 3.8: A voxel with global coordinates $x_g$ is projected onto the image plane of the disparity map $d(u,v)$ to get the projection coordinates $y_p$.

For future reference, we consider a vector-valued function $\rho^s(x)$ that describes the projection of a 3D point $x = [x, y, z]^T$ to image plane coordinates $y_p^s$ for stereo pair $s$ such that

$$y_p^s = \rho^s(x) = C^s x,$$  \hspace{1cm} (3.2)

where $y_p^s$ are the projection coordinates.

**Triangulation**

The projection coordinates $y_p$ tell us which 3D point of $\chi$ that is closest to the voxel coordinate $x_g$. In other words, for this particular voxel, this 3D point is the best representation of the surface that we wish to calculate the signed distance to. Therefore, we triangulate the corresponding 3D point $x_i = [x_i, y_i, z_i]^T$ from the disparity map using equation 2.14 (see figure 3.9).
**Figure 3.9:** The 3D point $\mathbf{x}_t$ that is closest to $\mathbf{x}_g$ is given by triangulation from the projection coordinates $\mathbf{y}_p$.

Again, for future reference, we define a vector-valued function $\tau^s(u,v,d)$ that triangulates a 3D point $\mathbf{x} = [x, y, z]^T$ from the image plane coordinates $\mathbf{y}^s = [u, v]^T$ and the corresponding disparity $d = d^s(u,v)$ for stereo pair $s$ such that

$$\mathbf{x} = \tau^s(u,v,d) = \begin{bmatrix} \frac{ub}{d} & \frac{vb}{d} & \frac{fb}{d} \end{bmatrix},$$ (3.3)

where $f$ and $b$ are the focal length and baseline distance retrieved from $\mathbf{C}^s$.

**Normal vector estimation**

Next, we estimate the normal vector $\hat{\mathbf{n}}_t = [\hat{x}, \hat{y}, \hat{z}]^T$ for the triangulated 3D point $\mathbf{x}_t$ from the disparity map. This estimation is built on the assumption that the local region around $\mathbf{x}_t$ is planar. Given the projection coordinates $\mathbf{y}_p$, we then triangulate 3D points from two adjacent pixels in the disparity map and form the two vectors $\bar{\mathbf{u}}, \bar{\mathbf{v}}$ spanning the plane.

As an example, consider the case when we choose the adjacent pixels above and left of $\mathbf{y}_p$, and triangulate the corresponding 3D points $\mathbf{x}_a$ and $\mathbf{x}_l$. Let $\bar{\mathbf{u}} = \mathbf{x}_a - \mathbf{x}_t$ and $\bar{\mathbf{v}} = \mathbf{x}_l - \mathbf{x}_t$, we then have

$$\bar{\mathbf{u}} = \tau(u_p, v_p - 1, d(u_p, v_p - 1)) - \tau(u_p, v_p, d(u_p, v_p)),$$ (3.4)

$$\bar{\mathbf{v}} = \tau(u_p - 1, v_p, d(u_p - 1, v_p)) - \tau(u_p, v_p, d(u_p, v_p)).$$ (3.5)
3.2 Proposed method

The normal $\hat{n}_t$ is then given by the cross product between $\bar{u}$ and $\bar{v}$, i.e.

$$\hat{n}_t = \frac{\bar{u} \times \bar{v}}{\|\bar{u} \times \bar{v}\|},$$  \hspace{1cm} (3.6)

where $\times$ denotes the cross product.

Since the disparity map has been filtered from noise and background, there may be cases when the adjacent pixels in the disparity map are undefined. In these cases, we simply search until we have found two pixels that we can form the vectors $\bar{u}$ and $\bar{v}$ from. Figure 3.10 illustrates this procedure of finding eligible pixels in the disparity map. Note that in order to estimate the correct normal, i.e. pointing towards the camera, the order in which we name $\bar{u}$ and $\bar{v}$ is important.

**Figure 3.10:** The procedure for finding eligible pixels in the disparity map to estimate the normal $n_t$ from.

Lastly, a short algorithmic overview of the normal vector estimation is provided in algorithm 1.
Algorithm 1: Estimation of normal vector $\hat{n}_t$ for 3D point $x_t$.

Input: Projection coordinates $y_p$  
Triangulated point $x_t$  
Disparity map $d(u, v)$

Output: Normal $\hat{n}_t$

1. if $d(u_p, v_p - 1) \neq 0$ then
   2. if $d(u_p - 1, v_p) \neq 0$ then
      3. $\hat{u} = \tau(u_p, v_p - 1, d(u_p, v_p - 1)) - x_t$
      4. $\hat{v} = \tau(u_p - 1, v_p, d(u_p - 1, v_p)) - x_t$
      5. return $\hat{n}_t = \frac{\hat{u} \times \hat{v}}{||\hat{u} \times \hat{v}||}$
   end
   7. if $d(u_p + 1, v_p) \neq 0$ then
      8. $\hat{u} = \tau(u_p + 1, v_p, d(u_p + 1, v_p)) - x_t$
      9. $\hat{v} = \tau(u_p, v_p - 1, d(u_p, v_p - 1)) - x_t$
     10. return $\hat{n}_t = \frac{\hat{u} \times \hat{v}}{||\hat{u} \times \hat{v}||}$
   end
12. else if $d(u_p, v_p, v_p + 1) \neq 0$ then
   13. if $d(u_p - 1, v_p) \neq 0$ then
      14. $\hat{u} = \tau(u_p - 1, v_p, d(u_p - 1, v_p)) - x_t$
      15. $\hat{v} = \tau(u_p, v_p + 1, d(u_p, v_p + 1)) - x_t$
     16. return $\hat{n}_t = \frac{\hat{u} \times \hat{v}}{||\hat{u} \times \hat{v}||}$
   end
   18. if $d(u_p + 1, v_p) \neq 0$ then
      19. $\hat{u} = \tau(u_p, v_p + 1, d(u_p, v_p + 1)) - x_t$
      20. $\hat{v} = \tau(u_p + 1, v_p, d(u_p + 1, v_p)) - x_t$
     21. return $\hat{n}_t = \frac{\hat{u} \times \hat{v}}{||\hat{u} \times \hat{v}||}$
   end
22. else
23. return $\hat{n}_t = 0$

Point-to-plane distance

Once we have the normal $\hat{n}_t$, we approximate the signed distance through the point-to-plane distance (see figure 3.11). Let $\bar{w}$ be the vector from the voxel center to the triangulated 3D point, i.e. $\bar{w} = x_t - x_g$. The point-to-plane distance $D_{ptp}$ is then given by

$$D_{ptp} = \bar{w} \cdot \hat{n}_t,$$  \hspace{1cm} (3.7)

where $\cdot$ denotes the scalar product. This metric is used as calculating the true signed distance is a time consuming task. Furthermore, as noted by Bylow et al
3.2 Proposed method

[3], the point-to-plane distance is generally more accurate than the point-to-point distance.

![Diagram](image)

**Figure 3.11**: By forming a vector \( \mathbf{w} = \mathbf{x}_t - \mathbf{x}_g \), the signed distance is approximated by the point-to-plane distance \( D_{ptp} \) given by the scalar product between \( \mathbf{w} \) and the normal \( \mathbf{n}_t \).

### SDF update

When we have calculated the signed distance, we want to update the stored distance \( \Psi \) and weight \( \Gamma \) in a proper manner. This is done using a running weighted average. For a given time instance \( t \), the update is given by

\[
\Psi^t = \frac{\Psi^{t-1}\Gamma^{t-1} + D^t W^t}{\Gamma^{t-1} + W^t},
\]

(3.8)

\[
\Gamma^t = \Gamma^{t-1} + W^t,
\]

(3.9)

where \( D^t \) is the calculated signed distance and \( W^t \) is the linear weight given by equation 3.1.

So far we have only considered a single disparity map in each calculation, which is true when we only use one stereo pair. When all stereo pairs are used \( (s = 3) \), however, some ambiguity arises as to which signed distance we should choose to update the SDF with. Therefore, two different approaches to solving this problem, with varying level of complexity, have been studied. The reason for this is
to investigate whether the more complex update model has a corresponding increase in reconstruction quality, and thereby also volume estimation accuracy.

Common for both update models, however, is the use of a visibility threshold $\lambda$. The visibility threshold needs to be exceeded in order for a voxel to be updated to the empty or unseen state. In more concrete terms, empty space voxels will have had more than $\lambda$ negative signed distances observed, while unseen space voxels will have had more than $\lambda$ positive signed distances observed. The motivation for this threshold is to increase the level of certainty for each update as falsely updated voxels will affect the volume estimation.

The first approach, referred to as update model 1, is to simply choose the shortest signed distance. In the more sophisticated second approach, update model 2, we associate each stereo observation with a weight $\omega$. This weight is in turn the average of three separate weights associated with the observed signed distance $D$, the distance $D_{cam}$ between the voxel and the stereo pair and the observation angle $\theta$ between the normal $\hat{n}_i$ and the principal axis of the left camera in the stereo pair. The idea behind these weights is to help determine the most optimal observation of the voxel. In other words, the observation with the largest combined weight is the theoretically most suitable observation in terms of visibility of the voxel and its distance to the object. Figure 3.12 shows an example where a voxel and its corresponding measures for stereo pair 1 have been indicated.

![Figure 3.12: An example of a stereo pair 1 observation where the signed distance $D$, stereo distance $D_{cam}$ and observation angle $\theta$ have been indicated for a given voxel.](image)
For a stereo pair \( s \), the stereo distance \( D^s_{\text{cam}} \) is given by

\[
D^s_{\text{cam}} = \| \mathbf{x}^s_{\text{cam}} - \mathbf{x}_g \|,
\]

(3.10)

where \( \mathbf{x}_{\text{cam}} \) are the global coordinates of the left camera of the stereo pair. The corresponding weight \( \omega^s_{D_{\text{cam}}} \) is then given by

\[
\omega^s_{D_{\text{cam}}} = 1 - \frac{D^s_{\text{cam}}}{\sum_{i=1}^{N} D^i_{\text{cam}}},
\]

(3.11)

where \( N \) is the total number of used stereo pairs. As equation 3.11 states, the closer the stereo pair is to the voxel, the larger is the stereo distance weight. Similarly, the observed signed distance weight \( \omega^s_D \) is given by

\[
\omega^s_D = 1 - \frac{|D^s|}{\sum_{i=1}^{N} |D^i|}.
\]

(3.12)

As equation 3.12 states, the closer the voxel is to the object \( \chi \), the larger is the signed distance weight. Lastly, the observation angle weight \( \omega^s_\theta \) is given by

\[
\omega^s_\theta = \cos(\theta^s) = \frac{(\mathbf{x}^s_{\text{cam}} - \mathbf{x}^s_i) \cdot \mathbf{n}^s_i}{\| \mathbf{x}^s_{\text{cam}} - \mathbf{x}^s_i \|}.
\]

(3.13)

As equation 3.13 states, an observation perpendicular to the object surface yields the highest observation angle weight. The combined weight \( \omega^s \) for the stereo observation \( s \) is then the average of these three weights, i.e.

\[
\omega^s = \frac{\omega^s_{D_{\text{cam}}} + \omega^s_D + \omega^s_\theta}{3}.
\]

(3.14)

When the weight for each stereo observation has been calculated, we simply choose the observation with the highest weight.

To put it all together, the main algorithm for the volumetric integration part is presented in algorithm 2. Note that, in this algorithm, approach 1 is used for the SDF update.
Algorithm 2: Main algorithm for volumetric integration of measurement set $M^t$.

**Input**: Measurement set $M^t = \{d^{t,s}(u,v), C^{t,s}\}$, $s \in \{1, 2, 3\}$
- Number of voxels $N = N_x \cdot N_y \cdot N_z$
- Truncation distance $D_{max}$
- Visibility threshold $\lambda$
- Number of stereo pairs $n_s$

**Output**: Updated $\Psi[x]$ and $\Gamma[x]$

for $i = 1 : N$ do
    $D = \infty$
    $W, a, b = 0$
    Transform grid coordinates $x_i$ to global coordinates $x_g$
    for $s = 1 : n_s$ do
        Voxel projection: $y^s_p = \rho^s(x_g)$
        if $d^s(y^s_p) \neq 0$ then
            Triangulation: $x_t = \tau^s(y^s_p, d(y^s_p))$
            if Neighbouring pixels to $y_p$ are defined in $d$ then
                Estimate normal $\hat{n}_t$ for $x_t$ according to algorithm 1
                Calculate point-to-plane distance $\hat{D}^s$ according to equation 3.7
                Calculate linear weight $\hat{W}^s$ according to equation 3.1
                if $\hat{D}^s < 0$ then
                    $a = a + 1$  ★ Negative signed distance count
                end
                if $\hat{D}^s > 0$ then
                    $b = b + 1$  ★ Positive signed distance count
                end
            end
        else
            $\hat{D}^s = \infty$
            $\hat{W}^s = 0$
        end
        if $|\hat{D}^s| < |D|$ then
            $D = \hat{D}^s$
            $W = \hat{W}^s$
        end
    end
    if $D = \infty$ then
        if $-D_{max} < D < D_{max}$ then
            Update SDF $\Psi^t[x_i]$ and weight $\Gamma^t[x_i]$ according to equation 3.8 and 3.9
        else if $\Gamma^{t-1}[x_i] = 0$ and ($a > \lambda$ or $b > \lambda$) then
            if $D < 0$ then
                $\Psi^t[x_i] = -D_{max}$  ★ Update SDF to empty space
            else
                $\Psi^t[x_i] = D_{max}$  ★ Update SDF to unseen space
            end
        end
    end
end
3.2 Proposed method

3.2.2 Volume estimation

The volume estimation is the second part of the pipeline, and where the SDF is converted into a visualized model as well as an estimate of the volume is produced. This part is divided into three parts: isosurface extraction and processing, visualization and volume calculation. An overview of this part is shown in figure 3.13.

![Volume estimation diagram](image)

*Figure 3.13: The main parts of the volume estimation.*

The input is an SDF $\Psi[x]$ and the output is a visualization of the polygonized model and a volume estimate $V_x$. Most of the operations performed in this section are implemented using the C++ open-source framework The Visualization Toolkit (VTK) [15]. The following sections describe each operation in more detail.

**Isosurface extraction and processing**

The SDF $\Psi[x]$ is stored using the VTK class `vtkImageData`. As a first step, the isosurface is extracted using a variant of the Marching Cubes algorithm (see section 2.4.1), namely `vtkContourFilter`. As we are interested in the surface (signed distance equal to zero), the isosurface is defined in terms of an isovalue = 0.

Next, as the SDF will contain some amount of noise affecting the isosurface extraction, we filter the isosurface from disconnected regions. This is done using the class `vtkPolyDataConnectivityFilter` along with the function `SetExtractionModeToLargestRegion()`. The result is a fully connected isosurface which corresponds to the best representation of the surface.
Visualization

The visualization is performed using the standard procedure of VTK. From the vtkPolyDataConnectivityFilter, the isosurface is mapped to graphics primitives using vtkPolyDataMapper and is represented using vtkActor.

The unseen space voxels are represented in a separate vtkImageData object. The mapping is then done using vtkSmartVolumeMapper and represented using vtkVolume. The color of the vtkVolume object is controlled using vtkVolumeProperty, along with vtkPiecewiseFunction and vtkColorTransferFunction.

The rendering of the isosurface and the unseen space voxels is then done using vtkRenderWindow and vtkRenderer.

Volume calculation

The volume of $\chi$ is calculated by simply counting the number of voxels representing the surface $N_s$ and the number of voxels representing the unseen space $N_u$ (inside $\chi$). However, approximately half of the surface voxels reside outside the isosurface, and should not be counted. Therefore, the total number of voxels $N_{tot}$ representing $\chi$ is approximated as

$$N_{tot} \approx \frac{N_s}{2} + N_u = \sum_{x \in \mathbb{N}^3} \Pi(x),$$

where $x$ is the voxel coordinate and $\Pi(x)$ is given by

$$\Pi(x) = \begin{cases} 1/2, & \text{if } -D_{\text{max}} < \Psi[x] < D_{\text{max}}, \\ 1, & \text{if } \Psi[x] = D_{\text{max}}, \\ 0, & \text{otherwise}. \end{cases}$$

Lastly, the volume $V_\chi$ is given by

$$V_\chi = N_{tot} \gamma^3,$$

where $\gamma$ is the spatial resolution.
3.3 Evaluation

In this section the methods for evaluating the algorithm are described. The properties that have been evaluated are accuracy, precision and robustness.

3.3.1 Accuracy

The accuracy of the proposed method was evaluated on test rig data with a mean relative percentage error (MRPE) measure. For a given number of datasets \( \{H_1, ..., H_N\} \), the MRPE \( \epsilon \) is given by

\[
\epsilon = \frac{100}{N} \sum_{i=1}^{N} \left| \frac{x_i^* - x_i}{x_i^*} \right|,
\]

where \( N \) is the total number of datasets, \( x_i \) is the estimated volume for the i:th dataset and \( x_i^* \) is the true volume for the i:th dataset.

3.3.2 Precision

The precision of the proposed method was evaluated on test rig data with a percentage standard deviation (PSD) measure. For a given number of datasets \( \{H_1, ..., H_N\} \) of the same object, the PSD \( \sigma_{perc} \) is given by

\[
\sigma_{perc} = \frac{100}{\mu} \sqrt{\frac{1}{N} \sum_{i=1}^{N} (x_i - \mu)^2},
\]

where \( N \) is the total number of datasets, \( x_i \) is the estimated volume for the i:th dataset and \( \mu \) is the mean measured volume for the \( N \) datasets given by

\[
\mu = \frac{1}{N} \sum_{i=1}^{N} x_i.
\]

3.3.3 Robustness

The robustness of the proposed method was evaluated qualitatively on real data. No measures were produced, merely a visual assessment of the reconstructed surface. This assessment included handling of missing data regions and the general appearance of the surface compared to the reconstructed 3D point cloud.

The robustness evaluation was accompanied by a quantitative comparison of the proposed method’s volume estimate and the manual volume estimate on real data. This comparison was performed with a measure similar to the MRPE measure, namely the mean relative percentage difference (MRPD). For a given number of datasets \( \{H_1, ..., H_N\} \), the MRPD \( \delta \) is given by
\[ \delta = \frac{100}{N} \sum_{i=1}^{N} \left| \frac{\hat{x}_i - x_i}{\hat{x}_i} \right|, \]  

(3.21)

where \( N \) is the total number of datasets, \( x_i \) is the estimated volume for the \( i \)-th dataset and \( \hat{x}_i \) is the manual volume estimate for the \( i \)-th dataset.
In this chapter, the results for the proposed method are presented. Section 4.1 presents the quantitative results achieved from test rig data, while section 4.2 presents qualitative results of the surface reconstruction achieved from real data. Section 4.2 also contains a comparison of the achieved volume estimates and the manual volume estimates performed by an operator.

All of the execution times were produced using two Intel(R) Xeon(R) CPU E5-2637 v3 with a clock rate of 3.5 GHz and 128 GB of RAM.

### 4.1 Test rig data

As mentioned in section 3.1.3, evaluation was performed on 30 test rig datasets in total. Out of the 30 datasets, 15 contained reconstructions of the rectangular box and the other 15 contained reconstructions of the pyramid. The results for the two geometric objects are presented in their respective section, for three different grid resolutions $N$. As described in section 3.2.1, two different update models were also evaluated. Therefore, each section provides results for both update models.
4.1.1 Box datasets

The results for update model 1 on the rectangular box datasets are shown in table 4.1, while the results for update model 2 on the rectangular box datasets are shown in table 4.2. Figure 4.1 and 4.2 show example images from the surface reconstruction of dataset $H_{t,box}^9$, using update model 1, $\lambda = 2$ and two different grid resolutions.

**Table 4.1:** Results for update model 1 on the rectangular box datasets, with varying grid resolution $N$.

<table>
<thead>
<tr>
<th>Grid resolution $N$</th>
<th>Spatial resolution $\gamma$ (mm)</th>
<th>True volume $x^*$ (cm$^3$)</th>
<th>Mean estimated volume $\mu$ (cm$^3$)</th>
<th>MRPE $\epsilon$ (%)</th>
<th>PSD $\sigma_{perc}$ (%)</th>
<th>Mean execution time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>80×64×32</td>
<td>2.4</td>
<td>581.175</td>
<td>592.966</td>
<td>2.098</td>
<td>1.223</td>
<td>81.623</td>
</tr>
<tr>
<td>160×128×64</td>
<td>1.2</td>
<td>581.175</td>
<td>592.419</td>
<td>1.989</td>
<td>1.190</td>
<td>607.585</td>
</tr>
<tr>
<td>320×256×128</td>
<td>0.6</td>
<td>581.175</td>
<td>592.471</td>
<td>1.981</td>
<td>1.177</td>
<td>5147.285</td>
</tr>
</tbody>
</table>

**Table 4.2:** Results for update model 2 on the rectangular box datasets, with varying grid resolution $N$.

<table>
<thead>
<tr>
<th>Grid resolution $N$</th>
<th>Spatial resolution $\gamma$ (mm)</th>
<th>True volume $x^*$ (cm$^3$)</th>
<th>Mean estimated volume $\mu$ (cm$^3$)</th>
<th>MRPE $\epsilon$ (%)</th>
<th>PSD $\sigma_{perc}$ (%)</th>
<th>Mean execution time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>80×64×32</td>
<td>2.4</td>
<td>581.175</td>
<td>578.667</td>
<td>1.125</td>
<td>1.367</td>
<td>81.623</td>
</tr>
<tr>
<td>160×128×64</td>
<td>1.2</td>
<td>581.175</td>
<td>578.592</td>
<td>1.054</td>
<td>1.245</td>
<td>607.585</td>
</tr>
<tr>
<td>320×256×128</td>
<td>0.6</td>
<td>581.175</td>
<td>578.596</td>
<td>1.046</td>
<td>1.224</td>
<td>5147.285</td>
</tr>
</tbody>
</table>
4.1 Test rig data

Figure 4.1: Example images from the surface reconstruction of dataset $H^{9}_{t,box}$, using update model 1, $\lambda = 2$, $\gamma = 2.4$ and grid resolution $N = 80 \times 64 \times 32$.

Figure 4.2: Example images from the surface reconstruction of dataset $H^{9}_{t,box}$, using update model 1, $\lambda = 2$, $\gamma = 0.6$ and grid resolution $N = 320 \times 256 \times 128$. 
4.1.2 Pyramid datasets

The results for update model 1 on the pyramid datasets are shown in table 4.3, while the results for update model 2 on the pyramid datasets are shown in table 4.4. Figure 4.3 and 4.4 show example images from the surface reconstruction of dataset $H_{t,pyr}^{12}$, using update model 1, $\lambda = 1$ and two different grid resolutions.

**Table 4.3:** Results for update model 1 on the pyramid datasets, with varying grid resolution $N$.

<table>
<thead>
<tr>
<th>Grid resolution $N$</th>
<th>Spatial resolution $\gamma$ (mm)</th>
<th>True volume $x^*$ (cm$^3$)</th>
<th>Mean estimated volume $\mu$ (cm$^3$)</th>
<th>MRPE $\epsilon$ (%)</th>
<th>PSD $\sigma_{perc}$ (%)</th>
<th>Mean execution time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>80 × 64 × 32</td>
<td>2.4</td>
<td>214.49</td>
<td>174.832</td>
<td>19.855</td>
<td>15.694</td>
<td>27.269</td>
</tr>
<tr>
<td>160 × 128 × 64</td>
<td>1.2</td>
<td>214.49</td>
<td>184.693</td>
<td>16.023</td>
<td>15.150</td>
<td>194.783</td>
</tr>
<tr>
<td>320 × 256 × 128</td>
<td>0.6</td>
<td>214.49</td>
<td>216.587</td>
<td>8.916</td>
<td>13.120</td>
<td>1581.043</td>
</tr>
</tbody>
</table>

**Table 4.4:** Results for update model 2 on the pyramid datasets, with varying grid resolution $N$.

<table>
<thead>
<tr>
<th>Grid resolution $N$</th>
<th>Spatial resolution $\gamma$ (mm)</th>
<th>True volume $x^*$ (cm$^3$)</th>
<th>Mean estimated volume $\mu$ (cm$^3$)</th>
<th>MRPE $\epsilon$ (%)</th>
<th>PSD $\sigma_{perc}$ (%)</th>
<th>Mean execution time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>80 × 64 × 32</td>
<td>2.4</td>
<td>214.49</td>
<td>174.339</td>
<td>18.719</td>
<td>13.772</td>
<td>28.385</td>
</tr>
<tr>
<td>160 × 128 × 64</td>
<td>1.2</td>
<td>214.49</td>
<td>183.360</td>
<td>14.608</td>
<td>13.144</td>
<td>205.090</td>
</tr>
<tr>
<td>320 × 256 × 128</td>
<td>0.6</td>
<td>214.49</td>
<td>215.342</td>
<td>8.380</td>
<td>12.395</td>
<td>1651.769</td>
</tr>
</tbody>
</table>
4.1 Test rig data

Figure 4.3: Example images from the surface reconstruction of dataset $H_{t,pyr}^{12}$, using update model 1, $\lambda = 1$, $\gamma = 2.4$ and grid resolution $N = 80 \times 64 \times 32$.

Figure 4.4: Example images from the surface reconstruction of dataset $H_{t,pyr}^{12}$, using update model 1, $\lambda = 1$, $\gamma = 1.2$ and grid resolution $N = 160 \times 128 \times 64$. 
4.2 Real data

As mentioned in section 3.1.2, 5 real datasets in total were used for the qualitative evaluation. However, as each of the real datasets consists of three separate timber loads with corresponding manual volume estimates, the total number of evaluated volume estimations was 15. Each dataset represented a subjectively chosen characteristic circumstance: sunny weather, cloudy weather, night, a glared camera and a speeding truck. Comparisons of the result for each dataset and the corresponding manual volume estimate for the two update models are shown in table 4.5 and 4.6. The third column in the tables describes whether the volume for each load was overestimated (+) or underestimated (-).

**Table 4.5:** Comparison of the result for each real dataset and the corresponding manual volume estimate for update model 1.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>MRPD δ (%)</th>
<th>Over/under estimation</th>
<th>Mean number of frames</th>
<th>Mean execution time (s)</th>
<th>Mean time per frame (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hₚ, sunny</td>
<td>20.2694</td>
<td>+++</td>
<td>77.5</td>
<td>1520</td>
<td>19.61</td>
</tr>
<tr>
<td>Hₚ, night</td>
<td>30.3215</td>
<td>+++</td>
<td>35.5</td>
<td>695.4</td>
<td>19.59</td>
</tr>
<tr>
<td>Hₚ, glare</td>
<td>17.0233</td>
<td>+ - -</td>
<td>25</td>
<td>449.4</td>
<td>17.98</td>
</tr>
<tr>
<td>Hₚ, fast</td>
<td>9.0362</td>
<td>- - -</td>
<td>26</td>
<td>377.5</td>
<td>14.52</td>
</tr>
<tr>
<td>Hₚ, cloudy</td>
<td>9.7059</td>
<td>+++</td>
<td>38</td>
<td>691.9</td>
<td>18.21</td>
</tr>
</tbody>
</table>

**Table 4.6:** Comparison of the result for each real dataset and the corresponding manual volume estimate for update model 2.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>MRPD δ (%)</th>
<th>Over/under estimation</th>
<th>Mean number of frames</th>
<th>Mean execution time (s)</th>
<th>Mean time per frame (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hₚ, sunny</td>
<td>17.1141</td>
<td>+++</td>
<td>77.5</td>
<td>1523</td>
<td>19.66</td>
</tr>
<tr>
<td>Hₚ, night</td>
<td>26.2231</td>
<td>+++</td>
<td>35.5</td>
<td>694.0</td>
<td>19.55</td>
</tr>
<tr>
<td>Hₚ, glare</td>
<td>17.1893</td>
<td>+ - -</td>
<td>25</td>
<td>446.3</td>
<td>17.85</td>
</tr>
<tr>
<td>Hₚ, fast</td>
<td>13.6104</td>
<td>- - -</td>
<td>26</td>
<td>379.6</td>
<td>14.60</td>
</tr>
<tr>
<td>Hₚ, cloudy</td>
<td>6.6879</td>
<td>+++</td>
<td>38</td>
<td>691.9</td>
<td>18.21</td>
</tr>
</tbody>
</table>

The results from table 4.5 and 4.6 show that the cloudy weather dataset produces volume estimates closest to the manual volume estimates (lowest MRPD). The fast truck dataset produces the second closest results, but with a low amount of frames and an underestimated volume for each timber load. The fast truck dataset also has the highest difference in MRPD between update model 1 and 2. The sunny dataset, which contains the highest amount of frames by far, produces the second highest MRPD. These datasets were chosen for the qualitative evaluation, where the surface reconstruction results are compared to the corresponding
3D reconstruction.

Figure 4.5 shows selected images of the second timber load from the 3D reconstruction of the cloudy weather dataset. Figure 4.6 shows selected images from the surface reconstruction of the second timber load in the cloudy weather dataset using update model 1.

\textbf{Figure 4.5:} Selected images of the second timber load from the 3D reconstruction of dataset $H_{r,\text{cloudy}}$. 

\textbf{Figure 4.6:} Selected images from the surface reconstruction of the second timber load in dataset $H_{r,\text{cloudy}}$ using update model 1.

The figures show a mostly continuous reconstructed surface with few holes, but open ends due to missing data. The open ends are likely to cause the overestimation of unseen space voxels.
Figure 4.7 shows selected images of the first timber load from the 3D reconstruction of the sunny weather dataset. Figure 4.8 shows selected images from the surface reconstruction of the first timber load in the sunny weather dataset using update model 1.

![Figure 4.7: Selected images of the first timber load from the 3D reconstruction of dataset $H_{r,sunny}$.](image)

The figures show a mostly continuous reconstructed surface with few holes, but the front end is open due to missing data. The rear end is mostly closed, but of very poor quality. Much of the crane and three poles have been reconstructed, but also of very poor quality. The reconstruction of the crane and the poles are likely to add to the overestimation of the volume.

Figure 4.9 shows selected images of the first timber load from the 3D reconstruction of the fast truck dataset. Figure 4.10 shows selected images from the surface reconstruction of the first timber load in the fast truck dataset using update model 1. Figure 4.11 shows selected images from the surface reconstruction of the first timber load in the fast truck dataset using update model 2.
4.2 Real data

Figure 4.9: Selected images of the first timber load from the 3D reconstruction of dataset $H_{r,fast}$.

Figure 4.10: Selected images from the surface reconstruction of the first timber load in dataset $H_{r,fast}$ using update model 1.

Figure 4.11: Selected images from the surface reconstruction of the first timber load in dataset $H_{r,fast}$ using update model 2.

Figure 4.9 shows a somewhat sparse 3D reconstruction, with missing data in both
ends as well as in the region surrounding the crane. This has produced corresponding holes in the surface reconstruction, where the crane hole consequently has left a large amount of voxels as empty space instead of unseen space (see figure 4.10b). This is likely to explain the underestimation of volume. The same problem becomes more severe with update model 2 (see figure 4.11), as fewer voxels are updated as surface which leads to a larger crane hole.
This chapter presents a discussion about the results achieved by the proposed method, as well as a discussion about the method’s general benefits and shortcomings.

5.1 Results

The results are discussed in terms of test rig data and real data. The test rig data results are mainly discussed by comparing between the two geometric objects and the demands for precision stated in section 1.2. The real data results are discussed by comparing the surface reconstructions from the different datasets.

5.1.1 Test rig data

The results achieved for both geometric objects suggest that the proposed method performs much better on reconstructions of the rectangular box than those of the pyramid. For the rectangular box, the MRPE was around 1-2% and the PSD 1-1.5%, to compare with the pyramid’s MRPE between 8-20% and PSD 12-16%. In other words, the proposed method is more accurate and precise for the rectangular box than for the pyramid. The main explanation for this may lie in the combination of method for calculating the volume and the geometric shape of the pyramid. Since the volume is calculated by counting the number of quadratic voxels belonging to the surface and the unseen space state, it is hard to get an accurate volume estimate of the triangular shape of the pyramid without a high grid resolution. The results in table 4.3 and 4.4 confirm this, as volume estimation accuracy increases dramatically with grid resolution. However, the results for the highest grid resolution are still relatively poor, which suggests that there
still are other large sources of error affecting the volume estimation.

The results for the rectangular box are better than expected. Not only is the MRPE low, the PSD measures for both update models well meet the demands for precision stated in section 1.2. Furthermore, the results are fairly invariable to grid resolution, which further solidifies the theory that the method for volume calculation is favourable for quadratic reconstruction objects. However, as figure 4.2 shows, the surface reconstruction of the rectangular box is still far from ideal, even with the highest resolution. Imperfect disparity maps are of course a large part of the explanation for this, but since the results are still fairly good, this also suggests that the volume estimation might not be as accurate as it seems. There could be other factors affecting the volume estimation that happen to cancel each other out.

A higher grid resolution has obvious advantages in terms of surface reconstruction quality, but the execution time scales cubically with it. This means that the choice of grid resolution should be a cautious one in order for the method to be performing at real-time rates.

The two update models perform approximately the same for each geometric object and grid resolution. Update model 2 has a slightly lower MRPE for both objects, but its PSD is slightly higher for the rectangular box. These differences in MRPE and PSD are still relatively small compared to the differences seen between different grid resolutions.

5.1.2 Real data

The comparison between the proposed method’s volume estimates and the manual volume estimates suggests that the proposed method performs the worst on datasets recorded during nighttime. However, the second highest MRPD was achieved on the sunny weather datasets, using the most amount of frames by far on average. This is slightly contradictory, as a higher number of frames should correspond to a better surface reconstruction. The results shown in figure 4.8 suggest that the overestimation of volume may be explained by the fact that much of the crane and poles have been reconstructed, which are parts not included in the manual volume estimation.

The cloudy weather dataset produces results closest to the manual volume estimates. The results in figure 4.6 show that both ends of the load are open due to missing data, but the rest of the surface is cohesive. The diffuse lighting conditions may be a key contributational factor for this.

The fast truck dataset produces the second lowest MRPD, but the volume is underestimated for each load. The results in figure 4.10 show that this might be due to the missing data region around the crane that causes a large amount of voxels to be updated as empty space instead of unseen space. The corresponding
results for update model 2 in figure 4.11 show that this problem is aggravated as even fewer voxels represent the surface around the crane, leaving a larger missing data region. However, the same argument of missing data does not hold for the remaining timber loads, which means that there is no evident reason for the underestimation of volume.

5.2 Method

The choice of a volumetric integration method was primarily motivated by the results of [6] and [3]. The results achieved were not only of comparable quality to current state of the art techniques, the time frame in which they were produced was also considerably shorter. This fact was one the main motivations of proceeding with a volumetric integration method, as execution time was favourably kept as low as possible.

Another large motivation was the promise of parallelization. Although volumetric methods generally suffer from poor scalability due to the imposition of a large memory footprint, the independent voxel calculations provide a key possibility for performing calculations on the GPU. No measures were taken in this thesis work to parallelize calculations due to time constraints, but neither was this a goal as much as to provide a scalable method.

Through the use of voxel projection, calculations are possible to parallelize. This will most likely shorten the execution time of the method substantially, but it does not guarantee scalability. The results in section 4.1 show that the time complexity increases rapidly with resolution, which means that parallelization efforts could prove redundant for large enough reconstruction objects. Also, projecting each voxel onto the disparity map of each stereo pair is a fairly inefficient way to deduce observed information, and further adds to the computational complexity. There are, however, several ways to streamline calculations. Using a hierarchical data structure for example, as Chen et al [4], the surface can be more densely represented than empty and unseen space. This allows finer surface reconstructions, while decreasing the amount of calculations performed for the less relevant empty space voxels.

The results for the real datasets nonetheless suggest that the proposed method suffers from more severe issues than execution time when estimating the volume of timber loads. Large missing data regions at the ends of the timber loads contribute to high variations in estimated volume. Poor surface reconstruction quality is especially noted for sharp geometries, like the protruding supporting poles. This can both be a resolution issue as well as an update model problem. The general assumption, judging from the results, is that the more sophisticated update model 2 has no distinct increase in reconstruction quality. Instead, it disregards a higher amount of surface observations, leaving larger holes in the
resulting surface. The reason for this is unclear, while it seems as the intended increase in certainty for each observation has rather resulted in fewer chosen observations. The low amount of evaluated real datasets also needs to be adressed, which means that no major conclusions can be drawn for this type of dataset. The results merely indicate that large missing data regions can both lead to over- and underestimation of the volume, and that the method produces volume estimates that are fairly inconsistent with manual volume estimates.

The method for calculating the volume is fairly crude, and highly dependant on grid resolution for certain geometric objects (see the results in table 4.3 and 4.4). A better approach might be to focus on producing a watertight surface representation, and calculate the volume by means of the divergence theorem. A few hole filling approaches were in fact considered, but time constraints limited extensive experimentation of any approach. The idea of estimating the volume from a parametric representation, however, remains intriguing. The combination of such a solution and an adaptive data structure, which represents the surface more densely, could theoretically yield promising results. However, the large missing data regions of the timber loads remain a major challenge for hole filling approaches, which means that a completely accurate surface representation may not be found.

### 5.2.1 Ambiguities of multiple stereo pairs

As mentioned in section 3.2.1, the main contribution of the proposed method may lie in the handling of simultaneous observations from multiple stereo pairs. The major challenge that this circumstance entailed was how to determine the most suitable observation for a given voxel. Evidently, the main reason for this type of ambiguity was the choice of voxel projection as the method for deducing observed information of the scene. For clarification of this statement, consider the three examples of voxel projection illustrated in figure 5.1, 5.2 and 5.3.

In figure 5.1, the voxel projections result in positive signed distances (green) observed for all three stereo pairs. Consequently, determining that the voxel represents unseen space is straight-forward. In figure 5.2, the voxel has a negative signed distance (red) in stereo pair 1, while it has positive signed distances in stereo pair 2 and 3. In this case, deciding which observation is the most correct is less obvious. A majority of the stereo pairs see positive signed distances, which intuitively might seem as enough to conclude the voxel as unseen space. On the other hand, stereo pair 1 is the only stereo pair with an unoccluded view of the voxel, which means that its observation is the most trustworthy. Fortunately, stereo pair 1 also sees the shortest signed distance, which means that, in this case, even the simpler update model 1 suffices to determine the correct observation.
Figure 5.1: Example 1 of voxel projection for voxel marked in red. The voxel projections result in positive signed distances observed for all three stereo pairs, which means that the voxel can be updated to the unseen state without ambiguity.

Figure 5.2: Example 2 of voxel projection for voxel marked in red. The voxel projections result in a negative signed distance in stereo pair 1, and positive signed distances in stereo pair 2 and 3. In this case, determining the state of the voxel becomes less obvious.
Lastly, in figure 5.3, the voxel no longer has a corresponding observation in stereo pair 1. This means that the only observations available are from stereo pair 2 and 3, which both are occluded. In this case, we cannot simply choose the shortest signed distance as this will lead to the voxel being falsely updated as unseen space. However, by introducing the visibility threshold \( \lambda \), this issue can be prevented as we can specify that the voxel needs to be seen as the same state by all three stereo pairs in order to be updated.

![Figure 5.3: Example 3 of voxel projection for voxel marked in red. The voxel projections result in positive signed distances in stereo pair 2 and 3, while no corresponding observation exists in stereo pair 1. Choosing the shortest signed distance here results in a falsely updated voxel.](image)

In all of the examples so far, the signed distances have been larger than the truncation distances, which means that the update problem is about determining whether the voxel is empty or unseen space. When at least two stereo pairs see a signed distance within the truncation distances, we still want to use one of the signed distances to update the SDF, even if the visibility threshold is not satisfied. In this case, a more complex update model than update model 1 is needed.

This type of ambiguity when choosing observation is the motivation behind the more complex update model 2. By assigning a weight to each observation based on viewing angle, distance to the voxel and observed signed distance, the observation with the highest weight should correspond to the optimal choice. However, as discussed earlier, this is generally not synonymous to a better surface reconstruction or volume estimation. This fact suggests that update model 2 is not sufficiently robust, and needs further elaboration to better handle the geometric
5.2 Method

complications that come with voxel projection.

A question that needs to be raised is whether raycasting, as used by Curless and Levoy [6], would have been a better choice of method than voxel projection. As information about the scene is deduced along the line of sight in each stereo pair, the risk of ambiguity during voxel update should be considerably lower. Consequently, the main difference compared to voxel projection is that we only update the voxels known to be empty space, rather than try to determine which of the two states each individual voxel belongs to. Naturally, this implies that we have to initialize every voxel to the unseen state, instead of the empty state as is currently done. When all observations have been integrated, the resulting unseen space voxels are then those who have not been updated.
In this chapter, conclusions about the work of this thesis are presented. These conclusions are mainly made in regard to the questions stated in section 1.4. Lastly, some suggestions are made about future work.

6.1 Conclusions

The proposed method for automatic volume estimation can achieve promising results under specific circumstances, but lacks robustness to a high amount of missing data. Out of the two evaluated geometrical objects, the method performed the best for reconstructions of the rectangular box using the highest grid resolution and update model 2, achieving a MRPE of 1.046% and a PSD of 1.224%. The method performed the worst for reconstructions of the pyramid using the lowest grid resolution and update model 1, achieving a MRPE of 19.855% and a PSD of 15.694%.

The results from the real datasets suggest that the proposed method is not robust to varying weather and lighting conditions. This is motivated by the poor handling of missing data regions and the varying MRPD towards the manual volume estimate. However, the low amount of evaluated real datasets means that the results are rather inconclusive, and that the robustness aspect needs to be evaluated more thoroughly.

All goals stated in section 1.3 are fulfilled with the proposed method. However, only the results achieved for the rectangular box satisfy the demands on precision stated in section 1.2. In order to satisfy all of the stated aims, robustness needs to be improved for both environmental factors and the shape of the reconstruction object. Tentatively, this could be achieved by including a hole-filling approach.
along with an increased grid resolution. The SDF should in this case be stored using an adaptive data structure to prevent an overwhelming imposition of memory.

In terms of scalability, the use of voxel projection allows for parallelization of calculations. However, without an efficient data structure, the efforts of parallelization might prove redundant as an increased grid resolution leads to a dramatic increase in execution time. Furthermore, voxel projection as a method for deducing observed information of the scene has obvious flaws. This is especially the case when using multiple stereo pairs simultaneously, as choosing the best stereo observation for a given voxel can be ambiguous. To resolve this ambiguity, a second update model which accounted for observation angle, observed signed distance and distance to camera was implemented. This update model produced marginally better results for the test rig datasets, but resulted in larger missing data regions for the real datasets. Although carrying theoretical potential, the update model might not be sufficiently robust to resolve the multiple stereo pair ambiguity. Instead, the use of raycasting as an alternative to voxel projection should be investigated. This is motivated by the fact that information is deduced along the line of sight, which means that updating each voxel theoretically becomes less ambiguous.

6.2 Future work

For future elaborations of this work, raycasting should firstly be investigated as an alternative method to voxel projection for deducing information about the scene. The primary reason for this is to determine its effect on volume estimation accuracy as well as execution time. Given little or no apparent effect, robustness to weather variations should be studied and compared to that of voxel projection. Furthermore, as the performance of the proposed method proved to be dependant on the geometrical shape of the reconstruction object, investigating additional types of shapes should be interesting. If the method then proves to be too dependant on geometrical shape, focus should be shifted towards calculating the volume in a way less dependant on grid resolution. This could, for instance, be a hole-filling approach, where the volume then is calculated directly from the parametric representation of the SDF using Gauss’ divergence theorem.


