Code:_____

Faculty of Engineering and Sustainable Development

Evaluation of 3D Reconstructing Based on Visual Hull Algorithms

Linus Fredriksson
June 2011

Bachelor Thesis, 15 credits, C
Computer Science

Computer science program
Examiner: Julia Åhlén
Supervisor: Stefan Seipel

Evaluation of 3D Reconstructing Based on Visual Hull Algorithms

by

Linus Fredriksson

Faculty of Engineering and Sustainable Development
University of Gävle

S-801 76 Gävle, Sweden

Email:
linfredriksson@gmail.com
ndv08lfn@student.hig.se

Abstract

3D reconstruction of real world objects is becoming more and more popular among computer graphics and computer vision researchers. One of the more practical approaches of achieving this is called Multi View Geometry, a approach that are using images taken of the real world objects to recreate it. But it is not always easy to know how to get the desired result out of this approach because there are many variables that affects the reconstructions accuracy, for example the number of images used and the resolution of these images.

In this paper a silhouette based 3D reconstruction algorithm is evaluated. Three programs are created each with its own purpose. The first program is used to create as perfect silhouette images as possible in order to get as accurate input data as possible. The second program uses these silhouettes and produces a volumetric reconstruction of the object being reconstructed. The third program creates a polygonal mesh from the volumetric data. The polygonal and volumetric reconstructions are then used when evaluating the visual and volumetric accuracy of the reconstructions.

The implemented algorithm is capable of running at real-time or near real-time and produces reconstructions with a high accuracy. It is shown how different silhouette resolutions and the resolution of the volumetric reconstructions influence the accuracy of the created reconstructions. It is also shown that the biggest gain in accuracy related to the number of silhouette images used is gained when increasing from one silhouette and up to ten silhouettes and that by increasing the number of silhouettes more only a low improvement in accuracy is gained.

Keywords: 3D reconstruction, Visual hull, GPGPU, Stencil buffer, Shape from silhouettes, Projective texture mapping

Contents

1 Introduction

3D object reconstruction of real world objects is a popular subject among computer graphics and computer vision researchers. 3D reconstruction is an important subject and can be used in many different areas. It can be used in medical visualization, machine vision, multimedia and many other fields. There are many different approaches to 3D reconstruction such as stereo vision, multi view geometry, tomographic reconstruction and medical scans. These methods use different ways of collecting the data needed for reconstruction, some use images others use MRI and so on.

One of the approaches that use 2D images also referred to as silhouettes is multi view geometry. Multi view geometry based algorithms uses multiple silhouettes to reconstruct the real world objects and these algorithms involves three different steps. The first step is to obtain the silhouettes and there are different methods for this. Szeliski [1] is using a turntable when obtaining the silhouettes. When using a turntable a real world object is placed on the turntable that is then rotated in steps while a camera is used to take a photo of the object between each rotation. Ladikos et al. [2] on the other hand use a fixed setup of 16 cameras mounted in a room. The Euclidian space containing the possible camera placements are referred to as viewing region and Laurentini [3] shows that different viewing regions may give different results.

After an image have been obtained the object silhouette must be extracted, a process that can be done with an intensity threshold as mentioned in [1, 3], by using foreground/background subtraction as in [4, 5] or by a combination of the two [2].

When the images are collected and silhouettes have been found the next step is to identify the camera parameters. According to Szeliski [1] these parameters include the cameras position relative the object, orientation and focal length.

The last step of multi view geometry based algorithms is the actual 3D reconstruction. Liang and Wong [6] classify the 3D reconstruction algorithms using 2D images as being part of one of two schools, the volumetric school and the differential school. Liang and Wong [6] describe the volumetric school as “treating objects as solid volumes” they also say that the differential school is defined by the fact that it “treats the object surface as an infinitesimally thin shell”. 3D reconstruction algorithms belonging to both schools are widely used and the approach introduced by Liang and Wong [6] is an example of an algorithm belonging to the differential school.

There is also many examples of algorithms [2, 3, 4, 5, 7] belonging to the volumetric school. One of these algorithms is called volume intersection and is described by Laurentini [3]. Laurentini says that volume intersection creates the 3D reconstruction by identifying the region of space that is completely covered by all the silhouettes and that the more silhouettes that are used the more accurate the 3D reconstruction become. Laurentini go on to explain that this is done by back projecting cones through the silhouettes and that the region of space contained by all the different cones is the closest approximation of the real world object that can be defined using the existing silhouettes. This region is also called visual hull, a term introduced by Laurentini [3] and is illustrated in Figure 1.

To improve on the concept of volume intersection Szeliski [1] introduces the usage of an octree to decrease the needed calculations and so also speed up the reconstruction algorithm. Ladikos et al. [2] examines the runtime when using an octree and when not using an octree and comes to the conclusion that for small voxel resolutions the implementation not using an octree is faster but when using a larger voxel resolution the implementation using an octree is preferred.

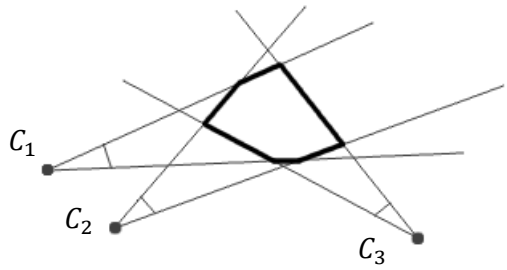

Figure 1. Illustration of a visual hull in 2D using three silhouette cones called C_1, C_2 and C_3.The area enclosed by a thicker line is the visual hull.

As the graphics hardware becomes more powerful another type of algorithms belonging to the volumetric school becomes more popular. These algorithms [2, 4, 7, 8, 9] perform the reconstruction with the help of graphics hardware.

Ming et al. [8] introduces a reconstruction algorithm that uses the GPU to perform the reconstruction and later as the graphics hardware grew more powerful they introduced new and improved algorithms [7] not limited by older graphics hardware. The reconstruction algorithm in [7] creates volumetric 3D reconstructions by loading silhouette images as textures to the CPU and then projecting them onto one of the volume layers at a time gradually building the reconstructed object.

In regards to limitations to the accuracy of the visual hull gained from using silhouette based reconstruction algorithms some research has been done by Laurentini [10]. In his research he explains that due to the fact that silhouettes may be unable to accurately identify concavities there is no guaranty that an object can be completely reconstructed. Because of this Laurentini describes the visual hull as being equal or bigger than the original object.

In another paper Laurentini [11] has also done some research to identify the theoretical number of silhouette images needed to exactly reconstruct a 3D object and he comes to the conclusion that for a polyhedron with n faces it takes O(n^5) silhouettes, given that the object can be exactly reconstructed, i.e., no concavities. This research will evaluate the reconstruction algorithm in a more practical way to see what accuracy can be achieved using a practical implementation.

1.1 Goal

The goal of this research is to gain an understanding of the reconstruction accuracy that can be achieved by using the above mentioned hardware accelerated 3D object reconstruction.

This will be done by evaluating the impact different variables have on the resulting 3D reconstructions accuracy. These variables includes the number of 2D silhouettes used when reconstructing an object, the pixel resolution of these silhouettes and the voxel resolution of the resulting volumetric reconstruction.

By better understand how these variables affect the reconstructions accuracy it will be easier to gain desired reconstruction accuracy.

2 Visual hull reconstruction algorithm

The visual hull reconstruction method used for this evaluation consists of five different steps. The first three steps are collecting images from the object to recreate, identify the configurations of the camera used to create the images and the extracting the silhouettes from the images. The last two steps are the actual reconstruction of an object and then the rendering of the reconstruction.

This chapter will describe how these steps were implemented for the purpose of evaluating the reconstruction step of the process. First the hardware used in the

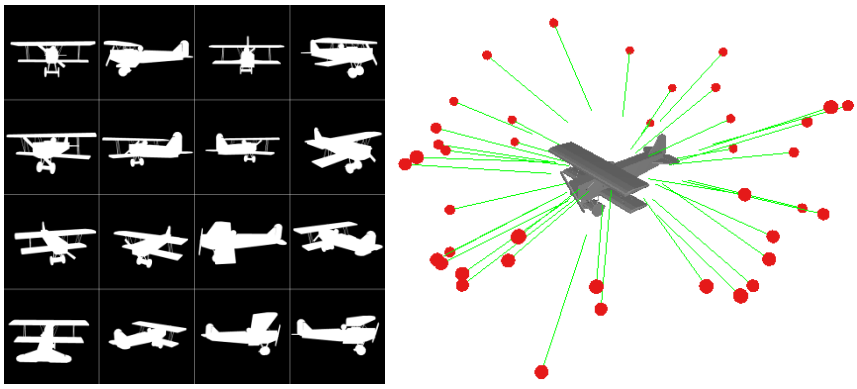

Figure 2. ***Left:*** *Sample of 16 out of 40 silhouettes taken from different viewpoint around an airplane model.* ***Right:*** *Viewpoint positions of the 40 silhouettes taken from an airplane model.*

research will be stated. Then the first three steps of the reconstruction process will be explained. Followed by a detailed explanation of the reconstruction algorithm and lastly the reconstruction rendering will be explained.

2.1 Hardware

All the programs used and the tests performed have run on a PC equipped with a 2.67GHz Intel Core i7 CPU, 6GB of RAM memory and a RADEON HD 5850 graphics card with 1GB GDDR5 memory.

2.2 Silhouette extraction

The first two steps in silhouette based 3D object reconstruction are the acquisition of silhouette images and the extraction of the silhouettes.

The acquisition of silhouette images can be done with the help of a turn table or by using a number of fixed cameras. There is also a need to obtain the used cameras settings to accurately use the silhouette images in a later stage.

When the images have been collected the objects silhouettes needs to be extracted, examples of silhouettes can be seen in Figure 2. This is an important part of the reconstruction process that is usually performed by using methods like intensity threshold where all pixels in a certain interval are considered to belong to a foreground object and the rest is considered to be background, foreground/background subtraction where an image taken on the scene without the object being present is used to identify the foreground object or by combining the two methods.

When performing the silhouette extraction it is important to get as accurate silhouettes as possible and to avoid static occluders. A static occluder is an object that is in the way of the object of interest, like a table or another object.

To eliminate all problems related to gathering and extracting the silhouette images when doing this evaluation, a program has been implemented that uses existing 3D models to create the silhouette images. This program starts by reading the vertex and normal data from existing model files, which is then stored in the program and used to render the models.

The models loaded into the program can be rendered with the Phong illumination model described in the following chapter or as perfect silhouettes which is done by setting the color of the models to white and everything else as black when the model is rendered. In this implementation this is done using shader programs.

The program makes it possible to move the camera around the model so that it can be observed from any viewpoint. When a model is viewed from a specific viewpoint the program is able to save the rendered image containing the object to an

image file and at the same time save the cameras parameters such as position, alignment, field of view and aspect ratio to a text file containing camera settings.

This program is based on the same code that is later used to reconstruct and render the 3D reconstructions and this guarantees that all camera parameters can be exactly transferred to the reconstruction algorithm.

Figure 2 illustrates 16 examples of silhouettes generated using this program and 40 viewpoints used to generate these and 24 other silhouettes.

2.2.1 Phong illumination model

The models loaded into the silhouette extraction program can as earlier stated be rendered either as a simple silhouette image or shaded using an illumination model. In the case of the shaded option the Phong illumination model [12] is used. Equation 1 describes the diffuse term, equation 2 describes the specular term and the ambient term does not contain any spatial information and is thus simply defined as a color.

$$I_d(x) = L_{d,in} * k_d * \max\left(\left|\nabla r\left(f(x)\right)\right| * L, 0\right)$$

Equation 1. Phong illumination model, diffuse term, where $\left|\nabla r\left(f(x)\right)\right|$is the normalized gradient and L is the normalized light vector.

$$I_s(x) = L_{s,in} * k_s * \max\left(\left|\nabla r\left(f(x)\right)\right| * H, 0\right)^{\alpha}$$

Equation 2. Phong illumination model, specular term, where $\left|\nabla r\left(f(x)\right)\right|$is the normalized gradient and H is the normalized half-way vector.

2.3 Visual hull reconstruction

The implemented reconstruction algorithm described in this chapter is based on the algorithm presented by Kim et al. [4] and the algorithm described by Li et al. [7].

Before the algorithm starts the reconstruction process the silhouette images created in the above described silhouette extraction program are loaded into a 3D texture on the graphics hardware. This is done by taking the silhouette images and stacking them on top of each other so that they form a 3D stack of pixel values.

For each of these silhouette images a texture matrix is also created by using the corresponding camera parameters also obtained from the above mentioned silhouette extraction program.

The output of the implemented algorithm is another 3D texture containing the volumetric reconstruction of the original object. Because that most graphics hardware is only able to render onto 2D images the reconstruction algorithm creates slices that are then in the same manner as the silhouette images stacked on top of each other to form a 3D data volume.

Figure 3 contains pseudo code giving a quick overview of the implemented reconstruction algorithm that will now be described in more detail. Because most modern hardware is only able to render 2D images the reconstruction needs to be rendered one layer at a time. This means that if the reconstructed volume has a resolution of 256^3 then 256 2D slices with a resolution of 256^2 needs to be created and put together to form this 3D volume.

Each of these slices is created in two steps. First all of the silhouette images are projected onto the slice currently being created and then the area covered by all the silhouettes, also known as the visual hull illustrated in Figure 1, is set to one color and the rest of the slice is set to another color. These two steps will now be explained more thoroughly.

In the first step the algorithm goes through all silhouette images stored in the 3D texture one by one and by using the silhouettes individual texture matrices they are projected onto the slice currently being rendered, this is also called texture mapping

```
foreach slice I
   Bind frame buffer object
   Clear stencil buffer, color buffer and depth buffer
   Enable stencil test
   Disable writing to color buffer
   Disable depth test
   Stencil function, always
   Stencil operation, increase
   foreach silhouette j
      Set up the projective texture matrix for silhouette image j
      Draw to stencil buffer
   End foreach
   Enable writing to color buffer
   Enable depth test
   Stencil function, equal number of views
   Stencil operation, keep
   Draw to color buffer
   Disable stencil test
   Unbind frame buffer object
   Save slice (frame buffer objects color buffer)
End foreach
```

Figure 3. Pseudo code describing the implemented reconstruction algorithm.

[13]. By using texture matrices the silhouettes can be projected onto any surface in the same way a projector would be able to project any image onto any surface.

This method with texture mapping can also be used when creating shadows in 3D games or 3D visualizations as seen in [13, 14, 15]. In these cases an image containing depth information is rendered from the point of the light source, for example a sun. Then the scene is rendered again from another viewpoint and the first image is projected onto the scene using a texture matrix in the same manner as mentioned above. When rendering the scene the second time the areas in shadows is determined by comparing a points distance from the light source with the value projected to that point from the depth image and if the depth images depth value is less than the points distance to the light source then something is between the point and the light source and the point should be covered by a shadow.

When the implemented reconstruction algorithm is going through the silhouettes and projecting them onto the slices the algorithm needs to keep track of what part of the slices is covered by the objects in the silhouettes. This cannot be done if the silhouettes were to be projected onto a color buffer because it is not possible to both read and write to a color buffer at the same time. So if the silhouettes were projected onto a color buffer one at a time the only silhouette information saved in the color buffer at the end would be the last projected silhouettes. In order to keep track of where the silhouettes are projected the implemented algorithm turns of rendering to the color buffers completely and instead renders to a stencil buffer. In contrast to a color buffer where its existing values are replaced when something is rendering to it a stencil buffer can be updated when rendered to multiple times as illustrated in Figure 4 to the right. This makes is possible to define what the stencil buffer should do when something is rendered to it. Witch in the case of the implemented reconstruction algorithm is to increase the stored value by one. Which means that where one silhouette has been projected onto the stencil buffer the number one is stored, where two silhouettes have been projected the number two is stored and so forth. Giving the program the ability of looking at the stencil buffer and knowing how many of the projected silhouettes that cover a specific point.

When all projected silhouettes have been rendered onto the stencil buffer the stencil buffer operation changes from increase to always keep the values stored.

Figure 4. ***Left:*** *Result from reconstruction algorithm.* ***Right:*** *Example of stencil buffer first with one silhouette rendered then three silhouettes. The numbers indicate the number of projected silhouettes covering different parts of the buffer.*

Rendering to color buffers are then turned back on again for one final rendering but instead of rendering to the screen the algorithm renders to a color buffer stored in a frame buffer object so that it is possible to later access it again and save its content, which will be a finished slice. This final rendering sets the color buffer to the color white and the alpha value one on all places where the value in the stencil buffer is the same as the number of silhouettes used. Giving the currently created slice white color at all points covered by all silhouettes or in other words one slice of the visual hull is created.

This slice is then saved by adding it to the end of the 3D texture soon to be containing a complete 3D reconstruction of some object.

When this process has been performed for all the slices the implemented reconstruction algorithm has done its job and a 3D reconstruction of the original object has been created and stored in a 3D texture ready to be displayed using the method explained in the following chapter.

2.4 Visual hull rendering

When a reconstruction has been created and saved it needs to be rendered. For this a program has been implemented that renders the reconstruction by converting the 3D volume created in the above described process into a polygonal mesh using a simple implementation of the marching cube algorithm. The marching cubes algorithm was first introduced by Lorensen and Cline [16] and is a well known and popular algorithm for creating polygonal meshes from volume data. One example of its usage is [17] where it is used to render medical data.

2.4.1 Marching cubes algorithm

When the implemented marching cubes algorithm creates the polygonal mesh from the 3D volume data it starts by determine where the surface polygons should be located.

The marching cubes algorithm creates the surface polygons by marching through the data volume collecting neighboring samples of eight forming logical cubes. By assigning a value to each of these eight points depending on if they are inside the object or outside the object these eight sample points determine how the surface should intersect the 3D volume at a specific location.

In the implementation used for this research the 3D volume created in the above mentioned program contains either white or black that is stored as one or zero. This means that every point with value one is considered to be inside of the object and every other point has value zero and is considered to be outside of the object. Now because that eight sample points are used and these points are either outside of inside of the object there are 2^8 or 256 different combinations possible. These combinations can be narrowed down to 15 different combinations or cases, shown in Figure 5 that can form all of the other cases by using rotation and inverting of values.

So by giving all of the 256 cases an individual index depending on what points are inside or outside of the object it is possible to identify how the surface should divide these eight sample point by using these same sample points to create and index

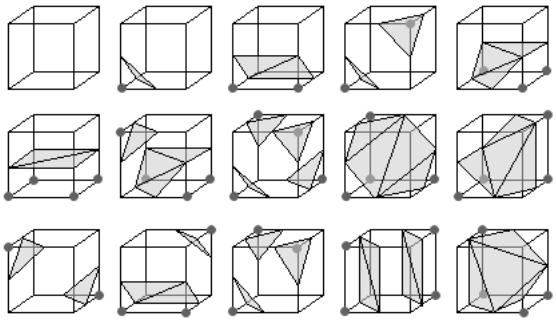

Figure 5.The 15 unique cubes cases. A dot indicates active point and is separated from non active corners by a polygonal surface.

corresponding to one of the cases index.

This index number is created by simply placing the eight sample point values in a row to form a binary integer. This integer will have a value ranging from zero to 255 depending on which sample points that are inside and which sample points that are outside of the object.

When the implemented algorithm has found the right intersection case by creating the index number and finding the intersection case with the same number the intersection cases polygons are created and stored in a list later used to render the object.

The algorithm also calculates the vertex normal of all vertices. This is done by taking the difference between neighboring points values in each axis direction, giving the gradient of the points.

2.4.2 Reconstruction shading

Once the reconstruction has been converted from volumetric data to a polygonal mesh it needs to be rendered. In the rendering the Phong illumination model is used in the same manner as when rendering the original model in the silhouette extraction program described in the beginning of this chapter.

3 Methods for evaluation of reconstruction algorithm

This paper evaluates the 3D reconstruction in three different aspects. These are reconstruction time, model accuracy and visual accuracy and will be explained in this chapter.

3.1 Reconstruction time

Reconstructing time is an important part of the process and is measured when performing reconstructions using different silhouette and volume resolutions as well as different numbers of silhouettes.

When measuring the time only the actual reconstruction process is measured, silhouette extraction and rendering of the result is not included in this test. This allows the evaluation to accurately measure the time it takes to reconstruct different objects without being influenced by badly optimized rendering algorithms or silhouette extraction algorithms.

3.2 Model accuracy

One of the two methods used to measure the accuracy of the 3D reconstruction is to simply count the number of voxels in the reconstructed volume that are active, i.e., voxels belonging to the reconstructed object. By creating reconstructions using

increasing number of silhouettes and then count the number of active voxels in these reconstructions it shows an image of the impact increasing number of silhouettes has on the reconstruction accuracy. The same is true regarding different silhouette resolutions.

3.3 Visual accuracy

The second of two methods used to measure accuracy measures the visual aspects of the 3D reconstructions. This is done by comparing images taken on the original model with images taken on the reconstructed model. These images are taken from viewpoints not used in the reconstruction process due to the fact that if viewing the reconstruction from one of these viewpoints it should be identical to one of the silhouettes used in the reconstruction.

Comparison between images is done by simply taking images on the original object and the reconstruction from the same viewpoint. These images are then used to create a new image containing all pixels not appearing in both images. Figure 6 illustrates the comparison of two silhouette images taken on a plane model and the reconstruction of that model.

When a difference image is created the number of pixels not belonging to the background is counted both in the difference image and the image of the original model. By dividing the number from the difference image with the number from the original image an error factor is found as in equation 3.

This process is performed both with silhouette images and images containing a shaded version of an object and its reconstruction to both evaluate the accuracy of the models shape and how the accurate of the surface of the reconstruction appears.

$$error = \frac{n\ diff\ pixels}{n\ pixels}$$

Equation 3. Calculation of error in visual accuracy test where "n diff pixels" is the number of active pixels in a difference image and "n pixels" is the number of active pixels in the original image.

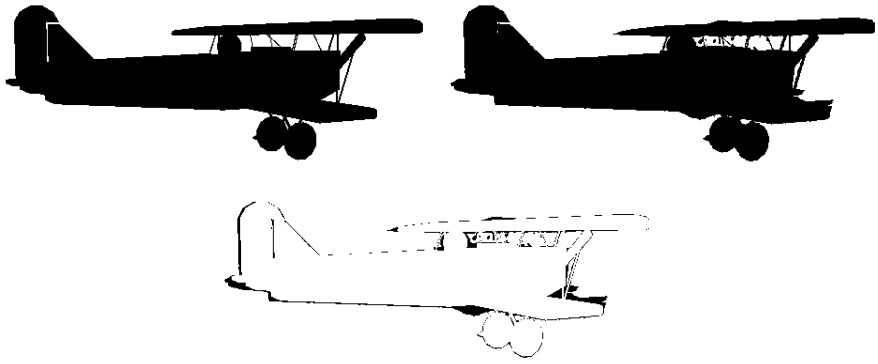

Figure 6. Illustration of how the visual accuracy is evaluated by comparing silhouette images. ***Top left:*** *Silhouette from original model.* ***Top right:*** *Silhouette from reconstructed model.* ***Bottom:*** *The difference between the silhouettes from the original model and the reconstruction.*

4 Result

An OpenGL based engine has been implemented and three programs have been built around it. The first of these programs is capable of loading and viewing 3D models used to create 2D silhouette images. The second program contains a 2D silhouette based 3D reconstruction algorithm that uses the graphics hardware for fast and reliable

reconstruction of objects. The third program is used to render the reconstruction using a simple implementation of the marching cubes algorithm.

The reconstruction algorithm has been evaluated in regards of rendering time, model accuracy and visual accuracy and the results will all be presented in this chapter.

All statistics are made from reconstructions of the plane model shown in Figure 7. These reconstructions are made from silhouette images taken on the plane model from the same 40 viewpoints that are shown to the right in Figure 2.

4.1 Reconstruction examples

Figure 7, 8, 9 and 10 give an overview of what the implemented 3D reconstruction algorithm is capable of creating. In these figures objects created in a modeling program is shown next to corresponding 3D reconstruction. All of which is created from 40 silhouette images with a resolution of 1024 by 1024 pixels using a reconstruction volume size of 512^3 units.

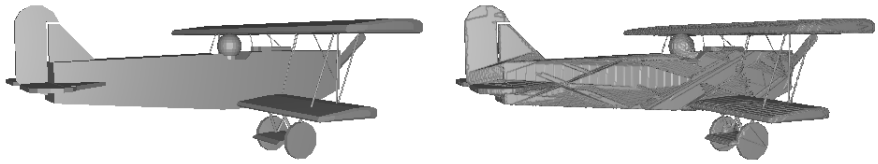

Figure 7. 3D object and its 3D reconstruction both shaded using the Phong illumination model. ***Left:*** *Airplane model.* ***Right:*** *Reconstruction.*

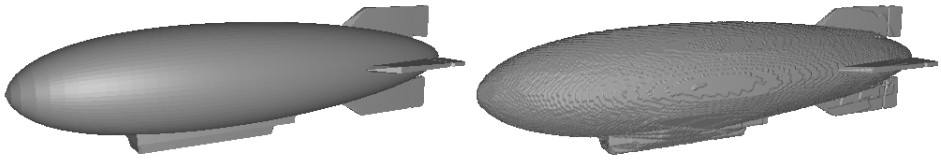

Figure 8. 3D object and its 3D reconstruction both shaded using the Phong illumination model. ***Left:*** *Zeppelin model.* ***Right:*** *Reconstruction.*

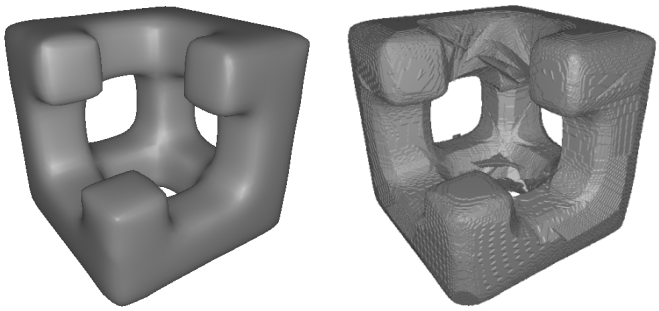

Figure 9. 3D object and its 3D reconstruction both shaded using the Phong illumination model. ***Left:*** *Complex model.* ***Right:*** *Reconstruction.*

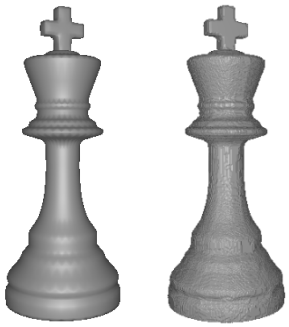

Figure 10. 3D object and its 3D reconstruction both shaded using the Phong illumination model. ***Left:*** *Chess piece model.* ***Right:*** *Reconstruction.*

4.2 Reconstruction time

All time measurements have been taken while recreating the plane model shown in Figure 7, multiple times using different settings. Time measurements have been taken to evaluate reconstruction time for different reconstruction volume resolutions, silhouette image resolutions and number of silhouette images.

Figure 11 illustrates the rendering time in milliseconds while creating the reconstructions with an increasing amount of 1024 by 1024 sized silhouette images starting with one and ending with 40. These measurements have been performed three times for different volume resolutions indicated by different colors.

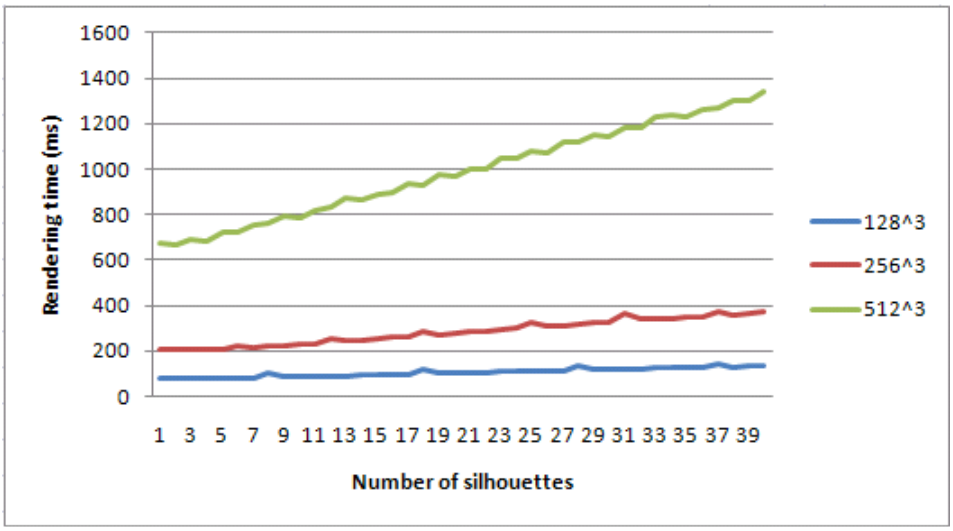

Figure 11. Reconstruction time for different volume sizes all using a silhouette size of 1024.

It is clearly shown that the rendering time increases when the reconstruction volume resolution increases and that the difference is bigger with larger volume resolutions. It is also clear that by increasing the used number of silhouette images the rendering time is increased. When going from one silhouette to 40 silhouettes the rendering time doubles regardless of what volume resolution is used.

Figure 11 also shows that the 3D reconstruction algorithm could be used for real-time or near real-time applications if using volume resolutions of maximum 256^3 without a need to use very expensive hardware.

Figure 12 illustrates the rendering time in milliseconds while creating reconstructions with a resolution of 512^3 with an increasing number of silhouettes starting with one and ending with 40. This is done using four different texture resolutions indicated with different colors.

Larger texture resolutions increases time it takes to reconstruct an object but the difference is small.

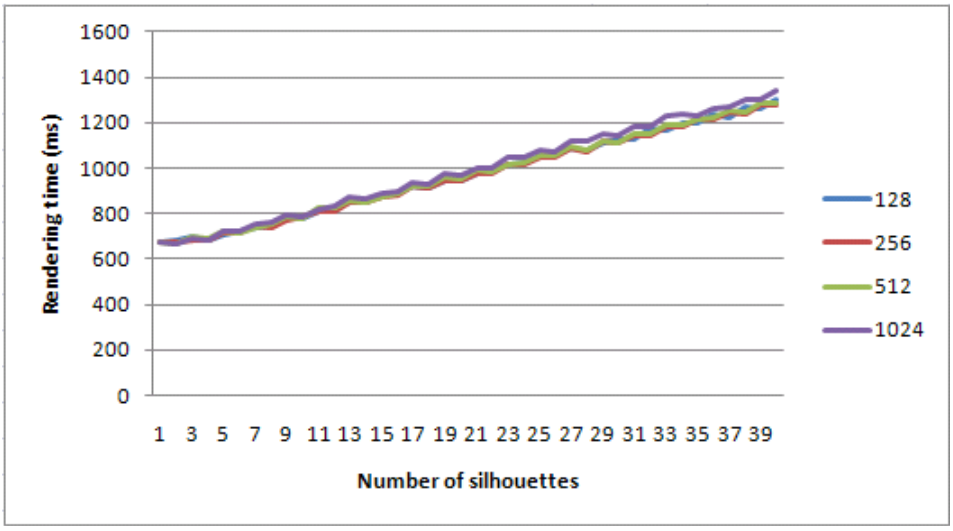

Figure 12. Reconstruction time for different silhouette sizes using a volume size of 512^3.

4.3 Volumetric accuracy

The accuracy of the reconstruction algorithm has been evaluated by the volumetric accuracy and the visual accuracy. Of which the volumetric accuracy results will be presented first.

The volumetric accuracy has been measured by counting the occupied voxels in the volumetric reconstruction made from the plane model from Figure 7. Measurements are made on multiple reconstructions using an increasingly amount of silhouette images starting at one and ending at 40 silhouette images.

Figure 13 illustrates the percentage of occupied voxels in a reconstructed volume with resolution 512^3 while using increasing number of silhouette images. These measurements are taken while using four different texture sizes indicated by different colors.

In the diagram it is shown that there is a rapid improvement in the volumetric accuracy when increasing the number of silhouettes from one to five or even ten silhouette images. After that rapid improvement adding more silhouettes doesn't help the volumetric accuracy.

The diagram also shows that only a small increase in volumetric accuracy can be achieved by using higher resolution on the silhouette images.

Figure 14 illustrates the percentage of occupied voxels in three reconstructed volumes of varying resolutions while using an increasing number of silhouette images with a resolution of 1024 by 1024. The diagram shows that the percentage of occupied voxels is close to constant when using different volume resolutions as long as the same silhouette images are used.

The volumetric accuracy is mostly depending on the amount of silhouette images and it have been shown that five to ten silhouette images is needed to get a high volumetric accuracy. After those five to ten silhouette images adding more only improve the volumetric accuracy by between zero and one percentage.

The volumetric accuracy is never going to be perfect, especially if the object being reconstructed have concave parts. This corresponds to the definitions of earlier mentioned visual hulls.

To get a better estimate of the algorithms accuracy the visual accuracy will now be presented.

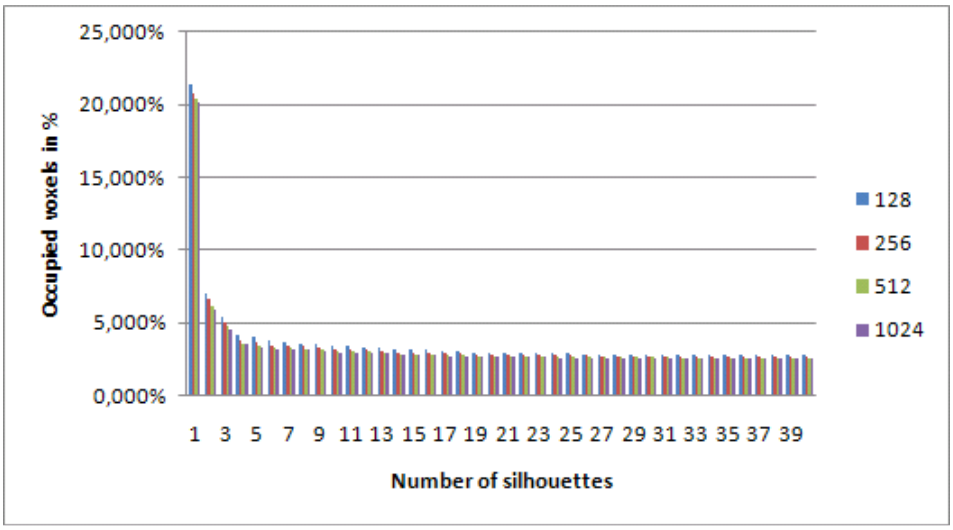

Figure 13. Percentage of occupied voxels using a volume size of 512^3 *using different silhouette sizes.*

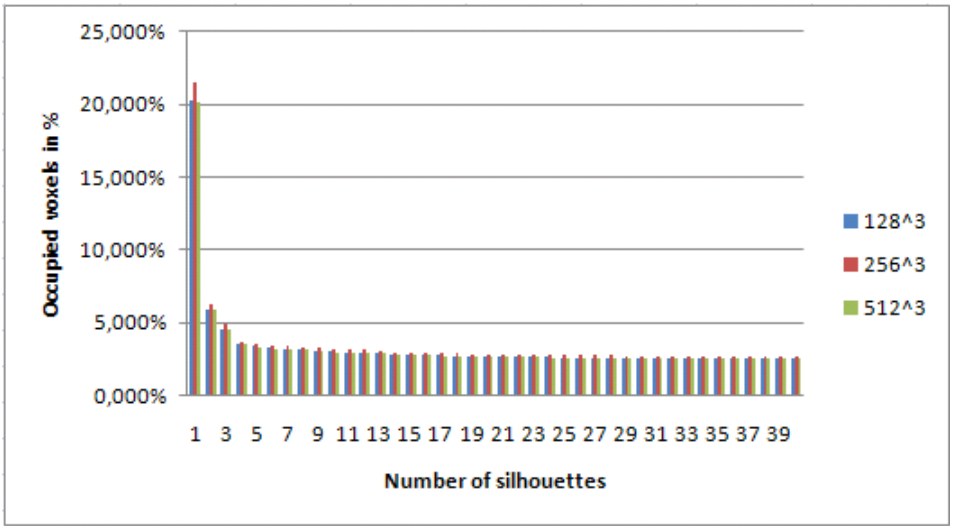

Figure 14. Percentage of occupied voxels using different volume sizes and silhouette size of 1024.

4.4 Visual accuracy

The visual accuracy of the 3D reconstructions are very important, if an object and its reconstruction does not look alike then there was no point in making the reconstructions.

Figure 15 contains four reconstructions of the plane model all created using the same 40 silhouette images. The model on the top to the left is created using a volume resolution of 128^3 and a texture resolution of 128^2. The model on the top to the right uses a 128^3 volume and textures with a resolution of 1024^2. Both of the two models on the bottom uses a volume resolution of 512^3 but the left one uses textures with a resolution of 128^2 and the right one uses textures with a resolution of 1024^2.

In other words the plane on top to the left is reconstructed using the lowest volume and texture resolutions used in these tests and the plane on the bottom to the right is reconstructed using the highest volume and silhouette resolutions used in the tests. The visual improvement when using greater silhouette resolution is best shown in the bottom two cases in the wing section. The figure shows that large silhouette resolutions make the most difference when also the volume resolution is large.

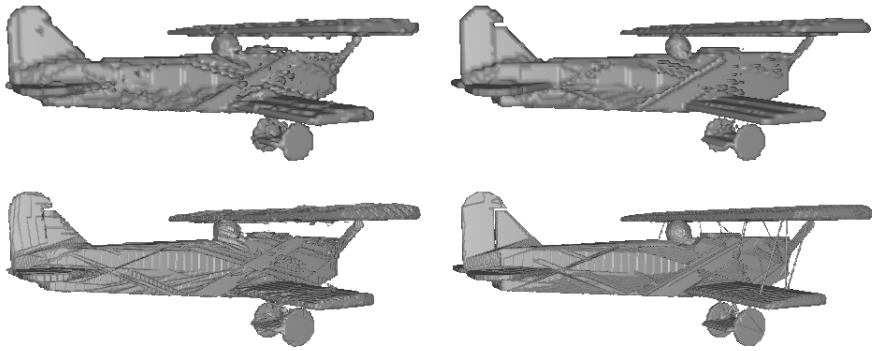

Figure 15. Reconstruction results using different volume and texture resolution. As the volume and silhouettes resolutions increase the accuracy of the reconstruction also increases. ***Upper left:*** *Volume resolution* 128^3 *and texture resolution* 128^2. ***Upper right:*** *Volume resolution* 128^3 *and texture resolution* 1024^2. ***Bottom left:*** *Volume resolution* 512^3 *and texture resolution* 128^2. ***Bottom right:*** *Volume resolution* 512^3 *and texture resolution* 1024^2.

Figure 16 illustrates the visual differences of using different numbers of silhouette images but using the same volume and texture resolutions. The two planes on the top are reconstructed using 5 and 15 silhouettes and the planes on the bottom uses 25 and 40 silhouettes.

The biggest visual improvements are shown in the red circles and are especially visual between the two reconstructions on the top. This corresponds to Figure 13 where it was shown that the biggest accuracy improvement happens before ten or more silhouette images are used. The bottom two reconstructions using 25 and 40 silhouettes is almost identical, there is some improvement at the plane rudders but nothing else clearly visible.

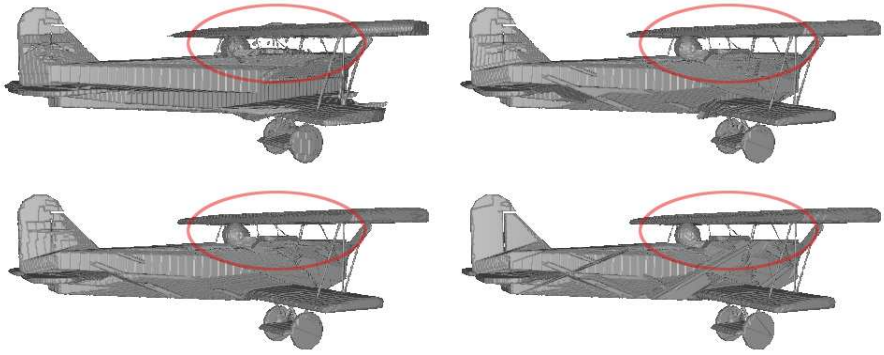

Figure 16. Reconstruction of object using a volume size of 512^3 *and silhouette sizes of 1024 using different number of silhouettes. As the number of silhouettes increase the artifacts decreases.* ***Top left:*** *Reconstructed using 5 silhouettes.* ***Top right:*** *Reconstructed using 15 silhouettes.* ***Bottom left:*** *Reconstructed using 25 silhouettes.* ***Bottom right:*** *Reconstructed using 40 silhouettes.*

4.4.1 Silhouette comparing

To evaluate the visual accuracy silhouette images taken from the original model and the reconstruction is compared.

Figure 17 illustrates the difference between a silhouette image taken from the original model and silhouette images taken from reconstructions created with an increasing number of silhouettes. The comparison is done as described in equation 3 and a lower number is better and zero indicates that the reconstruction is equal to the original image. The measurements are done for three different volumetric resolutions of reconstructions made from silhouette images with a resolution of 1024 by 1024.

The diagram shows that the visual accuracy is increased in the same manner as seen earlier when the number of silhouettes increases. After ten silhouettes have been used the improvement in accuracy gets smaller when adding more silhouettes. It is also clear that by using a higher volume resolution on the reconstructions the degree of error decreases. Figure 17 shows that the degree of visual error is half as great when using a volume resolution of 512^3 rather then 128^3.

Figure 18 contains the difference between the silhouette image taken from the original model and the silhouette images taken from two reconstructions created using five and 40 silhouette images. It is clear that there is some improvement when using 40 silhouettes.

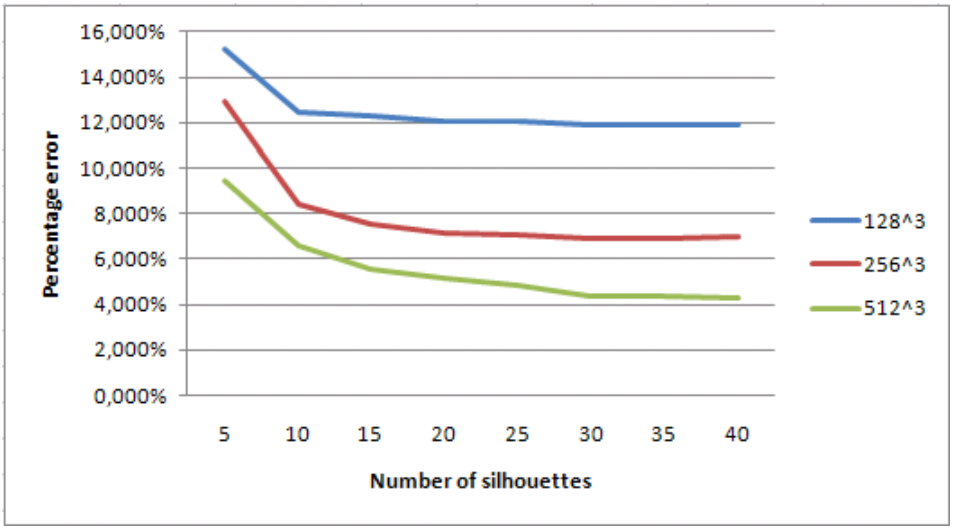

Figure 17. Percentage of error pixels when comparing silhouette image from plane object with silhouette images from reconstructed objects created using increasing number of silhouettes.

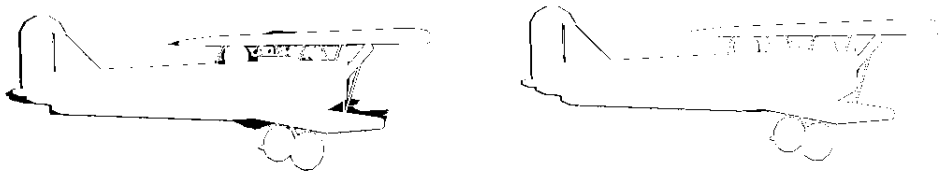

Figure 18. The difference between silhouettes from the original model and the reconstructed model taken from a unused viewpoint. ***Left:*** *Using five silhouette images.* ***Right:*** *Using 40 silhouette images.*

4.4.2 Shaded comparing

When evaluating the visual accuracy comparison is not only done on silhouette images but also images of the models shaded with the Phong illumination model.

Figure 19 illustrates the difference between a shaded image taken from the original model and a shaded image taken from reconstructions created with an increasing number of silhouettes. The comparison is done as when comparing the silhouette images and as described in equation 3. The measurements are done for three different volumetric resolutions of reconstructions made from silhouette images with a resolution of 1024 by 1024.

In contrast to what was seen in Figure 17 where the error was low even when using a low number of silhouettes Figure 19 shows a high error even when using a large number of silhouettes images. Figure 19 do show the same improvements as in Figure 17 but the error never gets close to zero.

Figure 20 illustrates the difference between a shaded image of the original model and shaded images of reconstructions using five and 40 silhouette images. Even here it is clear that the error is high even when using a large number of silhouette images.

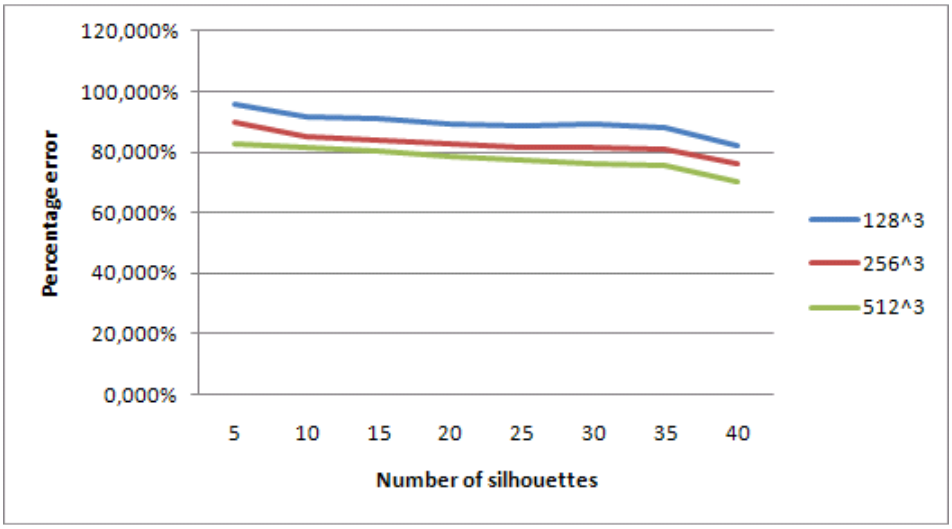

Figure 19. Percentage of error pixels when comparing shaded image from plane object with shaded images from reconstructed objects created using increasing number of silhouettes.

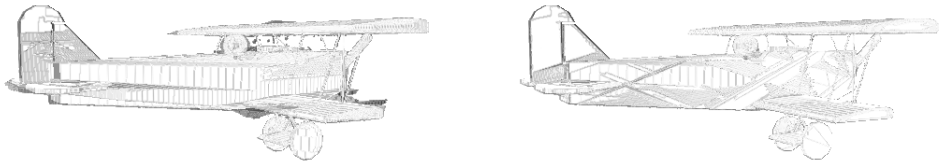

Figure 20. The difference between shaded images from the original model and the reconstructed model taken from a unused viewpoint. ***Left:*** *Using five silhouette images.* ***Right:*** *Using 40 silhouette images.*

5 Discussion

A working 3D reconstruction algorithm has been implemented. The reconstructions have been evaluated in regards of reconstruction time, model accuracy and visual accuracy.

When evaluating the reconstruction speed only the actual reconstruction processes was measured and not the complete process including silhouette extraction and the rendering of the reconstructions. This limits predictions of the processes capability of running at real times but it still gives an indication of what it is capable of doing. It should also be noted that the observed increase in rendering time for larger resolutions is nothing unexpected. It is clear that the rendering time should change when using larger volumetric resolutions, texture resolutions and when using larger number of silhouettes in the reconstruction.

The evaluation of the accuracy has been done in two steps. First the volumetric accuracy was evaluated by comparing the percent of occupied voxels in reconstructions created using different texture and volumetric resolutions as well as different number of silhouette images.

When evaluating the algorithms visual accuracy the reconstructions were rendered using a simple implementation of the marching cubes algorithm [16]. The implemented marching cubes algorithm was only capable of treating the reconstructed volumes voxels as completely inside or outside of the object, leading to an angular surface when converting the volumetric reconstructions to polygonal meshes.

The reason for this simplification is that when recreating the objects only the areas completely covered by all the silhouette images are considered as belonging to the object. This could have been extended to the considering of areas only covered by some of the silhouettes. This in turn could have made the surface smoother and more accurate.

Another improvement to the marching cubes algorithm would be to combine it with texture mapping as used in [5, 9]. This method uses the silhouette images and places then on the surface of the reconstructed object to give it the same look as the original object.

An alternative to the usage of the marching cubes algorithm that could increase the accuracy would be the usage of a volume rendering technique such as proposed by Krüger and Westermann in [18]. They propose a technique where rays are cast through from each of the screens pixels and through the volume while sampling the data at regular intervals. This method could also be run on the GPU making it a perfect match to a hardware accelerated reconstruction algorithm.

The results regarding the volumetric and visual accuracy gained in this research are not theoretically proven. The presented results are the results of empirical studies done with data created for these studies. If using different input data the results could be completely different.

The input data used in the performed evaluations have consisted of 40 silhouette images taken of the objects from simple random positions spread around the objects. If for example all of the silhouettes would show the objects from almost the same viewpoint the accuracy could be much lower than showed in this evaluation.

6 Conclusion

A silhouette based 3D reconstruction algorithm capable of real-time or near real-time usage has been implemented. In addition to the reconstruction time the algorithms accuracy has been evaluated in both volumetric accuracy and visual accuracy.

It is clear that by using larger resolutions on both the reconstructed volume and the silhouette images the algorithm produces higher quality results. It has also been observed that the increase in accuracy inherit from increasing the number of silhouette images is rapidly decreased between the usage of one to ten silhouette images. When using more than ten silhouette images the increase in accuracy is only visible in smaller details.

There is room for improvement and future work could be done using an improved rendering method for the reconstructions. As mentioned before improvements could be done by complementing the marching cubes algorithm with texture mapping or replacing it completely and instead use a volumetric rendering algorithm.

References

[1] R. Szeliski, “Rapid Octree Construction from Image Sequences”, *CVGIP-Image Understanding*, vol. 58, no. 1, pp. 23-32, Jul. 1993.

[2] A. Ladikos, S. Benhimane and N. Navab, “Efficient Visual Hull Computation for Real-Time 3D Reconstruction using CUDA”, in *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW '08. IEEE Computer Society Conference on*, pp. 1-8, 23-28 Jun. 2008.

[3] A. Laurentini, “The Visual Hull Concept for Silhouette-Based Image Understanding”, *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 16, no. 2, pp. 150-162, Feb. 1994.

[4] H. Kim, R. Sakamoto, I. Kithara, T. Toriyama and K. Kogure, ”Compensated Visual Hull with GPU-Based Optimization”, *Advances in multimedia Information Processing – PCM 2008*, vol. 5353, pp. 573-582, 2008.

[5] T. Matsuyama, X. Wu, T. Takai and T. Wada, “Real-Time Dynamic 3D Object Shape Reconstruction and High-Fidelity Texture Mapping for 3D Video”, *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 14, no. 3, pp. 357-369, Mar. 2004.

[6] C. Liang and K. K. Wong, “3D reconstruction using silhouettes from unordered viewpoints”, *Image and Vision Computing*, vol. 28, no. 4, pp. 579-589, Apr. 2010.

[7] M. Li, M. Magnor and H. P. Seidel, “Improved Hardware-Accelerated Visual Hull Rendering”, *Vision, Modeling, and Visualization 2003*, pp. 151-158, Nov. 2003.

[8] M. Li, M. Magnor and H. P. Seidel, “Hardware-Accelerated Visual Hull Reconstruction and Rendering”, *Graphics Interface 2003, Proceeding*, pp. 65-71, 2003.

[9] M. Li, M. Magnor and H.P. Seidel, “A Hybrid Hardware-Accelerated Algorithm for High Quality Rendering of Visual Hulls”, *Graphics Interface 2004, Proceedings*, pp. 41-48, 2004.

[10] A. Laurentini, “How Far 3D Shapes Can Be Understood from 2D Silhouettes”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 2, pp. 188-195, Feb. 1995.

[11] A. Laurentini, “How Many 2D Silhouettes Does It Take to Reconstruct a 3D Object?”, *Computer Vision and Image Understanding*, vol. 67, no. 2, pp. 81-87, Jul. 1997.

[12] E. Angel, “The Phong Lighting Model”, in *Interactive Computer Graphics: A Top-Down Approach Using OpenGL*, vol. 5, Pearson Education, pp. 289-328, 2008.

[13] M. Segal, C. Korobkin, R. Widenfelt, J. Foran and P. Haeberli, “Fast Shadows and Lighting Effects Using Texture Mapping”, *ACM SIGGRAPH Computer Graphics*, vol. 26, no. 2, pp. 341-349, Jul. 1992.

[14] L. Weng, D. Daman and M. Rahim, “Shadow Casting with Stencil Buffer for Real-Time Rendering”, *International Arab Journal of Information Technology*, vol. 5, no. 4, pp. 358-366, Oct. 2008.

[15] H.C. Batagelo and I. Costa Jr., “Real-Time Shadow Generation Using BSP Trees and Stencil Buffers”, in *Computer Graphics and Image Processing, 1999. Proceedings. XII Brazilian Symposium on*, pp. 93-102, Oct. 1999.

[16] W. E. Lorensen and H. E. Cline, “Marching Cubes: A High Resolution 3D Surface Construction Algorithm”, *Computer Graphics*, vol. 21, no. 4, pp. 163-169, Jul. 1987.

[17] F. Gong and X. Zhao, “Three-Dimensional Reconstruction of Medical Image Based on Improved Marching Cubes Algorithm”, in *Machine Vision and Human-Machine Interface (MVHI), 2010 International Conference on*, pp. 608-611, Apr. 2010.

[18] J. Krüger and R. Westermann, “Acceleration Techniques for GPU-based Volume Rendering”, in *Visualization, 2003. VIS 2004. IEEE*, pp. 287-292, Oct. 2003.