DCU | Dublin City University
School of Electronic Engineering

Generating 3D perspectives of textured volumetric scenes.

Student Name: Michael Carmody

Student ID: 98647636

Programme: Meng in Electronic Systems

Acknowledgements

First, I wish to record my sincere gratitude to Prof. Paul Whelan and John Mallon, who supervised the project, and whose dedication, commitment and enthusiasm were an encouragement to a successful completion.

This thesis grew out of a series of dialogues with John Mallon, whose valuable advice and guidance contributed very substantially to the realization of the educational aims of the project. Through his Socratic questioning John brought me closer to the reality I had initially perceived, eventually enabling me to grasp its rich complexity.

Thanks are due also to several researchers and staff at Dublin City University that I visited in the initial stages of the study: Conor Maguire for his set up tips, Billy Roarty for modeling, Robert Sadlier on his advice when using the DTI monitor.

I remain indebted to my parents, family members and friends for providing me the means to learn and understand. I thank them for their understanding, endless patience and encouragement when it was most required.

Declaration

I hereby declare that the work described in this report has been done entirely by me, that any computer programs used to obtain the results presented here were written by me, and that the text of the report is entirely my own work. I have not allowed others to make copies of this report or of any of its contents, nor has it been previously submitted for assessment. Any material contained herein which is based on previous work, such as program subroutines, is clearly identified as such in the report, with references to the source of the material.

Signed: ______________________________ Date: - 19th August 2005

Abstract

This report details the generation of multiple images of a textured volumetric data set for visualization in three dimensions. The issues that are addressed are the basic mathematics required to generate synthetic views and the problems posed as a result of generating these synthetic views.

The problems that are addressed are that of occlusion, where the synthetic images are not geometrically valid, and re-sampling where not all pixel values get assigned texture. Specifically, occlusions arise where one object in 3D blocking another object from view. This results in the absence of expected features and the generation of unexpected features an image, that have to be addressed. The re-sampling problem arises where one pixel gets assigned more than one texture value, where the last value is kept, and another pixel getting assigned no value which results in the generation of a hole.

A number of possible ways were considered in solving the occlusion and re-sampling problem. Consideration was given to the data organisation of the volumetric dataset. Also examined was the utilisation of distances to the synthetic camera. The latter method is used. Different image filters were used to try solving the re-sampling problem but were unsuccessful in obtaining a satisfactory result. A modified filter was designed which it was hoped would resolve the problem but that too failed to give sufficiently good results. This filter ignored the holes in the image. It worked on the basis of a low-pass filter. A scale space approach was finally adopted. This is where multiple scale images are combined to fill holes.

The images are tested on a 3D DTI (Dimension Technologies Inc) monitor which resulted in good 3D perception as expected. Some additional experiments are carried out on the location of the two synthetic cameras, regarding their influence on the perceived scene in 3D.

Table of Contents

Chapter 1

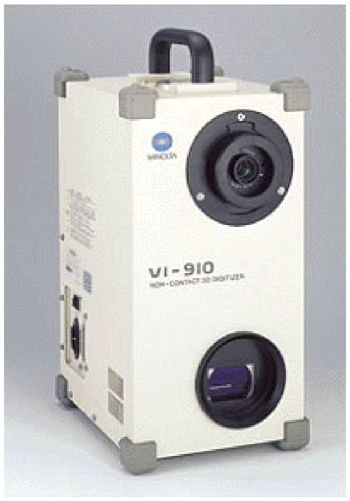
INTRODUCTION

1.1 Scope of the thesis

The objective of this report is to generate multiple images of a textured volumetric data set for visualization of a scene in 3D.

1.2 Acquiring the Data

Reconstructing high detail 3D scenes and object models is a process that consists of several independent, but highly interrelated steps. The first issue was about acquiring 3D data from the desired 3D scene. The data was obtained from a textured digitalization scene. The 3D data is the metric information that is returned from a laser range scanner representing the distance from the source of the laser to the nearest object along a straight line. Textured information comes in the form of intensity. Intensity data is the reflectance of an object in a scene received by a passive sensor such as a CCD array camera. Intensity images are dependent upon the location and direction of various kinds of light sources. From here we had the registration process where the relationship between different sets of data is defined. Finally the registered sets of data are integrated to form a final scene in 3D. The Konica Minolta VI-910 laser was used to acquire this textured data of a scene.



Laser scanners provide an accurate way of gathering data for textured 3D computer models of objects. The Konica Minolta VI-910 laser scanner provides an accurate method of capturing information about an object's surface. This scanner has accuracy in 3D of under a millimeter. It is also able to capture colour texture information for a model by way of the Charge Coupled Device (CCD).

Figure 1.1 Konica Minolta VI-910

The VI-910 works on the basis of laser triangulation. The objects are scanned by a plane of light. The plane of light is swept across the field of view by a mirror and rotated by a precise galvanometer. The laser light gets reflected from the scanned objects [1]. It works on the basis of the laser time of flight. It can capture 3D geometry and colour image using laser technology.

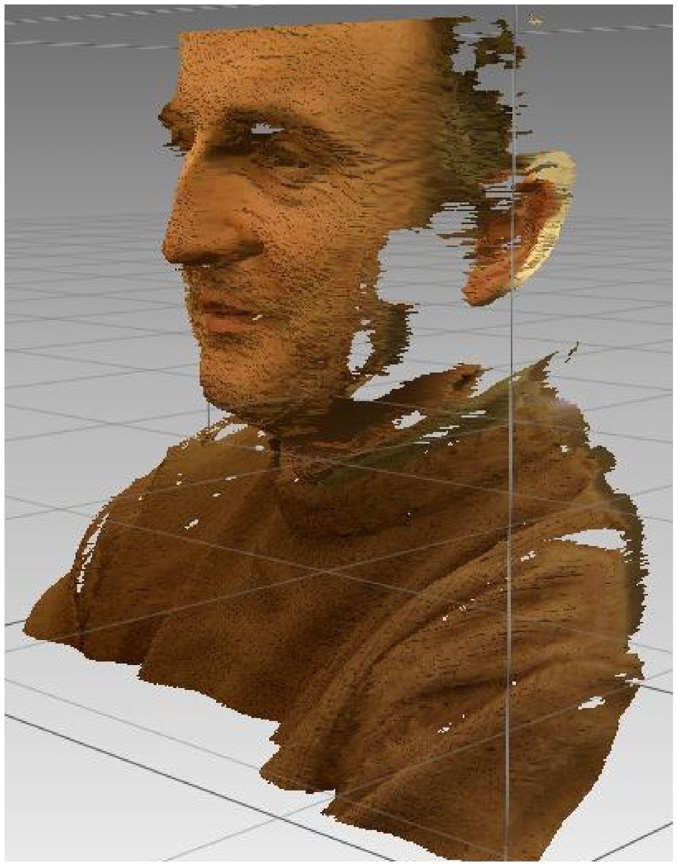


Figure 1.2 A 3D image of Billy

Fig. 1.2 is a scene of Billy Roarty, a member of the technical staff in Dublin City University (DCU), in a screenshot in 3D on rapidform 2004. This tool allowed the visualisation in 2D of real world data from the VI-910 into high quality, accurate and useful data. The data points from this image maybe stored in an ASCII file and used throughout the thesis to represent Billy data set. With this information a 3D visualization environment was designed.

Another image in Fig 1.3 shows an image of Michael in the vision laboratory in DCU. This image was viewed in Rapidform 2004 and the data points from the image were saved in an ASCII file.

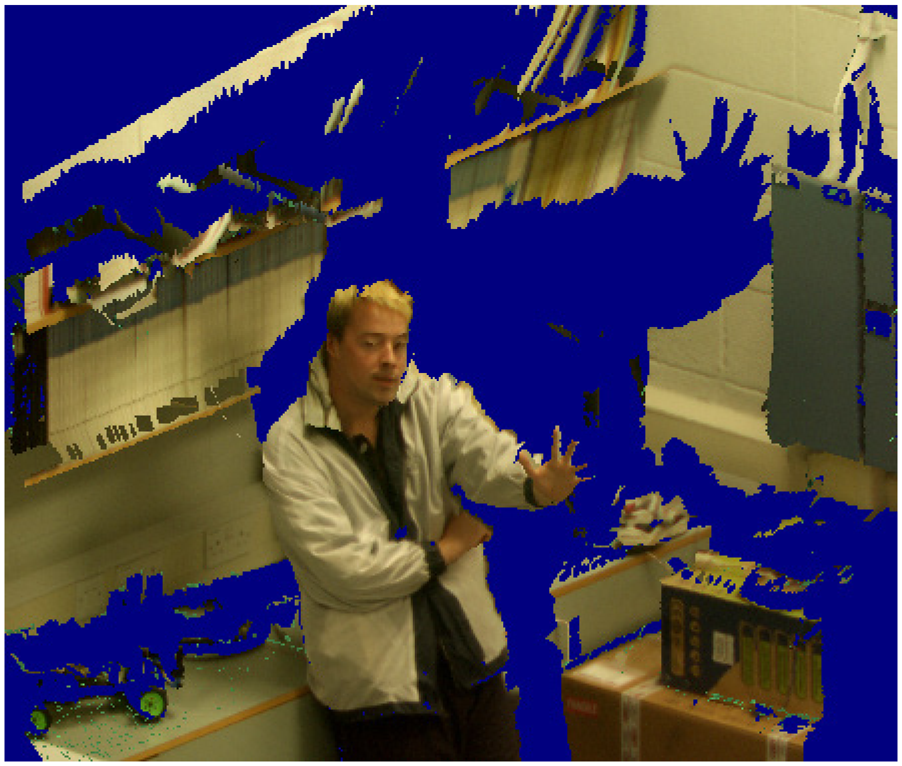


Figure 1.3 A 3-D scene of Michael in the vision lab in DCU

1.3 Review

1.3.1 Occlusion review.

During this experiment two main problems were encountered. The first is that of occlusion. This is the effect of one object in 3D space blocking another object from view. In chapter 2, looking at the images of Billy, one will observe that part of the object face was hidden and instead his shoulder is observed. This is known as object occlusions, which is where part of the object is hidden in both view points. Different camera translations yielded this problem which needed to be fixed. Another type of occlusion that exists is known as shadow occlusion [2]. An example of this would be Michael's hand, in Fig. 1.3, hiding some object behind it. It would not be possible to discover the object that is hidden so another view of the object would be required from a different angle.

Problems that are brought about because of occlusion include the discovery problem and the accessibility problem. The question that arises due to the discovery problem is how the user knows or goes about recovering hidden objects. The second issue that arises is if the user does know there is an object hidden how they go about accessing it. To solve this issue the user will have the view point in some non trivial way in order to retrieve the information encoded in the occluded object. The ability to move the viewpoint helps users to discover more objects [3].

There is also some related work done that deals with object discovery and access in complex 3D environments for robust navigation approaches. The Worlds-in-Miniature technique uses a small 3D map of the image to support both discovery and access [4]. Worldlets provide global overview maps as well as local views optimized for detail [5]. Bird's eye views combine overview and detail views of the world [6]. These deal more with the problem of occlusion in navigation scene rather than with just a single image.

In occlusion the main problem is identifying objects that hide, called occluders, before reconstructing the objects that are hidden known as occludes. Knowing these we can evaluate the consistency of a given voxel by using only the cameras [7] [8].

Two types of occlusions can be encountered in a range image. An occlusion arises either when the reflected laser light does not reach the camera or when the direct laser light does not reach the scene surface.

1) A camera occlusion arises when a part of the illuminated surface in the scene is occluded to the camera by another part of the scene.

2) A light occlusion arises when the direct laser light does not reach a part of the surface in the scene because it is reflected from another part of the scene [9].

This thesis will examine the camera occlusion that has arisen from the surface of a scene that is occluded to the camera by another part of the scene. By varying the viewing location around the Michael dataset one can examine objects that may be covered by his body.

A method described by Sung-Eui Yoon on the Z-Buffer Algorithm is similar to the solution that is used in this report for solving occlusion. It works on the principle of comparing surface depths at each pixel position on the projection plane. The object with the lowest z coordinate value is in front of the other objects, so therefore is displayed. The advantage of this method is its simplicity but it is memory intensive. Depending on the size of the image it requires a large per pixel memory requirement because it needs to store and compare values of individual pixels. [10]. The Z-Buffer offers a deterministic solution in solving the over-sampling problem. Without its use the colour pixel value is based solely on the organisation of the data set.

Other methods that are used for solving occlusion include the Newell's algorithm (Depth-sorting algorithm). This algorithm sorts objects by depth. Once there is an ordering we can paint them in from back to front. But there is the problem of overlapping objects and also forces the system to render every point in the image even if that polygon is occluded in the finished scene. [10].

1.3.2 Texture mapping and re-sampling.

Texture mapping also became a problem. This is the process of adding realism to a computer image. A texture image is mapped to a simpler shape that is generated in the scene. In the case of forward texture mapping the problem of adjacent texels not mapping to adjacent pixels resulted in holes.

Matsushita suggests that the primary reason why holes exist is that there are some invisible polygons due to a hollow or concave surface. He suggests that the holes should be classes into different type's eg (1) isolated holes, (2) a series of connected polygons which contain at least one polygon or (3) a collection of polygons which contain at least one un-textured polygon surrounded by un-textured polygons [11].

Braccini and Marino show that if you deposit the pixels of a texture scanline along the path of a Bresenham digital line, an image can be rotated or sheared. To fill the holes that result between adjacent lines, they draw an extra pixel at each kink in the line. This results in some redundancy [12].

Foley and Van Dam use Bresenham's array to resample an array. The is achievable because distributing m source pixels to n screen pixels is analogous to drawing a line with slope m/n [13].

Marta Fidrich provides a very useful document on Multi-scale analysis of invariants applications to volumetric medical image processing. She uses the scale-space approach to gather appropriate information when sizing an image. [14]

1.3.3 Organisation.

The issues that are addressed are the basic mathematics required to generate synthetic views and the problems posed as a result of generating these synthetic views.

The problem of occlusion is an issue in a scene in 3D. Questions are asked of the user about what objects are required to be viewed. This issue is dealt with in chapter 2. Chapter 3 examines some different methods of trying to solve occlusion. A method based on maintaining a valid depth buffer was used.

The re-sampling problems are dealt with in chapter 4. This is a problem where one pixel gets assigned more than one texture values, where the last value is kept, and another pixel gets assigned no value which results in the generation of a hole. Different image filters were used to but to no avail. A special filter was designed in matlab but, this too was unsuccessful. This filter ignored the zero values and worked on the principle of the low pass filter. The scale space approach was used and it yielded satisfactory results. The results that were gathered from a scale space approach had eliminated the holes in the image.

Various experiments are carried out in chapter 5. The camera parameters were modified and the effects were detailed. These images were displayed on the DTI monitor and the changes are commented on.

Chapter 2

PRELIMINARY PROJECTION

2.1 Mathematical background

A vital part of 3D graphics is mapping a point in 3D space into a 2D representation on a virtual viewing device. Mapping is the last transformation that is applied to the 3D geometric information before it executed to the viewing device. 3D projection is a mathematical transform used to project 3D points onto a 2D plane. This is done to simulate a view point to a scene. This is the first step in a process to represent scenes in 3D to images in 2D. Here the mathematics behind the camera projection is examined.

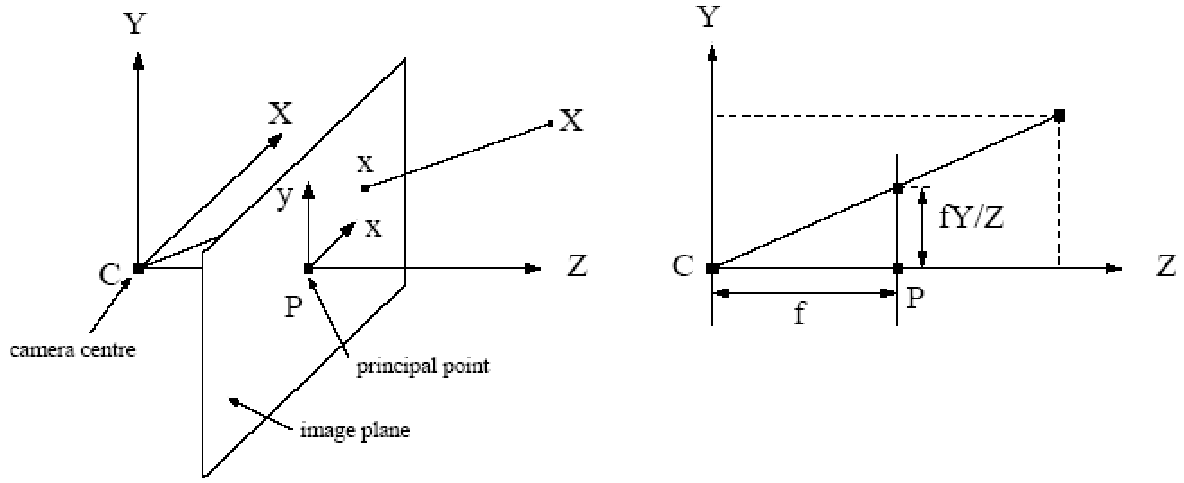


Figure 2.1 The Projective projection of a camera [15]

Textured information about objects is stored in a collection of points. Each of these points has three values which represent its X, Y and Z coordinates from the origin relative to the object they belong to. Each point also has a rotation associated with it which describes its position and orientation relative to a world reference frame. The camera has a set of three X, Y and Z coordinates and three different angles describing the observer's position and the direction along which it is looking.

2.2 Projection of a point

Taking a point of the form $[x, y, z, 1]^T$, a camera transform, P is applied resulting in the point of form $[x', y', z']^T$. The projected point on the screen is then at the 2D coordinates $\left[\frac{x}{z}, \frac{y}{z}, 1\right]^T$.

$$\Rightarrow P\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} x'/z' \\ y'/z' \\ 1 \end{bmatrix}$$

Where P = KRT being the camera projection matrix. The K is the internal camera matrix specifying the focal lengths and principle points.

$$K = \begin{bmatrix} f_x & 0 & C_x \\ 0 & f_y & C_y \\ 0 & 0 & 1 \end{bmatrix}$$

f_x and f_y are the focal lengths in the x and y directions respectively, and C_x and C_y are the centre offsets in the x and y directions respectively.

The external camera parameters specified by the RT matrix which includes the camera position and orientation.

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = RT\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

RT is the matrix of translation and rotation parameters.

$$RT = \left[\begin{array}{ccc|c} r_1 & r_2 & r_3 & t_x \\ r_4 & r_5 & r_6 & t_y \\ r_7 & r_8 & r_9 & t_z \end{array}\right]$$

r = is a rotation

t = a translation in a given point of space

2.3 Rotation and translation equations

The rotation around the z axis is described below.

$$\text{Rotation around the x-axis} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\text{Rotation around the y-axis} = \begin{bmatrix} \cos\beta & 0 & \sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\text{Rotation around the z-axis} = \begin{bmatrix} \cos\psi & -\sin\psi & 0 & 0 \\ \sin\psi & \cos\psi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The overall matrix consists of the product of the rotations and translation about the x, y and z axes.

$$R = R_x R_y R_z$$

$$T = T_x T_y T_z$$

It is important to keep the structure of multiplying matrices in a set order. Changing them around will change the results. The objects must be rotated before being translated otherwise the position of the object in the world would get rotated around the centre of the world.

Now to set up the camera transform we presume that the camera looks in its z direction, the x direction is to the left and the y direction is upwards.

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = P\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = KRT\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & C_x \\ 0 & f_y & C_y \\ 0 & 0 & 1 \end{bmatrix} * \left[\begin{array}{ccc|c} r_1 & r_2 & r_3 & t_x \\ r_4 & r_5 & r_6 & t_y \\ r_7 & r_8 & r_9 & t_z \end{array}\right] * \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Now the original points have been multiplied by the K matrix. A 2D projection of the scene must be taken. To do this the x and y pixels are normalized to the plane by dividing by their z values.

$$\begin{bmatrix} Pix_x \\ Pix_y \end{bmatrix} = \begin{bmatrix} x'/z' \\ y'/z' \end{bmatrix}$$

2.4 Image registration

The image is taken as $I(u,v)$ and the world point $\mathbf{M} = (x_i, y_i, z_i)^T$.

Where the image and 3D registration is giving according to

$$u = r\left[-w - (S_x \frac{x_i}{z_i} - dx)\right]$$

$$v = r\left[S_y \frac{y_i}{z_i} - dy\right]$$

r = round function value to the nearest integer

w = width of the image

S_x = x direction scaling

S_y = y direction scaling

h = height of the image

where $S_x = \dfrac{(w-2)}{\max\left(\dfrac{x}{z}\right) - \min\left(\dfrac{x}{z}\right)}$

$$S_y = \frac{(h-2)}{\max\left(\dfrac{y}{z}\right) - \min\left(\dfrac{y}{z}\right)}$$

dx is an x direction translation where $dx = \min\left(\dfrac{x}{z}\right) + 1$

dy is a y direction translation where $dy = \min\left(\dfrac{y}{z}\right) + 1$

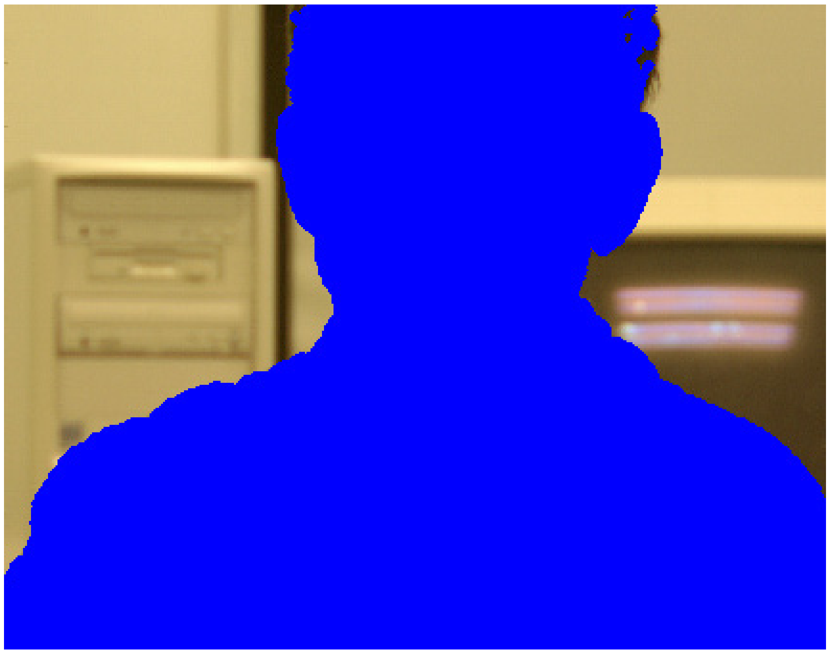


Figure 2.2 Projected points onto Billy's image

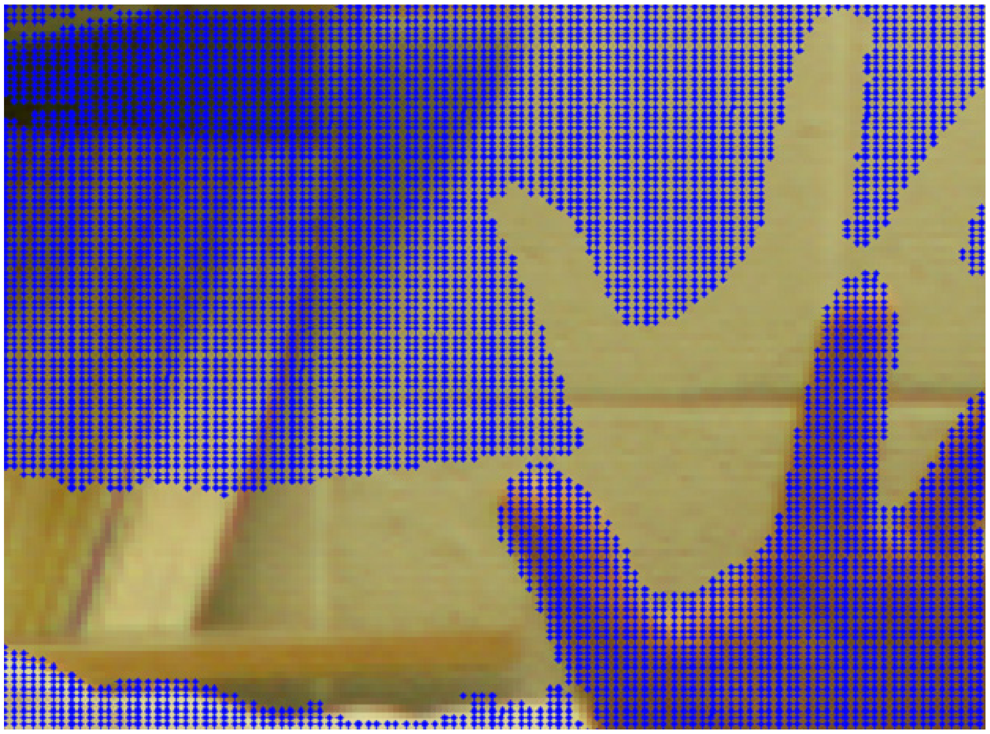


Figure 2.3 Projected points onto Michael's hand

Fig. 2.3 shows the points of projection marked out on Michael's hand. The projected points have been automatically scaled up so that they would match the original image. The data set was obtained from the scanner.

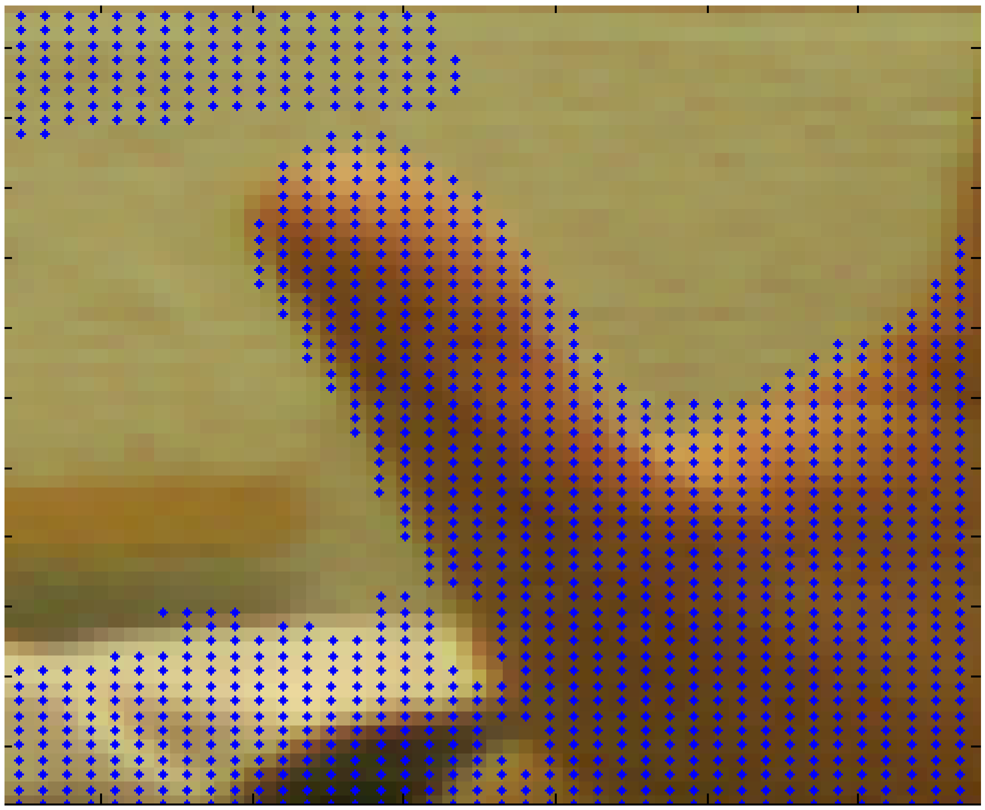


Figure 2.4 Projected points onto zoomed section of Michael's hand

It can be observed, in the zoomed up image of Michael's hand in Fig 2.4, that the points of projection have matched the image accurately.

2.5 Occlusion Problem

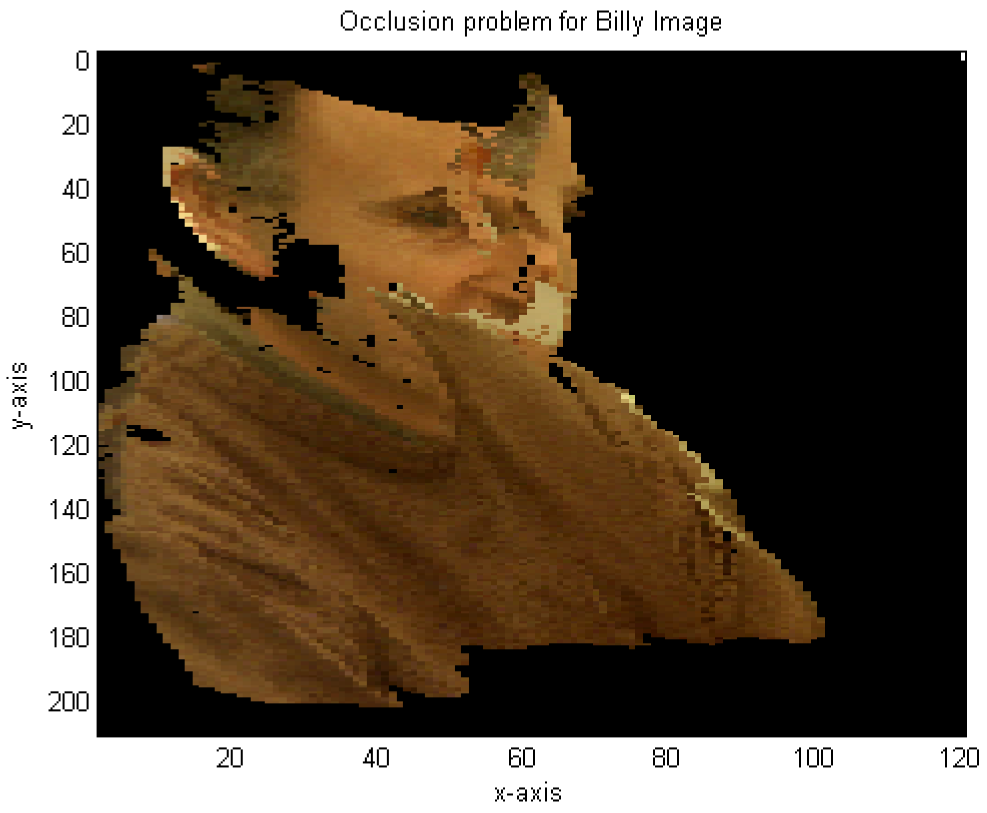


Figure 2.2 Billy with occlusion

Occlusion is the effect of one object in 3D space blocking another object from view. So following the application of the basic projections described in the equations in section 2.2, one can observe in fig. 2.2 that there is a problem with occlusion. Data is scanned in lines and the last texture point fills the image. Billy's left shoulder can be seen where his face should be appearing. This issue needs to be resolved will be examined clearly in the next chapter. At this point Billy is positioned at T = [1200, 1200, 0].

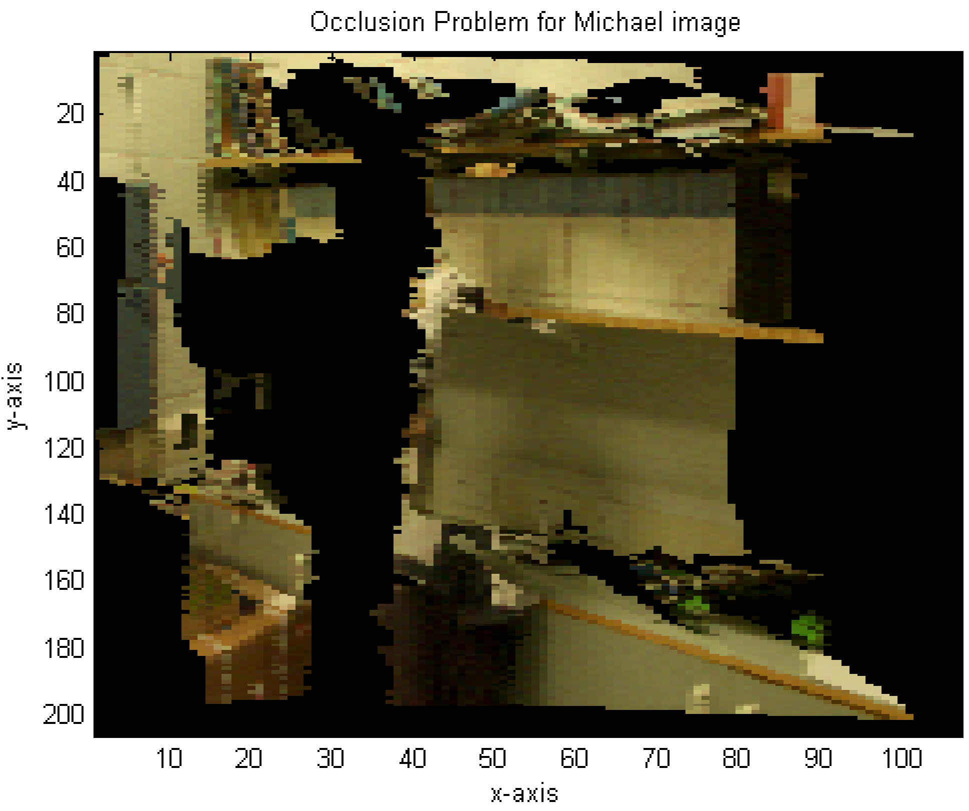


Figure 2.3 Michael with occlusion

Examining figure 2.3 one can clearly see that Michael is not visible in the image. The background has occluded the person. The camera in positioned relative to [-2000 -100 0]. A method similar to the Z-buffer algorithm proposed by Sung-Eui Yoon [10] was used to solve the occlusion problem. The depth algorithm that was designed works on the principle of scanning the data in lines and the last texture point fills the image. This method was chosen because it was efficient, produced good results and didn't require a large memory capacity.

This thesis examines the camera occlusion that has arisen from the surface of a scene that is occluded to the camera by another part of the scene [9]. By varying the transformation around the Michael one can examine objects that may be covered by his body.

Chapter 3

PROJECTION WITHOUT OCCLUSIONS

3.1 Screening Method

As seen in Fig2.2 and Fig2.3 there is a problem with occlusion. In the last Chapter it was shown that the synthetic images are not geometrically valid. Objects that should be appearing in the image were not there and objects that were there that should not be included.

The first option that is introduced to solve this problem is that known as the screening method. This method is based on the ordering of the data sets. The angle is not allowed overlap on itself. As a demonstration, the position of the camera was taken to be at [0 -3]. This type of scan is ideal for scan line viewing.

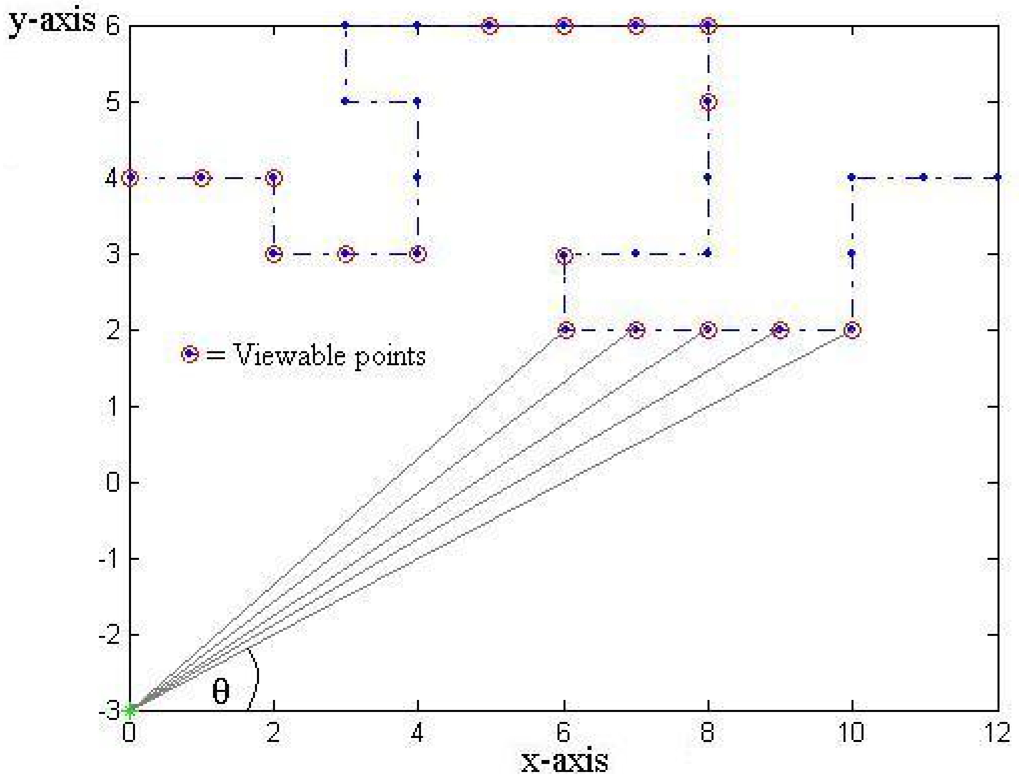


Figure 3.1 Screening of Data

Suppose we have two points at (x_1, y_1) and (x_2, y_2), then the distance between these two points is

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

The slope is calculated as,

$$slope = \frac{y_2 - y_1}{x_2 - x_1}$$

$$\therefore \theta = \arctan * slope$$

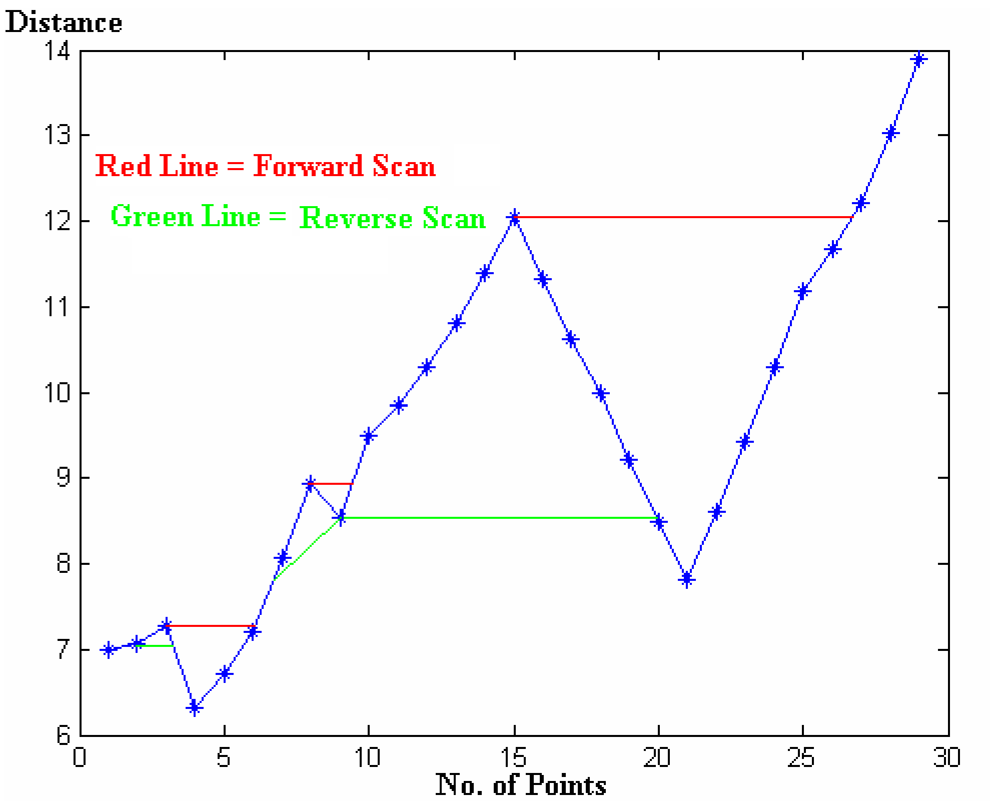


Figure 3.2 Distance from camera

Data is scanned in both forward and reverse directions. The problem with this method is that some data scanned in the forward scan and reverse scan is correct. The conflicting paths were solved by the distance from the camera to the point in view. This method depends on the data being acquired by a line scanning window. It was not used due to its complexity and computably intensive.

3.2 Depth buffer algorithm

The alternative method that is used to solve the problem of occlusion is similar to the Z-Buffer transform. It involves the management of image depth in images. It offers a good solution to the problem of deciding which elements of a scene are drawn in front and which are hidden.

The basic idea around the proposed algorithm to solving occlusion for conflicting pixels involves comparing the depth of a generated pixel on the z-coordinate to the likewise pixels that are on the same z-axis. The distances between the pixel and the camera is got from using the distance formula, $d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$. The values are stored in a buffer. If a second pixel is found on the same z-axis as another pixel the two depths are compared and the one closest to the observer is chosen as the viewable pixel.

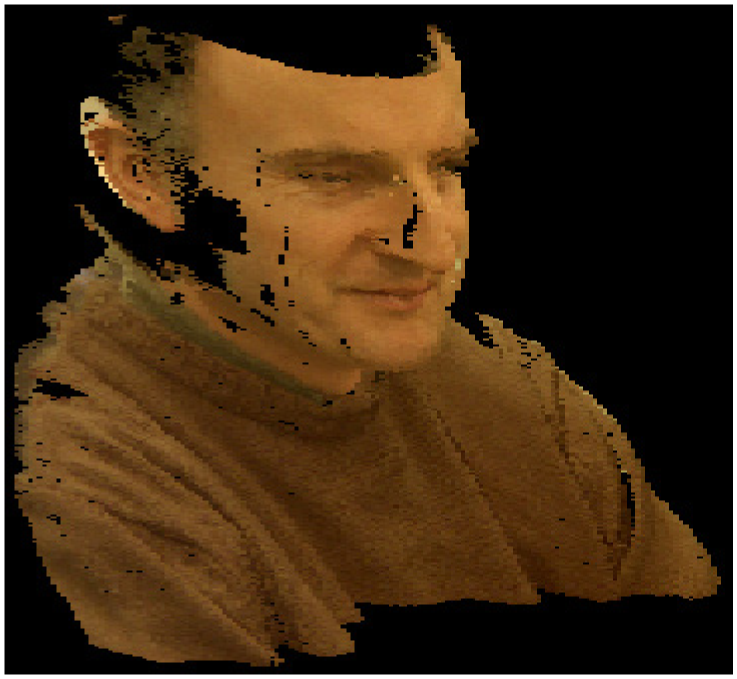


Figure 3.4 Billy without occlusion problem

Figure 3.4 shows that the occlusion problem is fixed by the method proposed on the Z-Buffer algorithm. Examining the image one can observe that Billy's left hand shoulder appears where it should be. It is behind his face and chin. Inspecting this image we can see that if two pixels lied on the same z-axis the nearest pixel was given priority.

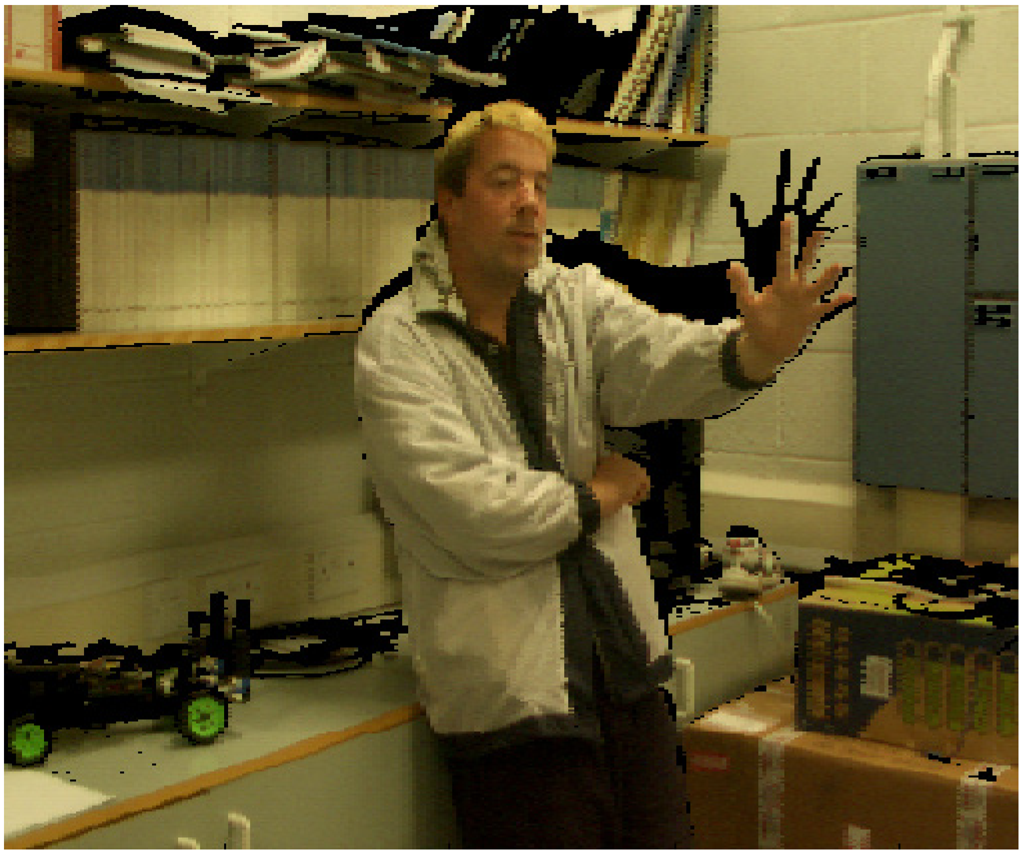


Figure 3.5 Michael without occlusion problem.

Figure 3.5 shows that the occlusion problem is fixed by the method proposed on the depth storage algorithm. Michael's body can be distinguished from the information in the background.

The images that were obtained show clearly that the depth sorting algorithm proved efficient. Michael's body, in Fig. 3.5, is distinguished from the background information. The depth sorting algorithm is easy to implement and it required no sorting of the surface in a scene. A disadvantage of the depth sorting algorithm is that it requires a second z buffer in addition to the frame buffer. Extra memory capacity is required to store the second frame. The speed of the algorithm is limited by pixel redrawing and for fine features it needs high precisions.

Chapter 4

IMAGE GENERATION

4.1 Dealing with re-sampling

Re-sampling an image leads to holes being generated in the image. This is the problem of one pixel gets assigned more than one texture value, where the last value is kept, and another pixel getting assigned no value which results in the generation of a hole. Over-sampling is solved by the depth sorting algorithm. A number of different solutions were carried out to solve the problem re-sampling. The first proposal described in this chapter was that of using image filters. These are described in this chapter and the reasoning why they were unsuccessful is also given. The second method that is proposed was to design a filter using Matlab which ignored the zero values (i.e. black values). Both these methods were unsuccessful.

The third method proposed was that known as the Scale space approach. This is an efficient multi-resolution technique for image structure analysis. Marta Fidrich proposed a scale space approach for registration and recognition of images. This method was followed by in this thesis [14].

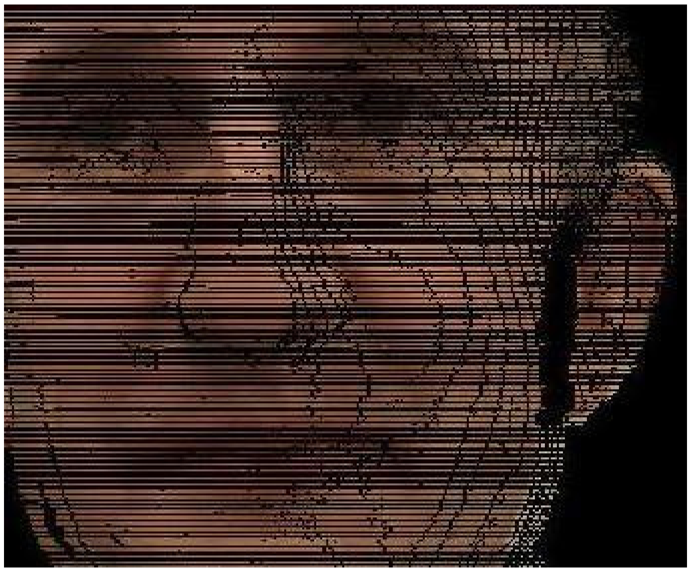


Figure 4.1 Billy with Holes

4.2 Image filtering

4.2.1 Various filtering in Neatvision

A small section of the original Billy image was sampled. Various image filters were used and the results were recorded. Neatvision was used to examine the images in different filters.

NeatVision is a Java based image analysis and software development environment, which provides high level access to a wide range of image processing algorithms through well defined and easy to use graphical interface. [16]

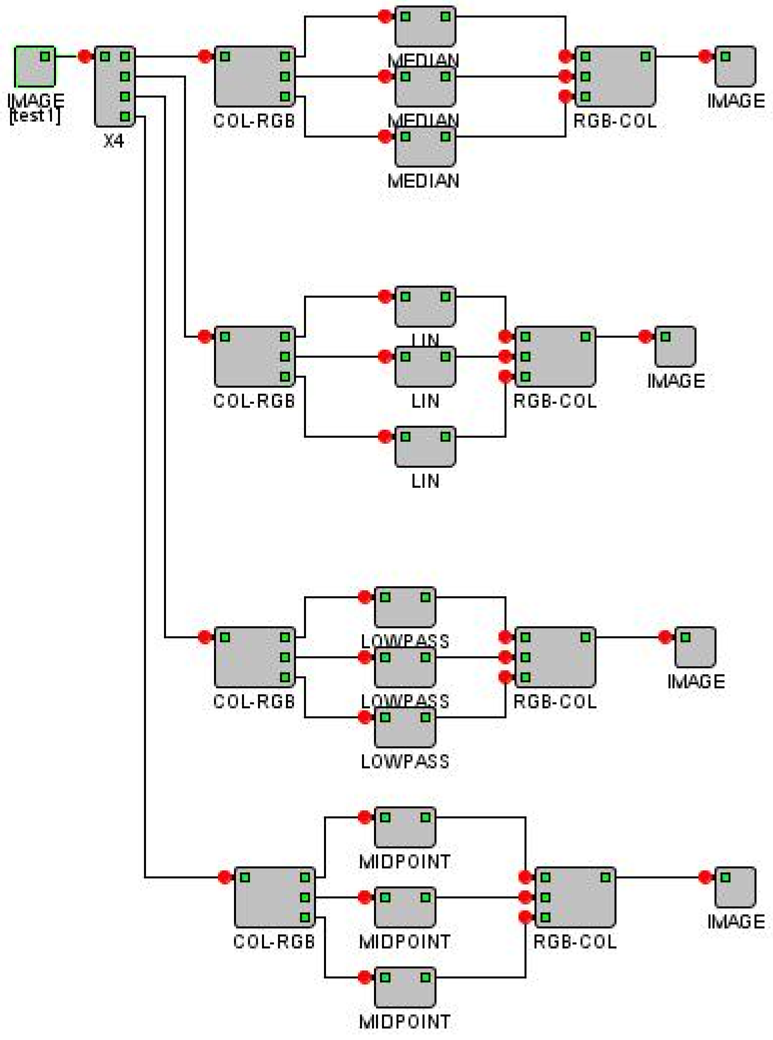


Figure 4.2 Four filters designed in Neatvision

Fig. 4.2 shows four different filters that were used. The median block represents the median filter, LIN represents the largest intensity neighbour, lowpass represents the lowpass filter and the midpoint filter is represented by the midpoint filter. The images were tested and the results were recorded.

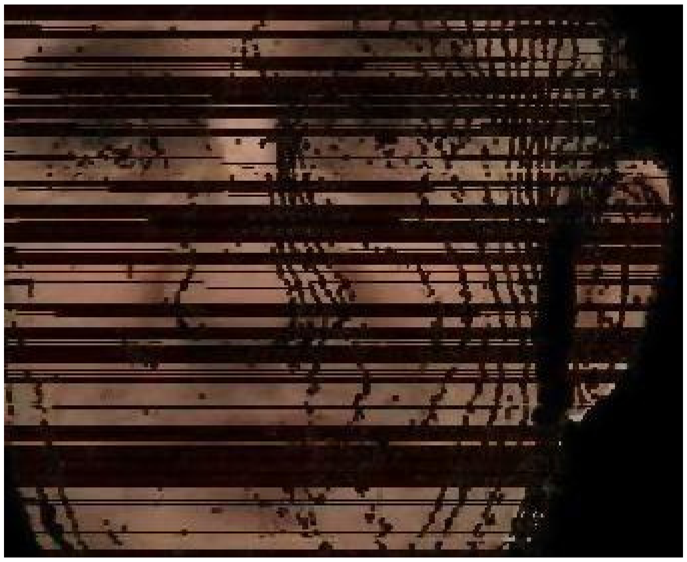


Figure 4.3 Output from the Median Filter

In general a median filter is useful for reducing the level of noise in an image. Median filters consider each pixel in the image in turn and looks at its neighbouring pixels to decide whether or not it is representative of its surroundings. So instead of simply replacing the pixel value with the mean of the neighbouring pixel values, it replaces it with the median of those values. To calculate the median the pixel values from the surrounding neighbourhood are sorted into numerical order and then the pixel being considered is replaced with the middle pixel value.

Comparing Fig. 4.3 with the original image in Fig. 4.1 it can be observed that the median filter degrades the image. This is credited to the fact that there is a larger amount of black pixels in the original image. In every 3 x 3 window if there are at least five black values out then the centre pixel get replaced by a black value. The median filter takes the fifth largest value in a 3 x 3 window and replaces its centre pixel with this value.

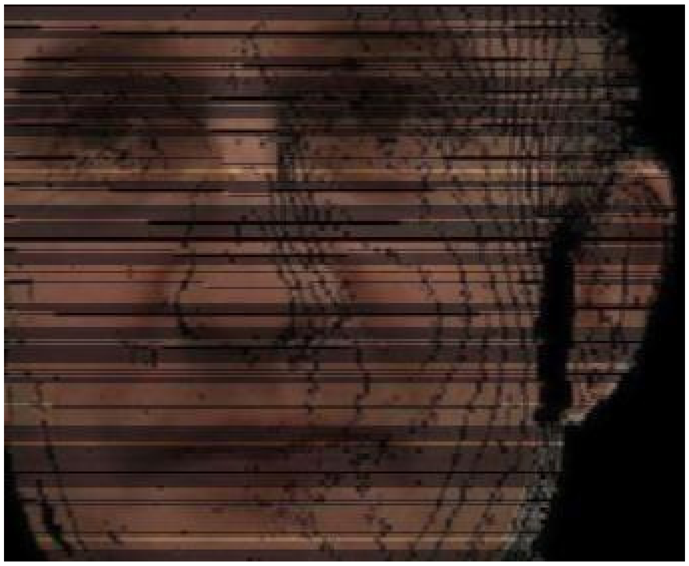


Figure 4.4 Output from the low pass filter

A low pass filter tends to pass low frequencies fairly well and blocks high frequencies. Low pass filtering causes a blurring effect to the image. It is usually used to remove noise from an image.

Comparing Fig. 4.4 with the original image in Fig. 4.1 it can be observed that the low pass filter reduces the intensity of the image. The low pass filter acts like an averaging filter. Since there was a large amount of black holes in the original image the average intensity was low. Though Billy's face seems to be improved when comparing it to the output of a median filter, it would still be difficult to recognize features on his face.

Examining the output of the low pass filter it can be observed that the bright pixels are removed and removed by lower values. There still remains a larger amount of black areas which have the effect of decreasing the overall intensity of the image. But the low pass filter, looking at fig. 4.4, clearly doesn't remove holes.

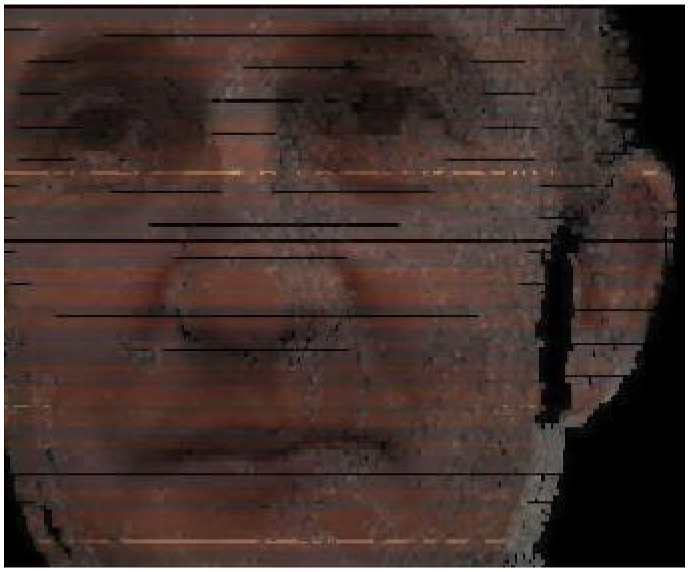


Figure 4.5 The output from the midpoint filter

The midpoint filter finds the average of the minimum and maximum values of values within a window. This leads to a darker image because we have a lot of low pixel values (black streaks). The midpoint filter is generally useful for Gaussian and uniform noise.

Analyzing Fig. 4.5 with the original image in Fig. 4.1 it can be observed that the midpoint filter reduces the intensity of the image. The overall intensity of the image is reduced because there are more black hole intensities than bright intensities. Assessing Fig 4.5 the problem of holes is reduced. More of the pixels on Billy's face have a colour value associated to them. Billy's face is very dark when it is compared with the original image in Fig. 4.1.

Finding detailed features of Billy's face would be difficult because the quality of the output from the midpoint filter is poor.

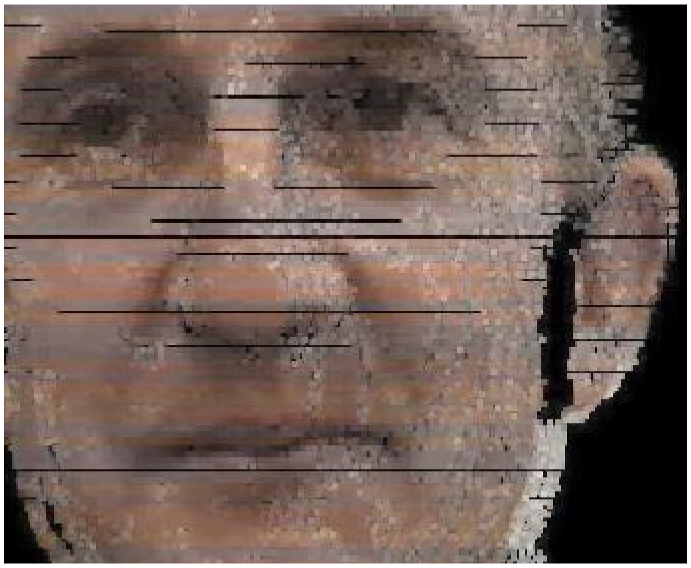


Figure 4.6 Output from the Largest Intensity Neighbour Filter (LIN)

The largest intensity neighbourhood function spreads bright regions and contracts dark ones. This function considers each pixel in the image in turn and takes the pixel with the largest value.

Fig. 4.6 shows that the output of the largest intensity neighbour has fewer black holes that the original image in fig 4.1. Therefore since there are fewer black holes the overall intensity increases. More of the pixels on Billy's face have colour associated with them.

The overall quality of detail in Billy's face is poor. It would be difficult to extract details on certain features of Billy. There is a large quantity of noise associated with the output of the largest intensity neighbour. The largest intensity neighbour is not good for smoothing as it increases the peaks. This is what causes the spotty effect in Fig 4.6.

Investigating the output of the median, low-pass, midpoint and largest intensity neighbour filters it can be observed that there is a problem with hole content, image noise and the overall quality of the images is poor. Good quality feature extraction would be near difficult to obtain. These filters were not going to be useful for the purpose of this thesis.

4.2.2 Modified filter

A modified filter was designed in matlab. This was similar to the low pass filter with the exception that it would ignore the zero values up to a value of twenty. So if the 3 x 3 window had found seven intensities greater than twenty and two less it would add the seven large values, ignore the smaller values, divide by seven and output the result. The value twenty was taken as a cut-off point because it was decided that values below this point would affect the output.

It was hoped that the remaining low frequencies would be passed and the higher values blocked. But again it caused a blurring effect to the image. Low pass filters are good for removing noise in an image. The intensity had improved because the low values below twenty were excluded. It would still be difficult to recognize features on Billy's face because of the blurring effect.

So far in chapter 4, filters in neatvision and one designed in matlab were examined but to no degree of satisfactory success. Each filter had it own degree of poorness with regards to hole content and image noise. This leads to the examination of an approach known as scale space. The basic idea behind this approach is to grow a large image from different smaller scaled images.

4.3 Scale space approach

The scale space approach is an efficient multi-resolution technique for image structure analysis. It describes the image structure at different scales. A main argument behind the construction of scale space is that if there in no prior information available about what are appropriate scales for a given data set, then the only reasonable approach for an uncommitted vision system is to represent the input data at multiple scales.

Below Fig 4.7 shows two images of Billy, that are equally scaled up, with the image on the right having been scaled up by the scale space approach. Fig 4.7(a) is used as a reference frame to build up the image of Fig. 4.7(b).

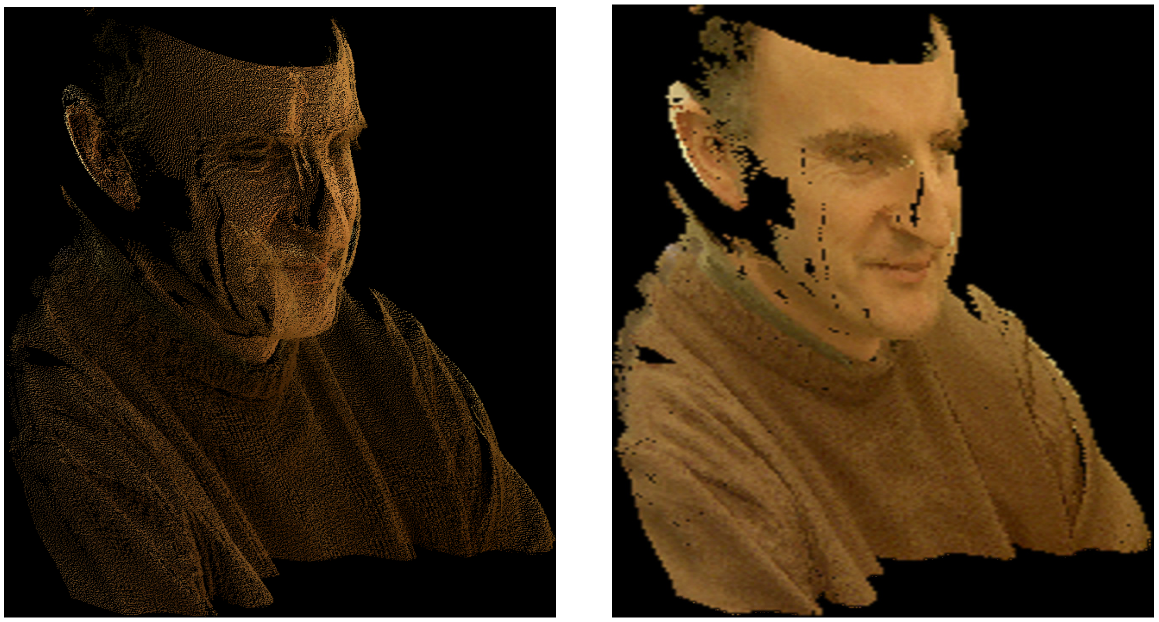


Figure 4.7(a) Image without scale space method (b) Image with scale space method

The scale space approach is relatively easy to implement. The image is of good quality but there does seem to be a slight blurring effect. Edges would not be preserved. This may require having to go back to the original image to locate the edges. This would also be a concern for increasingly complex visual modules. Implementing the scale space approach can be intensive.

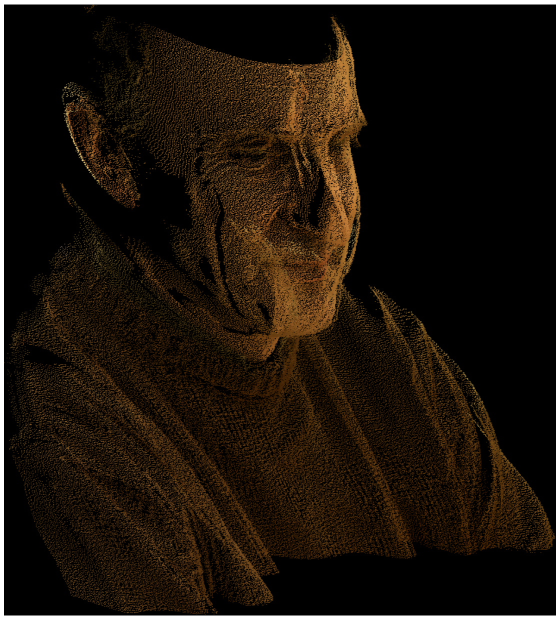


Figure 4.8(a) Image without scale space.

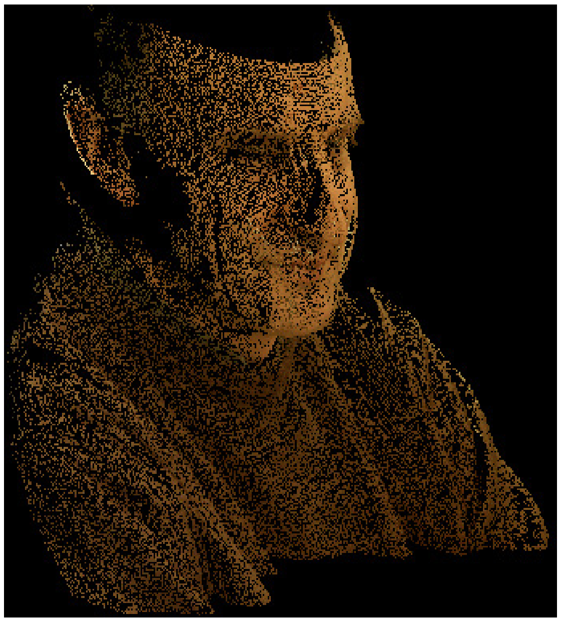


Figure 4.8(b) Image grown from fig 4.8(a)

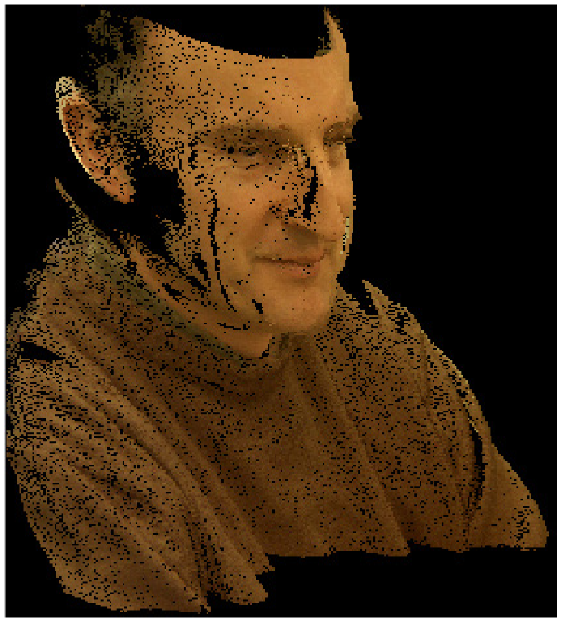


Figure 4.8(c) Image grown from fig 4.8(b)

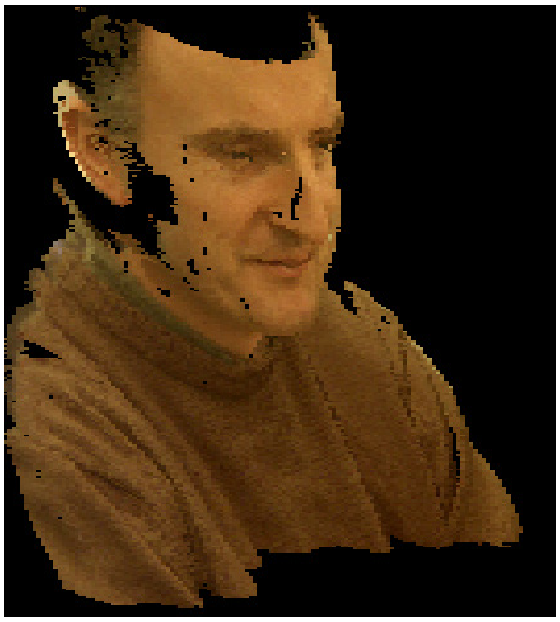


Figure 4.8(d) Final up-scaled image

The final output image of fig 4.8(d) is a combination of multiple resolutions. Fig 4.8(a) is the reference image. The height and width of the images are increased each time as the images are scaled upwards. Fig 4.8(b) starts off at being, in this case, at a quarter the size of the final image, Fig. 4.8(c) is double the size of Fig (b).

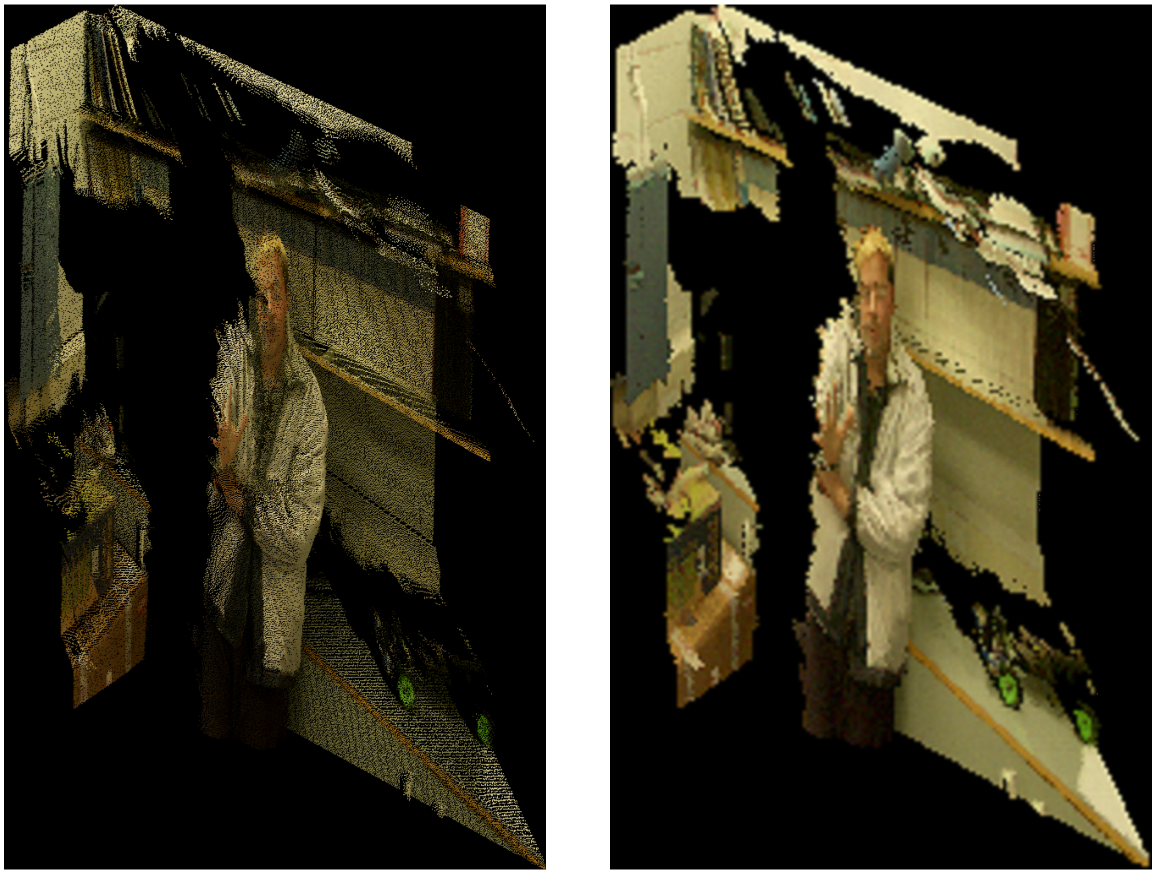


Figure 4.9(a) Image without scale space method (b) Image with scale space method

Fig 4.9(b) consists of a combination of multiple resolutions of up-scaled images. Examining this image it can be observed the scale space approach introduces a blurring effect. Edges are poorly preserved. Features such as Michael's face would be difficult to extract and may require the user to observe the original image to locate the required features.

At the moment the biggest disadvantage with the scale space approach is the incorporation of scale-space techniques into increasingly complex visual modes and the extension to non-linear scale-space concepts more committed to specific tasks [17]

Chapter 5

RESULTS

5.1 Testing the image by rotation.

This chapter examines what the effect of changing the parameters on the images. Images will be tested by rotation on the x-axis and z-axis. Java was implemented to display images on the DTI (Dimension Technologies Inc) monitor. A program was written that allowed the user to display 2 images side by side. The DTI monitor is a stereo monitor that does not require glasses. The stereo results are view on the DTI monitor and the results are commented on.

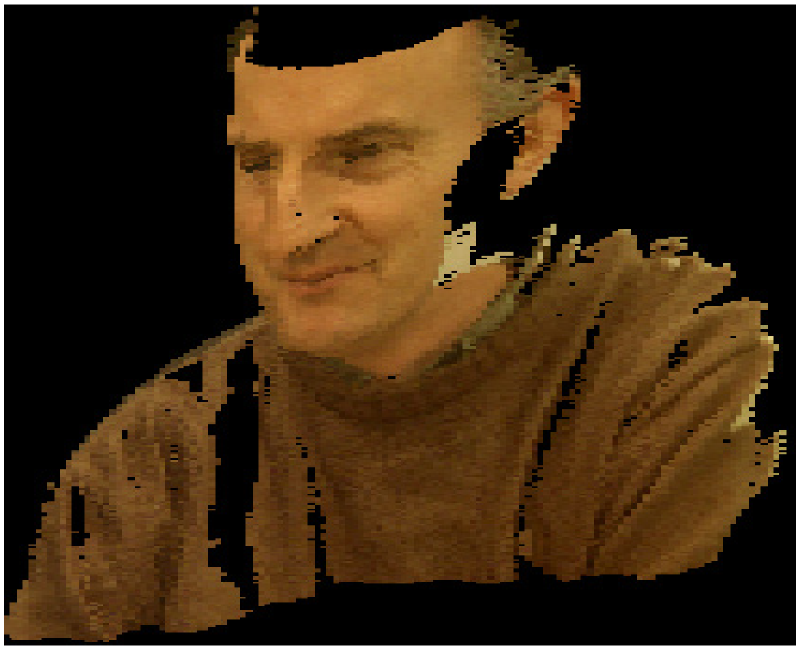


Figure 5.1 Rotation of Billy on the x axis.

Different translation values were tested on the x-axis. The results obtained gave a satisfactory result for different values. The x value that was taken for the above was 500mm. Examining the image in fig. 5.1 one can observe that Billy's right hand shoulder does not show through his face.

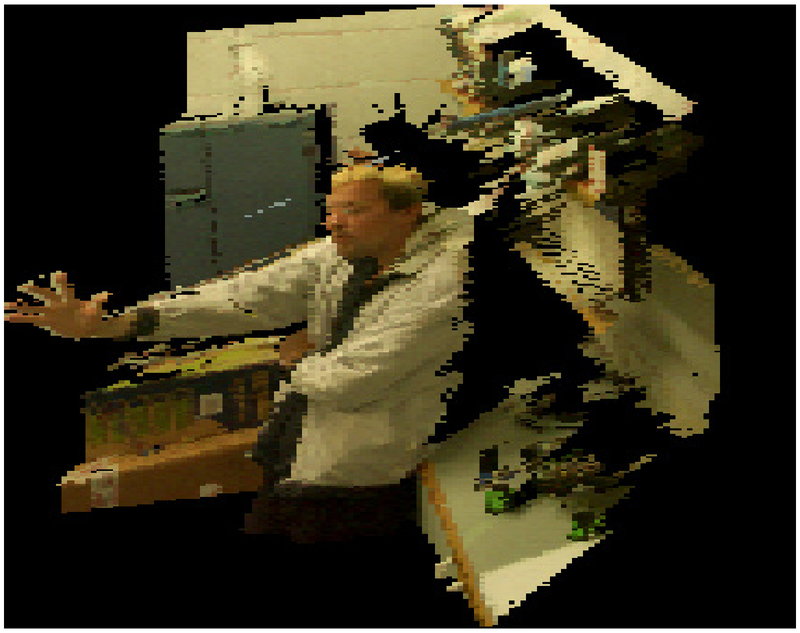


Figure 5.2 Rotation of Michael on the x axis.

The image of Michael was also rotated around the x-axis. The value for x was taken as 1800mm. In Fig. 5.2 it can be observed that the distinction between Michael's right arm and that background can be clearly seen. Overall the algorithm was easy to implement but its speed was limited by pixel redrawing. A large amount of memory space was required because frame buffer a storage buffer was also required. But if fine features were required high precisions would be needed.

5.2 Stereo view tests.

5.2.1 Eye distances.

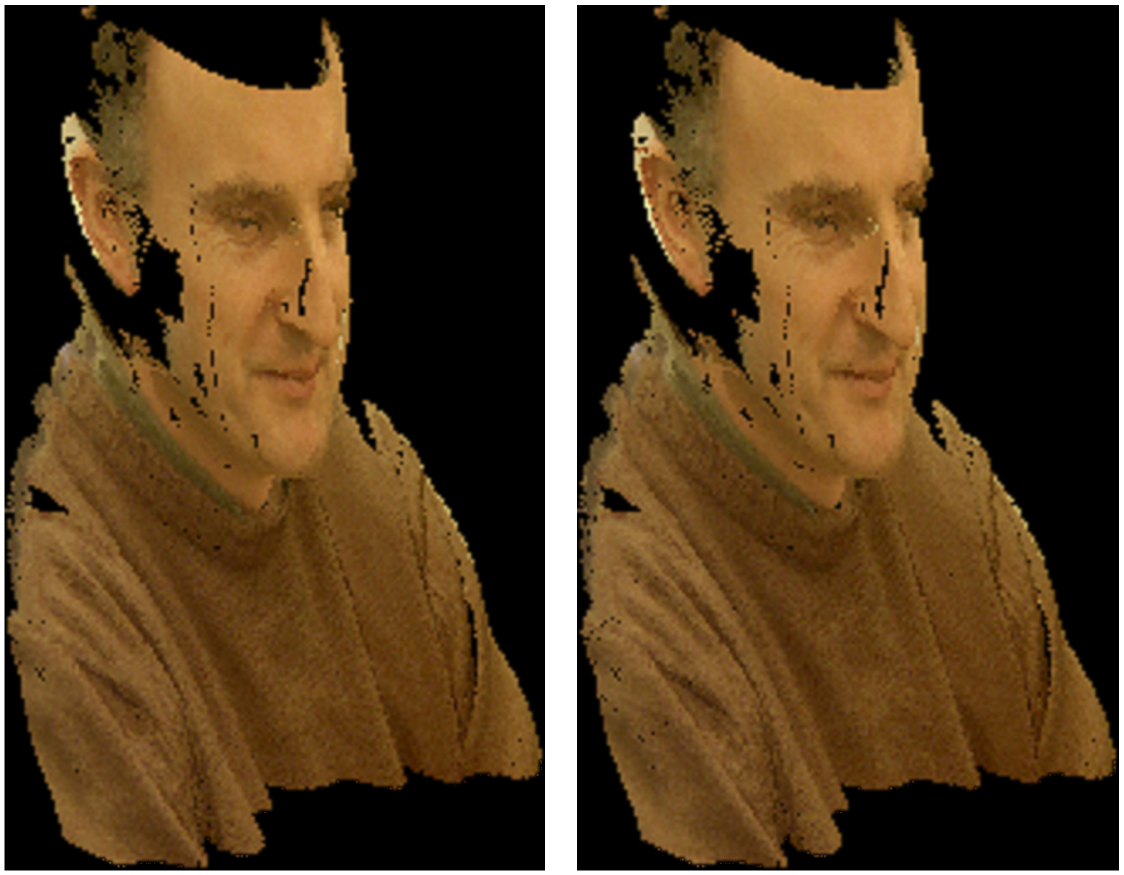


Figure 5.3 Stereo view of Billy

The above images represent a stereo view of Billy. Stereo viewing is a technique used to increase visual realism in 3D scenes. Each stereo view consists of two images, one for the left eye and one for the right eye. Stereo viewing increases visual realism. DTI interactive 3D image processing utility software was used to display these images on the 3D DTI monitor. A Java program was written which served the same purpose. The distance between the left image and the right image above is 100mm ($\approxeq$the distance between both left and right eye). The apparent depth of objects is a function of the difference in the positions from the left and right eye views.

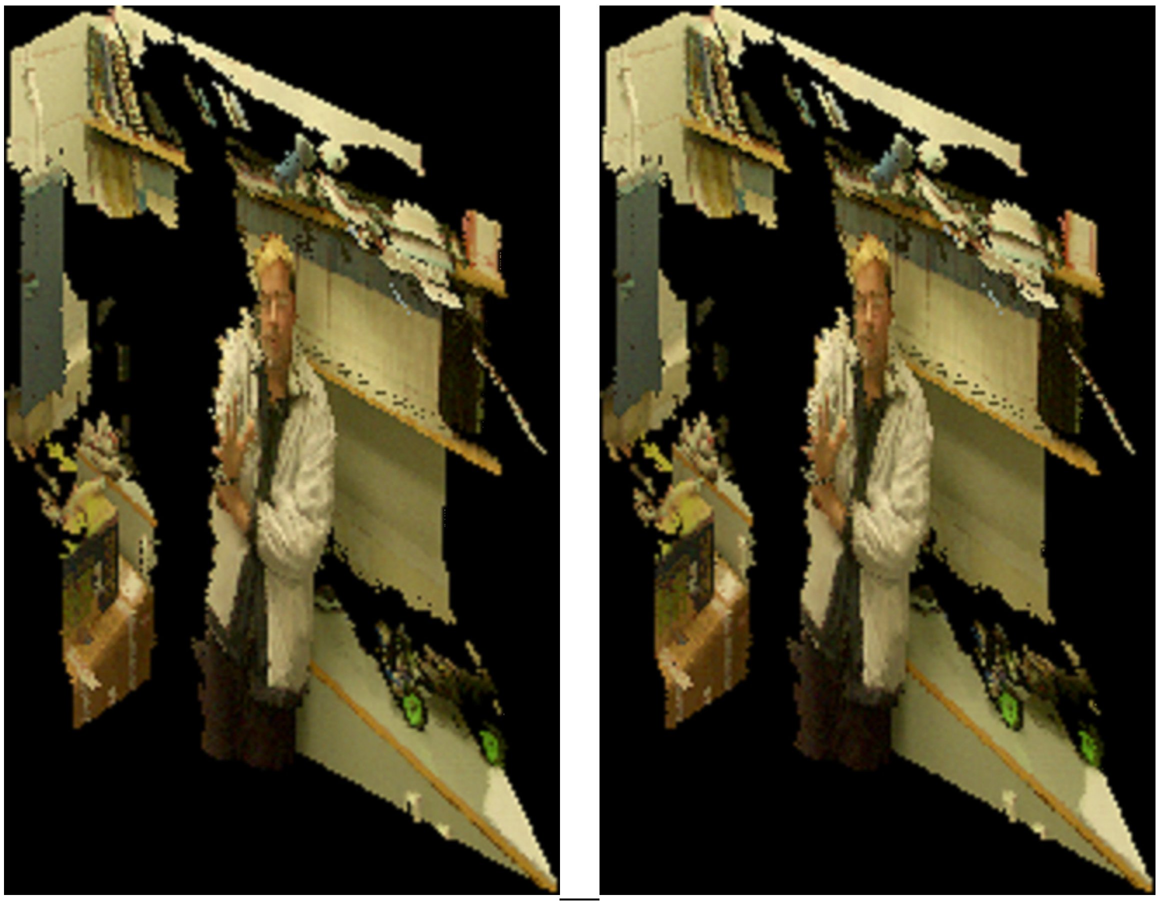


Figure 5.4 Stereo view of Michael

Fig. 5.4 gives a stereo view of Michael in the vision laboratory in DCU. The two images above include the view from the left and right eye. In this case the java program was used to view the images on the screen in 3D. The distance between both images was taken as being 100mm. Again the results were displayed on the 3D DTI monitor.

5.2.2 Larger eye distances.

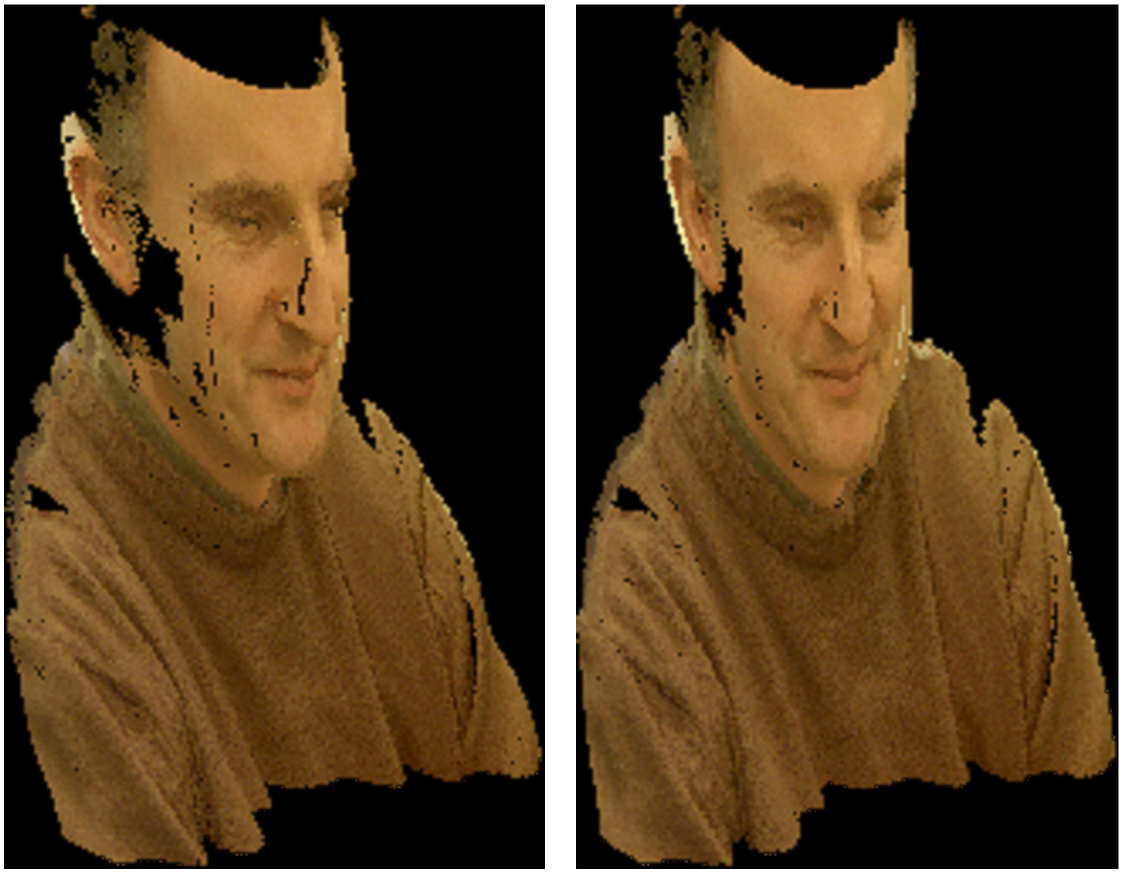


Figure 5.5 Stereo view of Billy with a large change in x axis

A large separation of around 1000mm was taken between the images in Figure 5.5. Observing the images we can see that there is a distinct difference between the view points of each image. Viewing these on the DTI monitor resulted in the images being out of focus. This made it difficult for the viewer to focus on distinct parts.

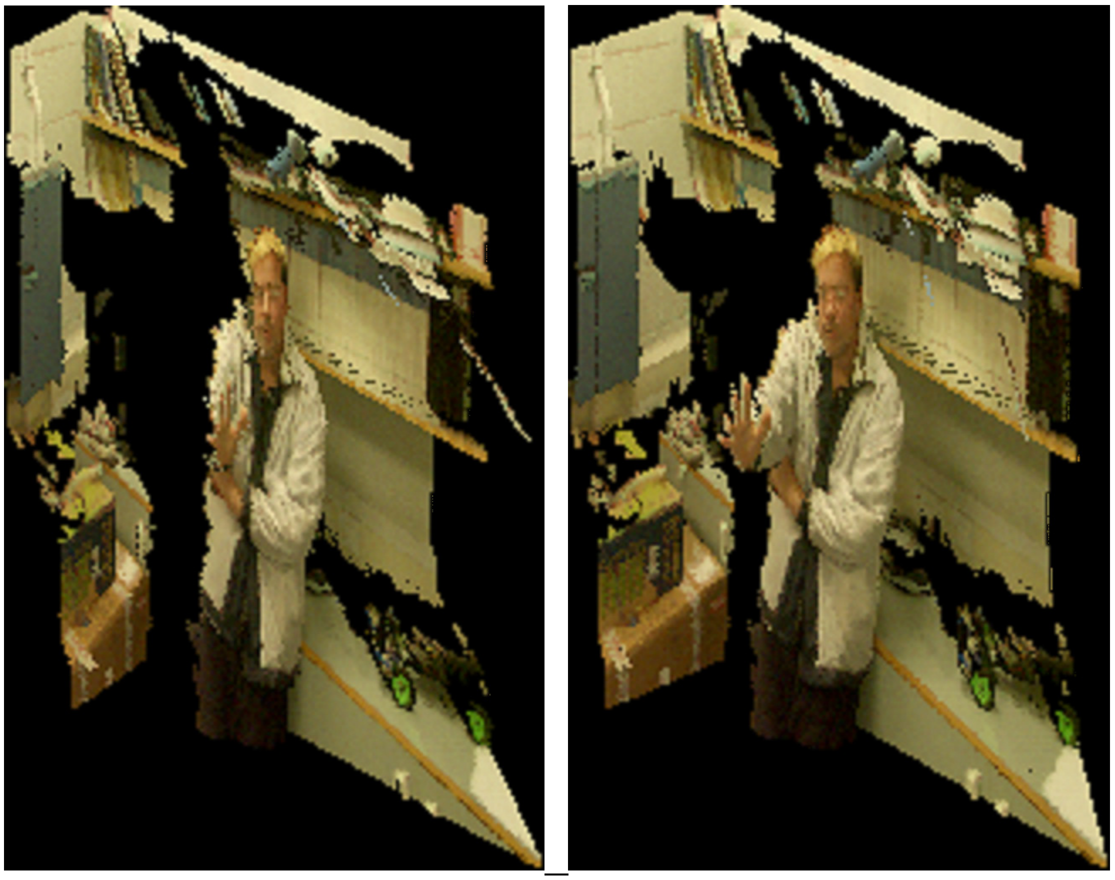


Figure 5.6 Stereo view of Michael a large change in x axis

The distance between these two images was changed to 1200mm. This resulted in the image being put off focus. Having a large separation meant that a viewer looking at the image on the 3D DTI monitor could not focus on distinct parts of the image which could give eyestrain.

5.2.3 Smaller eye distances

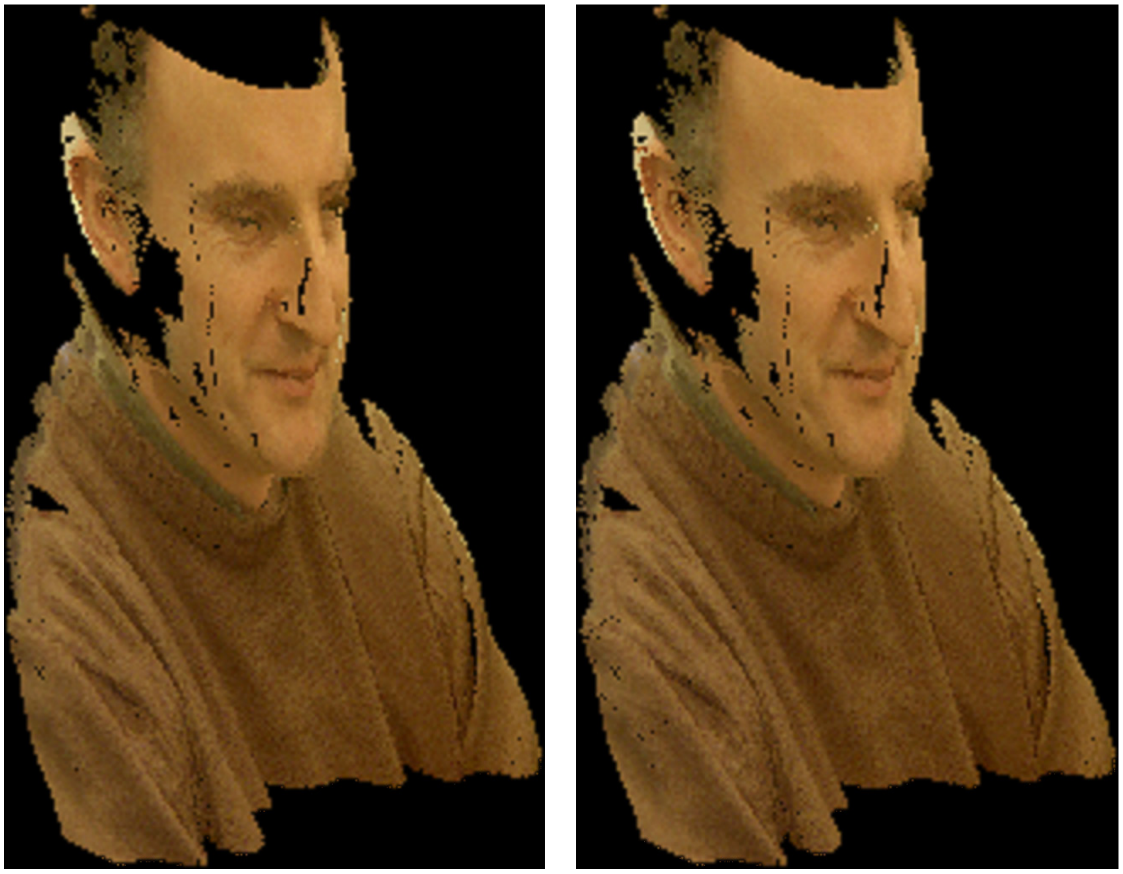


Figure 5.7 Stereo view of Billy with a small change in x axis

If the separation between the two cameras is too small then the 3D image on the DTI monitor will appear not as deep. The image will still appear to be in 3D but objects that should appear far away will not.

5.3 Variation of the z-axis

5.3.1 Large variation of the z-axis

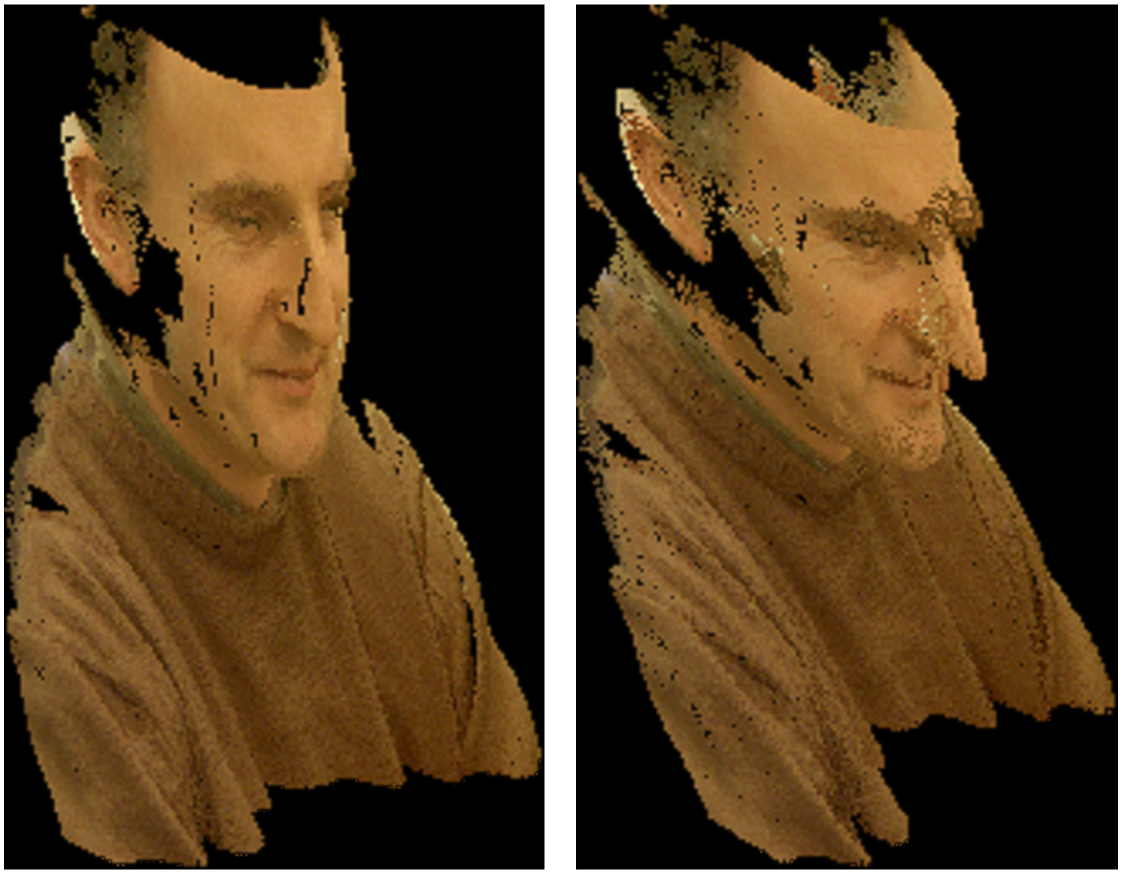


Figure 5.8 Stereo view of Billy with a large change in z axis

In this section of the thesis the z axis was varied from 0mm to 1000mm. One can notice that the right hand image in Fig. 5.8 is stretch in the z-axis. Again displaying these on the DTI monitor resulted in the image in 3D being out of focus.

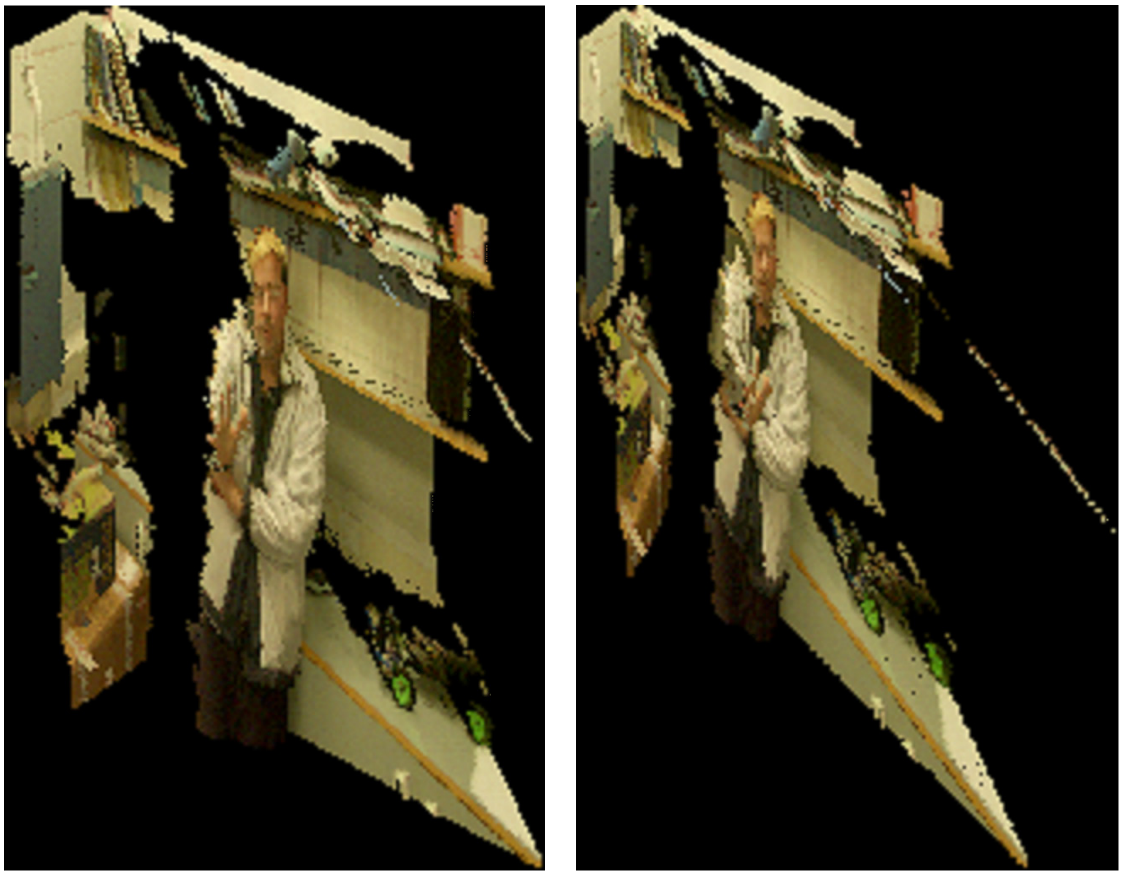


Figure 5.9 Stereo view of Michael with a large change in z axis

For this test the z axis was varied from 0mm to 1200mm. Again it can be observed that the right hand image in Fig. 5.8 is stretch in the z-axis. Displaying these on the DTI monitor resulted in the image in 3D being out of focus.

5.3.2 Smaller variation in z-axis.

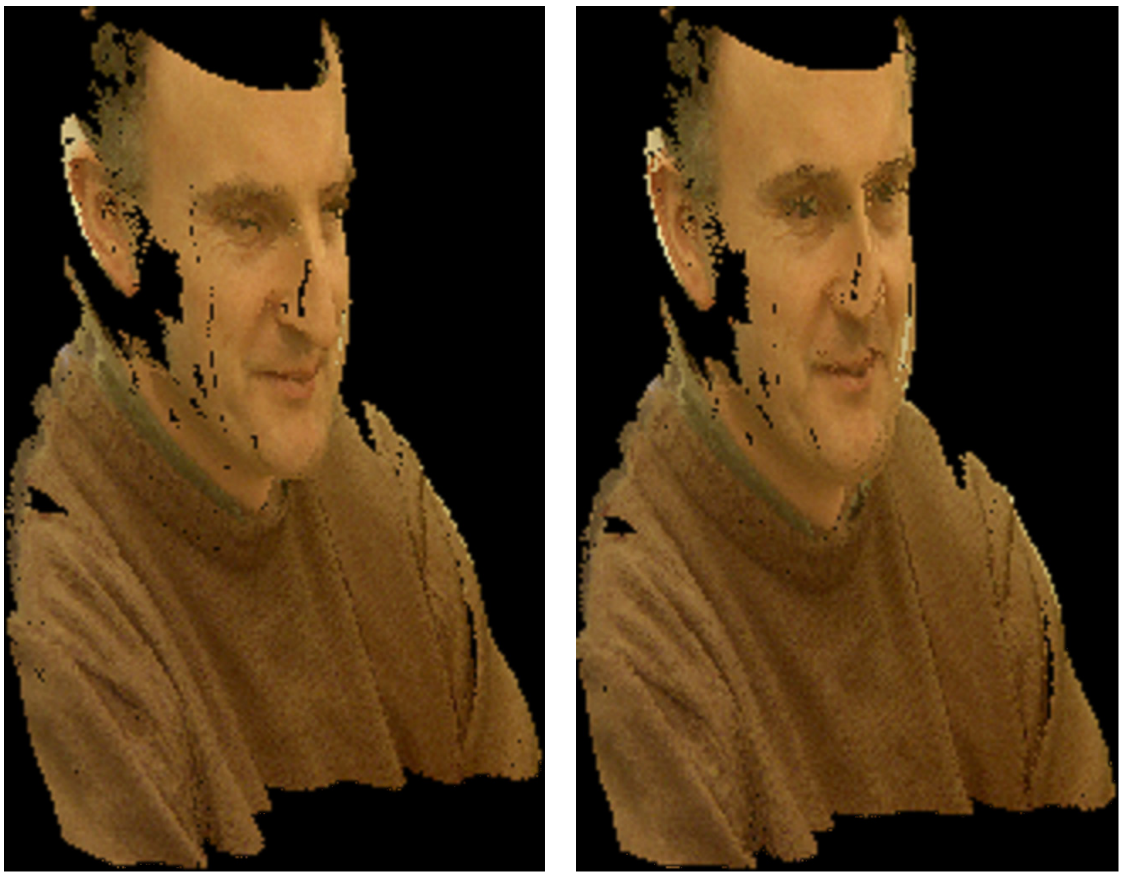


Figure 5.10 Stereo view of Billy with a small change in z-axis

The z values were set to -250 for this test. Looking at the right hand image in fig5.10 one can see that Billy’s face looks very flat. When these two images were displayed on the DTI monitor they were clearly out of focus. It would have been difficult to extract features in Billy.

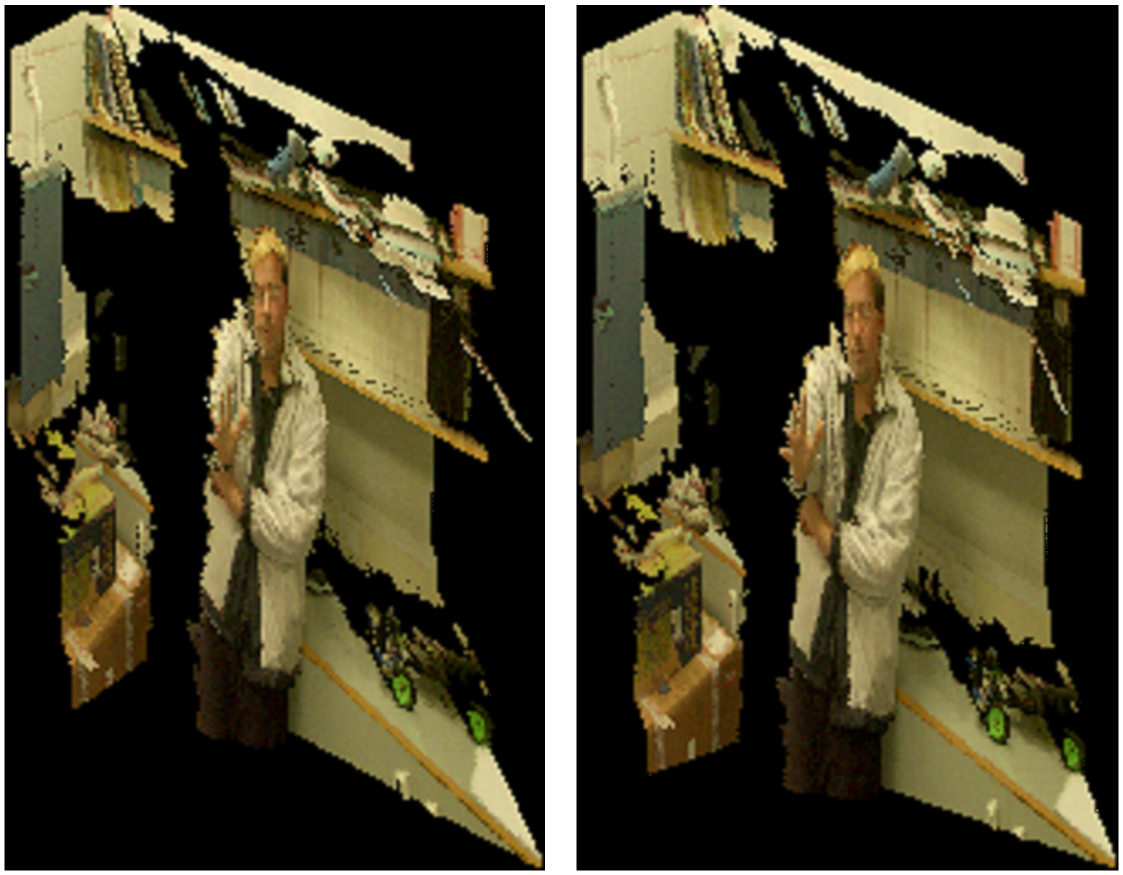


Figure 5.11 Stereo view of Michael with a small change in z-axis

Again setting the z-axis very small would cause this image to look flat. It would be hard to extract any detail from it.

Distances $\approx 100\,\text{mm}$ between the left and right eye images yield good images on the 3D screen. Large distances or too small a distance yield problems which were addressed. Therefore it is important to have a fixed distance between cameras to achieve correct 3D perception.

Chapter 6

CONCLUSIONS

In this thesis, we have successfully progressed in the generation of multiple images of a textured volumetric data set for visualization in three dimensions. The issues that have been addressed are the basic mathematics required to generate synthetic views and the problems posed as a result of generating these synthetic views.

The problem of occlusion was addressed and corrected. It was structured on the z-buffer algorithm proposed by Sung [11]. Various methods were examined to solve the re-sampling problem which had an issue with one pixel gets assigned more than one texture values, where the last value is kept, and another pixel getting assigned no value which results in the generation of a hole. After several tests using various filters it was decided on the scale space approach for yielding the best results. The filters that were used produced unsatisfactory results. Some experiments were carried out to investigate the 3D perception as the relative camera locations were varied. These results were examined in chapter 5.

References

[1] www.unimatic.co.uk/acrobat/Minolta_VI910.pdf This site provided the background information on the Minolta VI-910 laser scanner.

[2] http://www.esat.kuleuven.ac.be/~pollefey/tutorial/node115.htm Mention of a problem known as shadow occlusion.

[3] N. Elmqvist, P. Tsigas, "Reducing Occlusion in 3D Environments Through Smooth Camera Projection Animation."

[4] Richard Stoakley, Matthew J. Conway, and Pausch Pausch. "Virtual reality on a WIM: Interactive worlds in miniature." In Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems, volume 1 of Papers: Innovative Interaction I, pages 265–272, 1995.

[5] T. Todd Elvins, David R. Nadeau, and David Kirsh. Worldlets – "3D thumbnails for wayfinding in virtual environments". In Proceedings of the ACM Symposium on User Interface Software and Technology, 3D Interaction Techniques, pages 21–30, 1997.

[6] Shinji Fukatsu, Yoshifumi Kitamura, Toshihiro Masaki, and Fumio Kishino. "Intuitive control of "bird's eye" overview images for navigation in an enormous virtual environment." In Proceedings of the ACM symposium on Virtual Reality Software and Technology, pages 67–76, 1998.

[7] S.M. Seitz, C.R. Dyer, "Photorealistic scene reconstruction by voxel coloring", Int. J. Comput. Vision (IJCV) 35 (2) (1999) 151–173.

[8] M. Ulvklo, H. Knutsson, G.H. Granlund, "Depth segmentation and occluded scene reconstruction using ego-motion", in: S.K. Park, R.D. Juday (Eds.), Visual Information Processing, SPIE, vol. 3387, 1998, pp. 112–123.

[9] J. Maver and R. Bajcsy, “Occlusions as a guide for planning the next view,” Univ. of Pennsylvania, Tech. Rep. MS-CIS-91-27 GRASP Lab.257, 1991.

[10] Sung-Eui Yoon, “Comparison among two hidden surface algorithms and z-buffer algorithms”.

[11] K. Matsushita, T Kaneko, “An Efficient Image Mapping Method for 3D Surface Textures”, Systems and Computers in Japan, Vol. 32, No. 4, 2001

[12] Carlo Braccini, Giuseppe Marino, ‘‘Fast Geometrical Manipulations of Digital Images’’, *Computer Graphics and Image Processing*, Vol. 13, 1980, pp. 127-141.

[13] James D. Foley, Andries van Dam, “Fundamentals of Interactive Computer Graphics”, Addison-Wesley, Reading, MA, 1982.

[14] Fidrich Márta, “Multiscale analysis of invariants Application to volumetric medical image processing”

[15] http://www.brucelamond.net/documents/thesis.pdf Bruce J. Lamond, “An Investigation into the Recovery of Three-Dimensional Structure from Two Dimensional Images.”

[16] www.neatvision.com NeatVision is a Java based image analysis and software development environment, which provides high level access to a wide range of image processing algorithms through well defined and easy to use graphical interface.

[17] Lindeberg, Tony, "Scale-space: A framework for handling image structures at multiple scales", In: Proc. CERN School of Computing, Egmond aan Zee, The Netherlands, 8-21 September, 1996

Appendix

For the code for this project please find the attached CD.