# Acquiring graspable features on unmodelled objects using range data

Mark W Wright and Robert B Fisher

## 1 Introduction

This report describes how graspable features of unmodelled objects were obtained using range data. Output from this process is used for two purposes:

1. To provide a volumetric model to compute the center of gravity of the object and to test if a grasp is reachable.
2. As input to a grasp stability analysis to see grasps are force closed, *i.e.* that they can offset any wrench applied to the object.

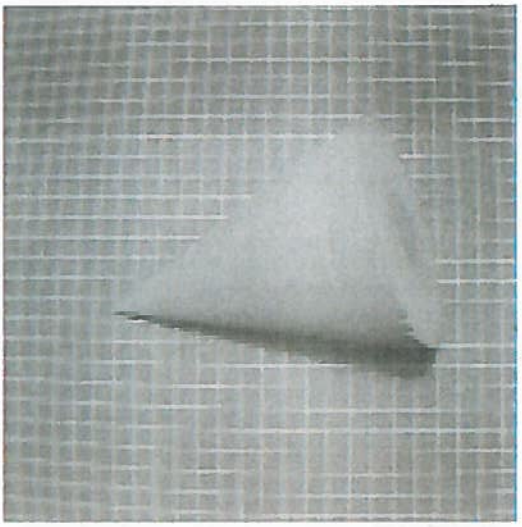
The volumetric model needs only crude space occupancy information. In contrast, the stability analysis requires the extraction of properties relating to the differential geometry of the object surface; specifically normal orientation, but also in some cases curvature information. Grasps on edges or corners will be unstable, so we also require that discontinuities are made explicit. Conversely, it is also desirable that these patches are reasonably large and few in number; this will aid any combinatorical search based on patch properties. These data requirements can be met if we recover smooth patches of the surface by fitting some kind of model as is done here.

The data was acquired using a laser striper built in house and was segmented using software which has been developed at Edinburgh over a number of years. A separate report [1], describes the details of the range data segmentation software; here, we describe its use to segment object surfaces into graspable patches.

The main stages of this process are:

- Acquire two range views of the object from different vantage points using the laser striper.
- Segment the data into patches based on the Mean and Gaussian curvature of the object.
- Fit planar and quadric surfaces to the data.
- Register the output from the two views.
- Post-processing:
    1. Patch subsampling to reduce computational costs.
    2. Patch erosion to allow for finger tip radius.

The structure of the report is as follows; firstly, we describe each of the above stages in detail. To aid this process we will use the cone in Figure 1 as a test object. Secondly, we present segmentation results for a data set of different objects. Finally, we discuss these results and reach conclusions as to the effectiveness of the method.



A right circular cone of height 70mm and diameter 50mm used to illustrate the various stages of the segmentation process.

Figure 1: Grey level intensity image of the cone

## 2 Acquisition of range data

The first stage is to acquire the raw range data using the laser striper. A schematic diagram of the striper is given in Figure 2. The striper calculates depth using triangulation; a stationary laser projects a light stripe vertically onto the object and the image of the stripe is captured by two cameras mounted either side of the laser. (Two cameras are used so data is not lost due to occlusion.) The object is passed through the beam of the laser by a moving platform which is powered by a linear motor.

To build a more complete description of the object we require data from two distinct views. This is achieved by putting the object on a rotating device shown in Figure 3 which is in turn placed on the moving platform. (The rotator simulates having a mobile scanner capable of taking data from multiple views which we envisage a robot would require in a fully integrated system.) The ground plane on which the object rests is rotated a specified amount from the horizontal (usually 45 deg) and a range image is taken. The object is then rotated the same amount from horizontal in the opposite direction and another range image is captured. The angle through which the object was rotated is recorded for the future registration of the range images. Two raw range images for the cone in two different orientations are shown in Figure 4.

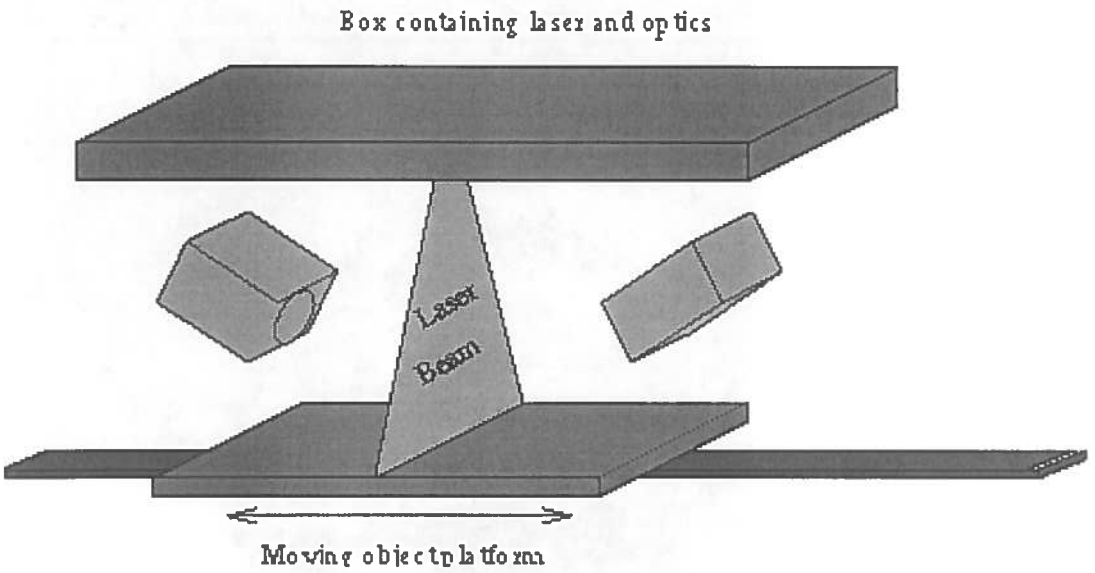



Figure 2: Schematic diagram of laser striper

# 3 Range data segmentation

The next stage is to segment the data into surface patches and this is done using the **rangeseg** program. **Rangeseg** is a complex program consisting of many processing modules, but these can be grouped into three main stages:

**Discontinuity detection** Rangeseg first performs a pre-segmentation of the data in order to find discontinuities. These discontinuities act as simple boundary conditions for the smoothing, labelling and fitting stages. If this is not done smoothing naturally degrades edges *etc* and leads to poor or incorrect segmentation.

**Curvature segmentation** An initial labelling of the surfaces is attempted. Classification proceedes according to the Mean (H) and Gaussian (K) curvature of the surface at each pixel.

**Surface fitting** Given the initial labelling, planar and quadric patches are fitted to the data using an iterative region growing approach.

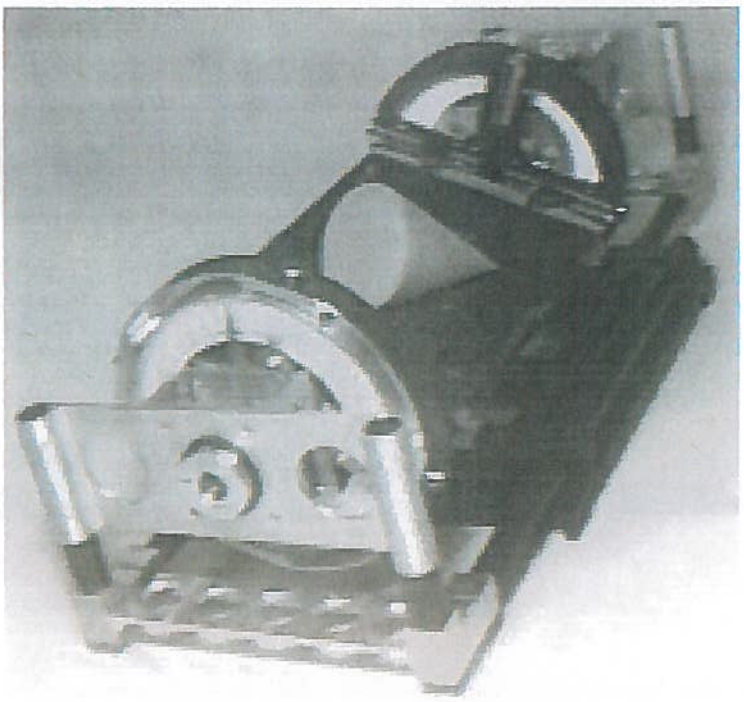
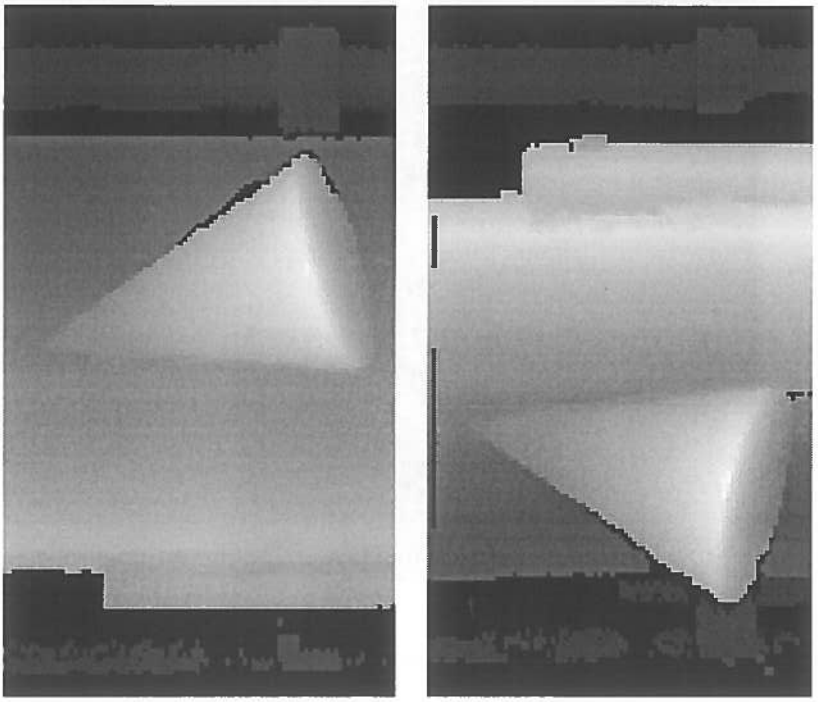


Figure 3: Rotating device used to acquire different views of the object

## 3.1 The order of the segmentation processes

A complete list of the processes and the order in which they are processed is now given:

| Input Image | -> | Process | -> | Output Image |
|---|---|---|---|---|
| **Discontinuity Finding** | | | | |
| 1) Raw Depth Image | -> | Depth Edge Detection | -> | Blade Edges |
| 2) Raw Depth Image | -> | Normal Calculation | -> | Normals |
| 3) Normals | -> | Fold Edge Detection | -> | Fold Edges |
| 4) Blade+Fold Edges | -> | Morphological closing | -> | Closed edges |
| 4) Closed Edges | -> | Non-smoothing regions | -> | Non-background |
| **Surface Smoothing** | | | | |
| 6) Non-background+Raw Depth | -> | Smoothing | -> | Smoothed Depth |
| **Curvature-based Segmentation** | | | | |
| 7) Smooth Depth | -> | Shape Analysis | -> | H+K Values |
| 8) H+K Values | -> | Distribution Analysis | -> | H+K Histogram |
| 9) H+K Values | -> | Thresholding | -> | H+K Thresholds |
| 10) H+K Thresholds | -> | Morphology | -> | H+K_MORPH |
| 11) H+K_MORPH | -> | Shape Sign Combination | -> | Shape Classes |
| 12) Shape Classes | -> | Grouping | -> | Unique Labels |
| **Surface Fitting** | | | | |
| 13) Unique Labels | -> | Surface Fitting | -> | Slurping |
| 14) Slurping | -> | Image marking | -> | Slurped Pixels |



Raw range images of the cone for rotations of $\pm 45$ deg. Depth is encoded as inversely proportional to intensity.

Figure 4: Raw range images of the cone

## 3.2 Discontinuity segmentation

Figure 5 shows the various stages of the discontinuity segmentation using one of the cone range images as input. The rangeseg output displays for the discontinuity segmentation are as follows:

**Raw Depth Image:** image of input data, dark pixels are further away. (The black line on the ridge is a colour map bug.)

**Blade Edges:** Green is position of blade edges.

**Non-Background:** Green is non-background, non-edge pixels (i.e. usable for smoothing).

**Smooth Depth:** depth image after smoothing.

**Normals:** cosine shaded image of the surface normals. Brighter means surface points more towards viewer.

**Fold Edges:** surface fold or orientation discontinuities in Blue, depth discontinuities in Green.

**Closed Edges:** gaps in the edges filled in using a morphology technique.

## 3.3 Curvature segmentation

The principal curvatures of the object's surface are estimated at every pixel and then classified based on the signs of the Mean (H) and Gaussian (K) curvatures. This table shows what surface types can be discriminated when used in combination.

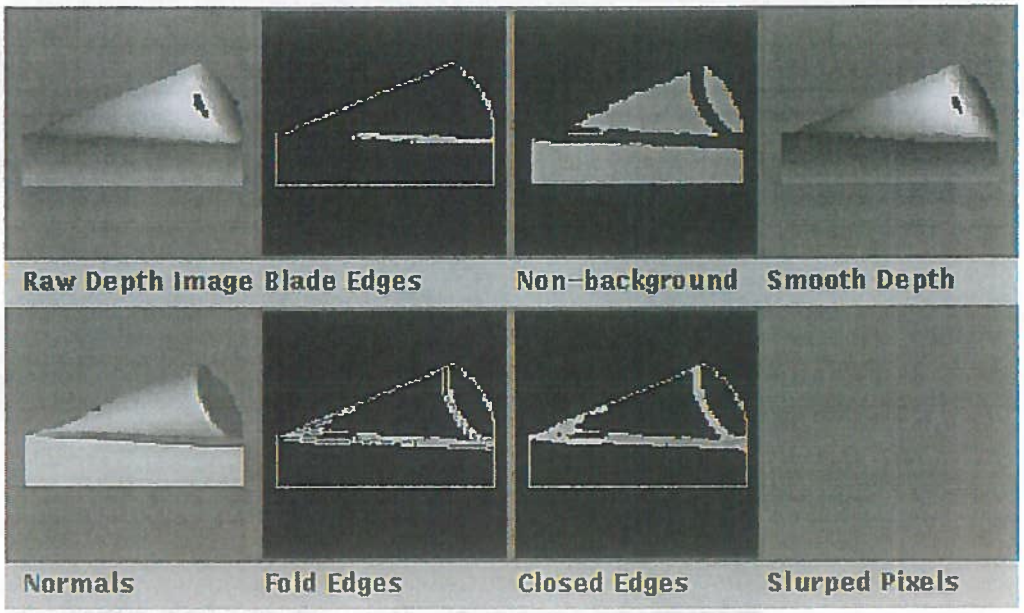



Figure 5: Discontinuity segmentation stages for the cone

| Curvature $K = \kappa_1 \times \kappa_2$ | Signs $H = (\kappa_1 + \kappa_2)/2$ | shape class | Display colour |
|---|---|---|---|
| 0 | 0 | plane | Yellow |
| 0 | − | negative cylindrical | Purple |
| 0 | + | positive cylindrical | Orange |
| + | − | negative elliptic | Blue |
| + | + | positive elliptic | Green |
| − | any | hyperbolic | Red |

*Surface patches classification scheme. The $\kappa_i$ are the principal curvatures ($H$ for a cylinder is $1/2R$ and for a sphere is $1/R$ where $R$ is the radius.)*

In the **rangeseg** program, the pixels in the H and K images are displayed with the signs of curvature coded by Red: negative, Yellow: zero, Blue: positive, and Purple: unlabeled. By grouping together connected pixels of the same "colour", the image is divided into surface patches. On the combined "Shape Classes" display, the colour coding in the table is used.

Figure 6 shows the various stages of the curvature segmentation using one of the cone range images as input. The rangeseg output displays for the curvature segmentation are as follows:

**H Values:** image shaded by the magnitude of the H curvature (brighter is larger).

**H Histogram:** histogram of H values (Red), with Hi curvature threshold bars (light Green).

**H Thresholds:** Initial pixel H curvature labeling using Lo and Hi thresholds.

**H_MORPH:** the result of applying the morphology schedule to the H Thresholds image.

**Unique Labels:** random colouring of all image regions segmented on the basis of curvature class.

**K Values:** image shaded by the magnitude of the K curvature (brighter is larger).

**K Histogram:** histogram of K values (Red), with Hi curvature threshold bars (light Green).

**K Thresholds:** Initial pixel K curvature labeling using Lo and Hi thresholds.

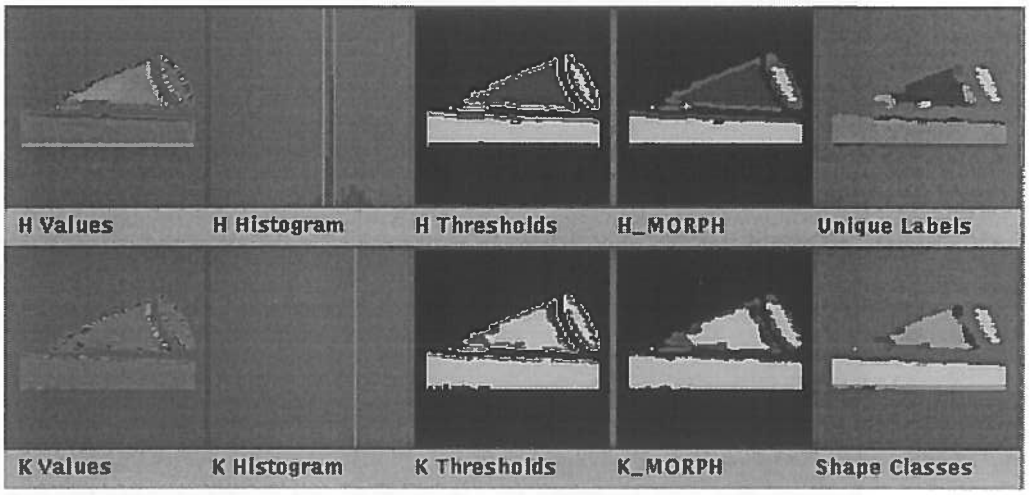



Figure 6: Curvature segmentation stages for the cone

**K_MORPH:** the result of applying the morphology schedule to the K Thresholds image.

**Shape Classes:** combined H & K labels to classify surfaces into one of 6 shapes, colour coded as follows:

**Plane** Yellow

**Hyperboloid** Red

**positive Ellipsoid** Blue

**negative Ellipsoid** Green

**positive Cylinder** Orange

**negative Cylinder** Purple

## 3.4 Surface fitting

The final segmentation process in **rangeseg** is the iterative surface fitting module called slurp. Figure 7 shows the output for this process.

**Slurped Pixels:** Shows what portions of the image have been tested for fitting a surface already (dull green is portions still being processed).

**Slurping:** Shows what pixels belong to each surface fit. Each region (ideally) has a colour different from its neighbours.

## 3.5 Parameter Variation

We now describe in turn the various parameters that effect the segmentation processes. We start with the parameters which control the boundary fitting process:

**Depth Disc. Thresh. (mm):** Specifies the maximum amount (in mm) adjacent surface depths may vary before blade or depth discontinuity edges are declared. Affects what is marked in green as an edge in the "Blade Edges" display. Too small a value means depth discontinuities are found everywhere, too large means the object is not even segmented away from the background.

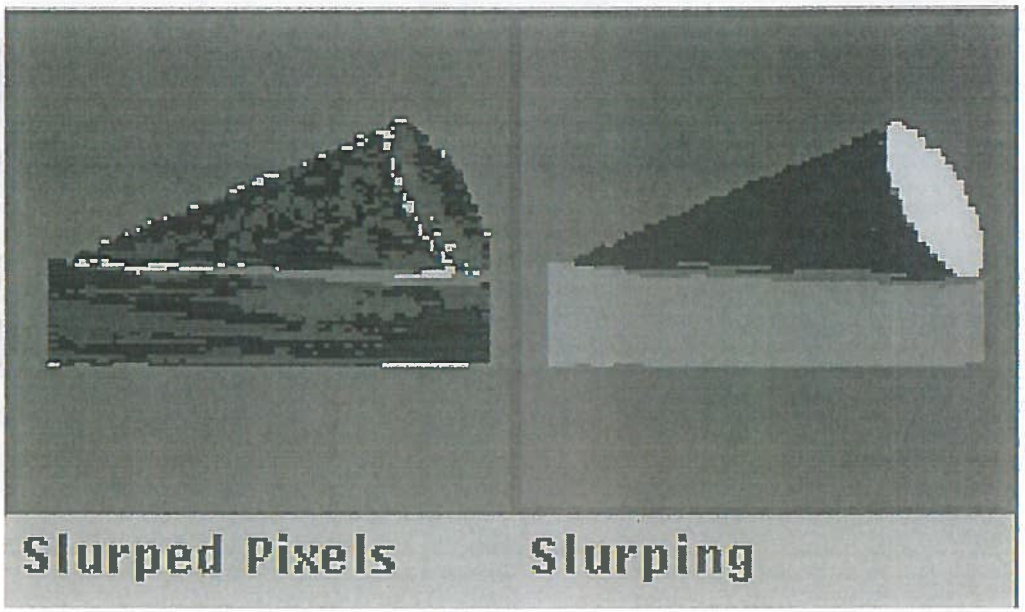



Figure 7: Surface fitting stages for the cone

**Fold Edge Threshold (deg):** Specifies the maximum amount (in degrees) adjacent surface normals may vary before orientation or fold edges are declared. Affects what is marked in blue as an edge in the "Fold Edges" display. A too small threshold results in fold edges everywhere from image noise and a too large threshold means no fold edges are found.

**Number of Smoothings:** the number of times smoothing is applied to remove image noise. Affects what is seen in the "Smooth Depth" display. Too few smoothings leaves noisy data, giving a rather fragmented segmentation. Too many smoothing iterations distorts the shape of the object. A small number is probably best here.

Next to be described are the parameters which control the initial H+K pixel classification.

**H Curvature Thresh Hi(or Lo) (1/mm):** Any pixels whose absolute value of the H curvature is larger than the Hi value are definitely curved and are labeled as positive (Blue) or negative (Red) depending on the sign of curvature. Any pixels whose absolute value of the H curvature is smaller than the Lo value are definitely planar (Yellow). Any pixels in between are unlabeled (Purple). The corresponding Gaussian curvature thresholds are defined as a function of the H thresholds. Because of image noise, some pixels will always be mis-labeled. The low threshold should be set so that planes are mostly labeled as plane or unlabeled. The high threshold should be set so that curved surface are mostly labeled as curved or unlabeled.

**Morphology Schedule:** this is a string of "+" and "-", which denote dilations and erosions, respectively. So, a "++-" means 2 dilations followed by 1 erosion. Each dilation fills in unlabeled (Purple) pixels by labeled (Red, Yellow, Blue) neighbours, thus removing unlabeled pixels. One might want to do this to make larger regions. Each erosion expands unlabeled pixels into labeled pixels, thus removing labeled pixels. This might be required to remove falsely labeled pixels (arising because of noise, etc.) If one erodes too much, then one can lose all of the good pixels. If one dilate too much, one might merge regions inappropriately. The correct goal is to obtain large regions of the correct surface class. Usually only a few dilations and erosions are needed.

Finally we describe the parameters controlling the surface fitting. By this time the segmentation should should consist of 1-2 large patches per true surface.

**Minimum Region Size:** minimum allowed number of pixels before a surface fit is attempted. A too small number creates many noisy fragments that will try to be made into surfaces. Too large means the major patches won't get started.

**Fitting Residual (mm):** the allowed deviation between a plane (or biquadratic) surface being fitted to the data and a data pixel. If the deviation is larger, then the pixel is not added to the fitted surface. Too small means that the patch won't grow to the edge and much internal data will be missed. Too large causes the patch to grow past its boundaries. The best parameter value (e.g. $3\sigma$) is related to the sensor noise level $\sigma$. Here the noise level $\sigma$ is approximately 0.15mm.

The frequency of the laser stripes on the objects is one stripe per millimeter. The table of figure 8 lists the actual parameters used.

| *parameter* | *value* | *units* |
|---|---|---|
| Depth threshold | 2 | mm |
| Fold threshold | 62 | degrees |
| No of smoothings | 4 | integer |
| H curvature threshold (Hi) | 0.05 | 1/mm |
| H curvature threshold (Lo) | 0.04 | 1/mm |
| Morphology schedule | +-++ | n/a |
| Min region size | 15 | integer |
| Fitting residual | 1.5 | mm |
| No of Slurp iterations | 7 | integer |

Figure 8: Parameters used in **rangeseg**

# 4 Registration of range data from the two views

Once **rangeseg** has produced planar and quadric surfaces for the two different views these need to be registered with one another. In this instance this was done manually although the group has worked on automatic registration of range images. Range images of the rotator itself were taken in various positions and the axis of rotation extracted. The range data was then rotated about this axis by the amount recorded earlier from the rotator dial. It was found that the registration was sometimes a little off and minor corrections were needed to be made manually. Recall that the purpose of registering the two images by hand was to simulate the effect of a mobile scanner, we assume that the transformation between the two views would be known in this case. Figure 9 shows the registered range views for the cone. Generally the error was a translation offset rather than a rotation error but in this case we can see a slight error in the alignment of the ground plane. Having registered views we will get overlapping segments, ideally these should be removed but we have ignored this problem as the grasp synthesis algorithms are robust to this problem.

# 5 Post processing of the patches

Once the data is registered it is loaded into *Matlab* for the final post processing before being passed to the grasp synthesis software modules. This processing consists of 'rastering' the data on meshes for each patch, eroding the patches and ,if necessary, subsampling the patches.

When the data arrives from **rangeseg** each patch consists of a quadric or plane equation, a list of the points which are on that patch and the surface normals at those points. The first task

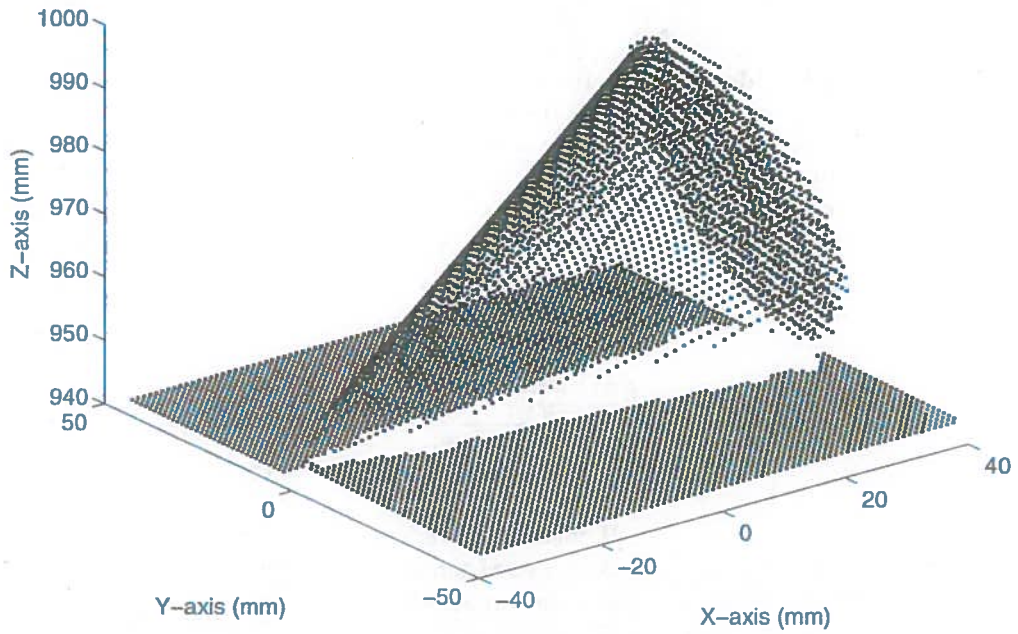



Figure 9: Registration of range data views for the cone

is to 'raster' the data into a 'mesh' surface. This process is required because there is no concept of adjacency within the set of raw surface points and this is required for certain operations such as:

- Displaying the patch as a connected mesh rather than a cloud of points.
- Erosion of the patches.
- Determination of extrema of quantities such as torque on each patch.

Putting the cone example to one side for a moment, in Figure 10a we see the 1482 raw patch points for a a typical quadric patch. Figure 10b shows those points made into a mesh. Some points have already been lost because the mesh plotting algorithm cannot display a mesh for points which have only two neighbours. Figure 10c shows the patch after erosion. This is achieved using a simple iterative erosion by one pixel applied four times. Any patch pixel which is 4-connected to the background is deleted. At this stage there are 669 points in the patch. Figure 10d shows the result of subsampling the patch by a factor of three. This is done by simply skipping pixels and creating a sparser mesh with the rest of the data. At this stage there are 77 points left in the patch. The subsampling function has not been used much in the project as it was found that the techniques employed did not lead to large combinatorical evaluation problems.

To conclude the detailed description of the processing stages, in Figure 11 we return to the cone description which consists of registered mesh patches which have been eroded and subsampled.

# 6 Segmentation of the data set

Figures 12- 19 show the segmentations achieved for a number of different objects. In each case we show the grey level intensity image for the object from which the size and shape of the object

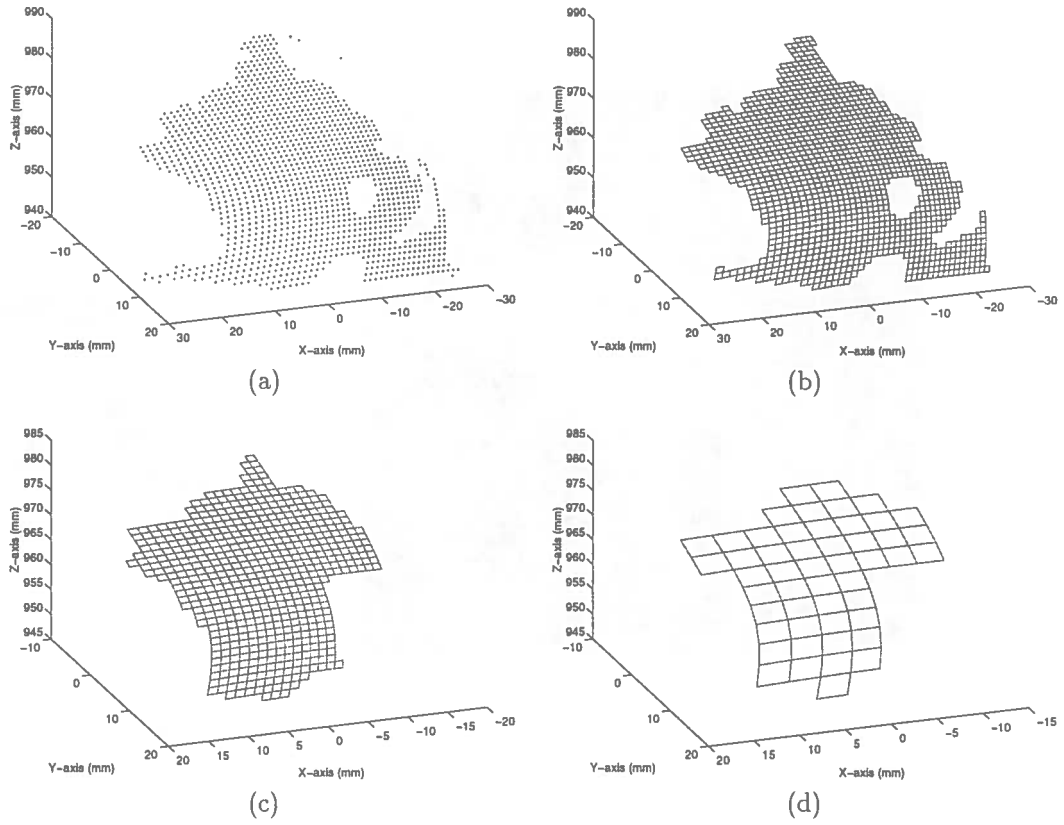



Figure 10a shows the raw points of the quadric. Figure 10b shows the 'rastered' mesh surface. Figure 10c shows the eroded patch. Figure 10d shows the subsampled patch.

Figure 10: Post processing operations for a typical quadric patch

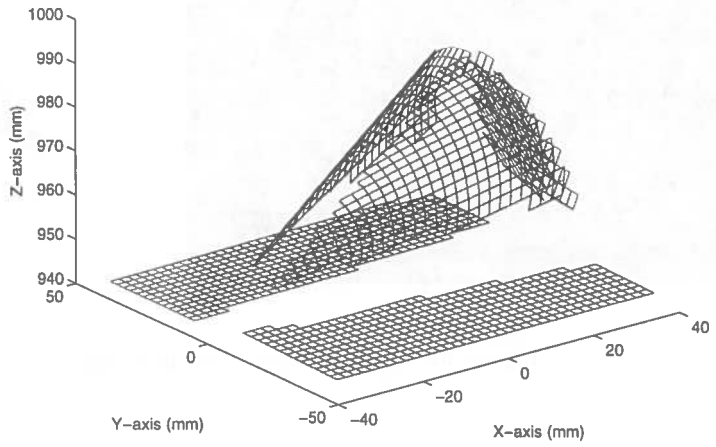



Figure 11: Final data after postprocessing for the cone

can be judged (the objects lie on a square grid of 5mm pitch). Beside the grey level image we also show the final post-processed segmentation using erosions of two pixels and subsampling by a factor of two in all cases.

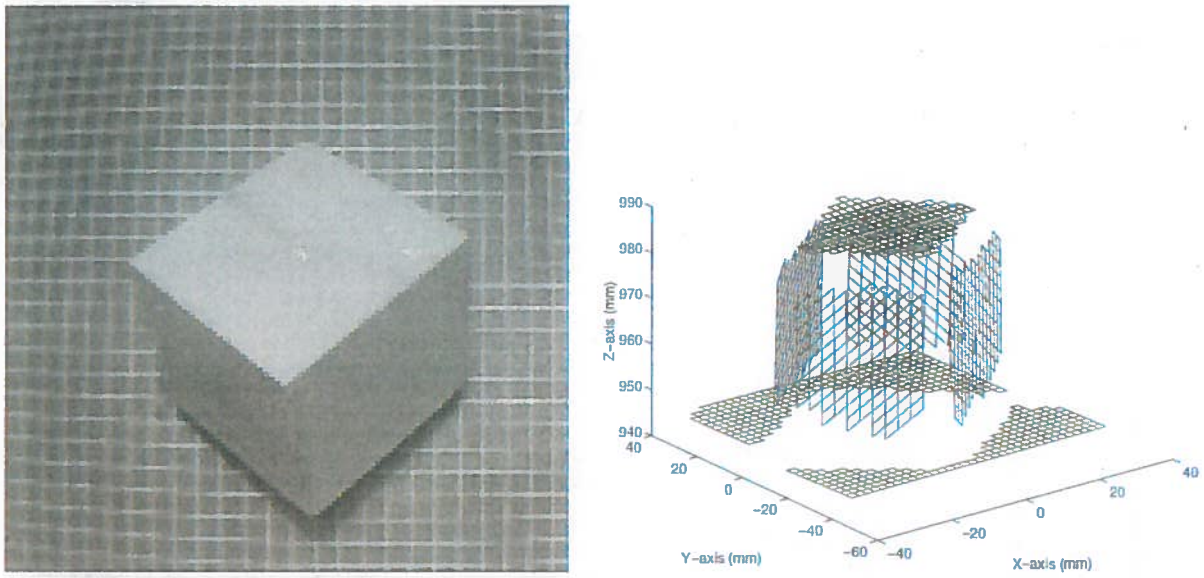



Figure 12: Grey level image and final segmentation of a cube

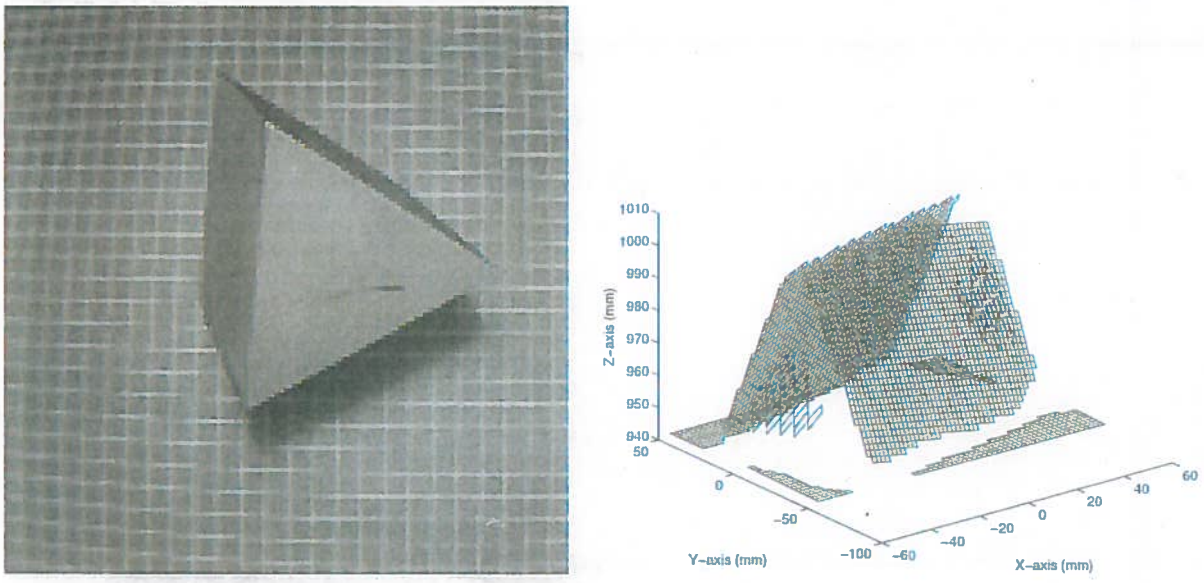



Figure 13: Grey level image and final segmentation of a prism

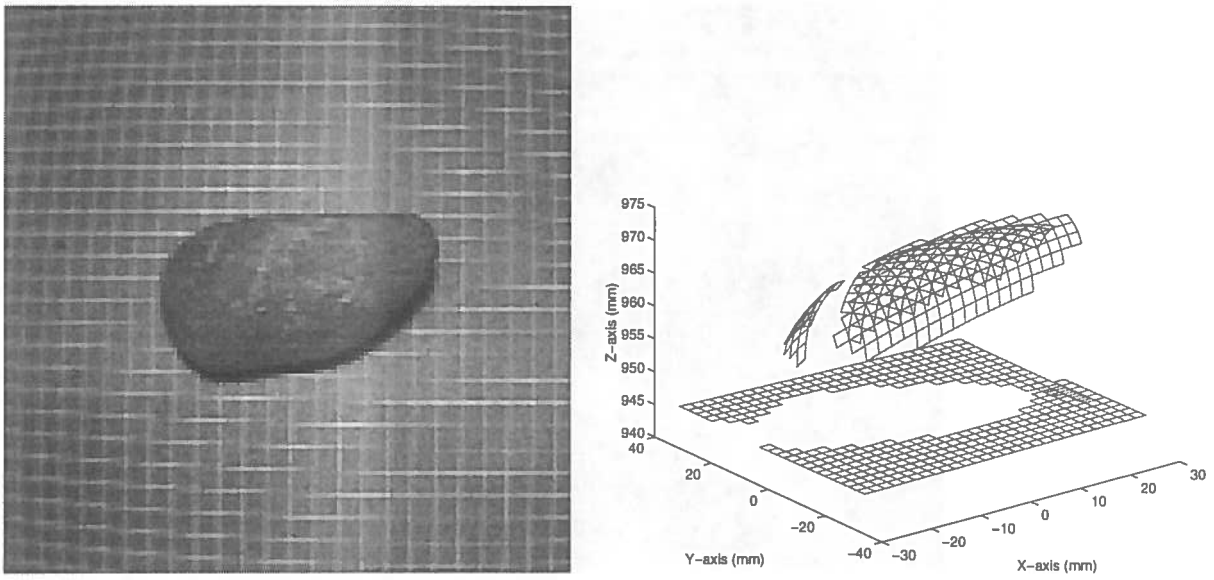



Figure 14: Grey level image and final segmentation of a smooth stone

# 7 Discussion

The segmentation of the planar objects in Figure 12 and Figure 13 are quite good. The faces of the cube and prism are completely segmented apart from the gaps made by the erosion process to prevent the fingers from being applied too close to the edges.

Segmentation of the smooth and faceted stone in Figure 14 and Figure 15 are less good. Although the segmentation is complete on the top of the objects much detail is missing on the undersides of the objects. This is because, unlike the cube and prism which have bases flush with the ground, these objects have bases that are curved and recessed. This would mean that the laser stripe could not penetrate here. The erosion process further lifts the curtain of the patch data above the ground plane.

In the segmentation of the indented stone of Figure 16 and the rough stone of Figure 17 we begin to see more serious modelling errors. Although the upper surface of the indented stone is well displayed the concavities in the object are not modelled. The surface of the rough stone has led to large portions of the surface not being modelled at all.

Although the broken pipe of Figure 18 has smooth surfaces it is poorly represented in the final segmentation. This is due to the the erosion and subsampling being too great for the scale of this object and the range data sampling frequencies.

The forked stick of Figure 19 is modelled most badly. The rough dark surface and recessed base have all conspired to give a poor segmentation. This object gave **rangeseg** the most trouble even generating impossible 'ghost' points under the ground plane.

The performance of the segmentation process can be explained in terms of the primitives used, which are quadrics. This system was developed to model man-made objects which quadrics and planes fit well. Rough natural object surfaces tend to disrupt the segmentation into many arbitrary patches. These patches can then be eroded away by the morphological processes of the curvature segmentation or the post-processing. An alternative would be B-spline surface patches which could be deformed using an active contour approach and would enforce continuity constraints at patch boundaries.

The problem of unmodelled areas due to recession of object bases could perhaps be alleviated by more intelligent post-processing which would, for example, not erode patch edges that were close to the ground plane and nearly vertical.

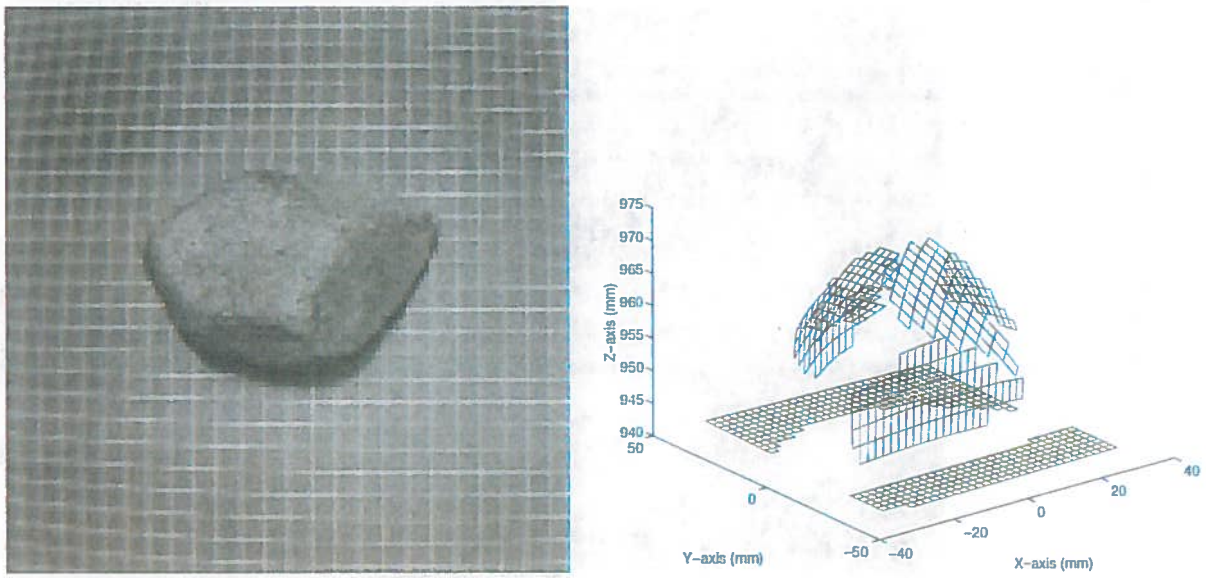



Figure 15: Grey level image and final segmentation of a faceted stone

## 8 Conclusions

The segmentation process produced a small number of graspable opposing patches for all objects and so we conclude that it achieved its primary objective. However, the performance is not entirely uniform; the segmentation process works well on man-made objects but less well on natural objects. For natural objects the the **rangeseg** segmentation is very sensitive to segmentation parameters. These problems stem from the fact that the system was primarily designed for object recognition and reverse engineering of man-made objects and the plane/quadric surface primitives were chosen as they model a wide range of these objects well. It is difficult to obtain a good automated segmentation of a natural object using a model based on quadrics and planes. We therefore suggest that future work should investigate the use of other primitives such at B-splines. Missing data due to undercutting at object bases is a problem, intelligent erosion may offer a solution. The scale and type of object should be considered when choosing erosion and subsampling parameters. Despite all these problems it does appear to be possible to obtain patches suitable for grasping from a variety of objects using range data.

## References

[1] A. W. Fitzgibbon, D. W. Eggert, and R. B. Fisher. High-level CAD model acquisition from range images. *Computer Aided Design*, page (submitted), 1996.

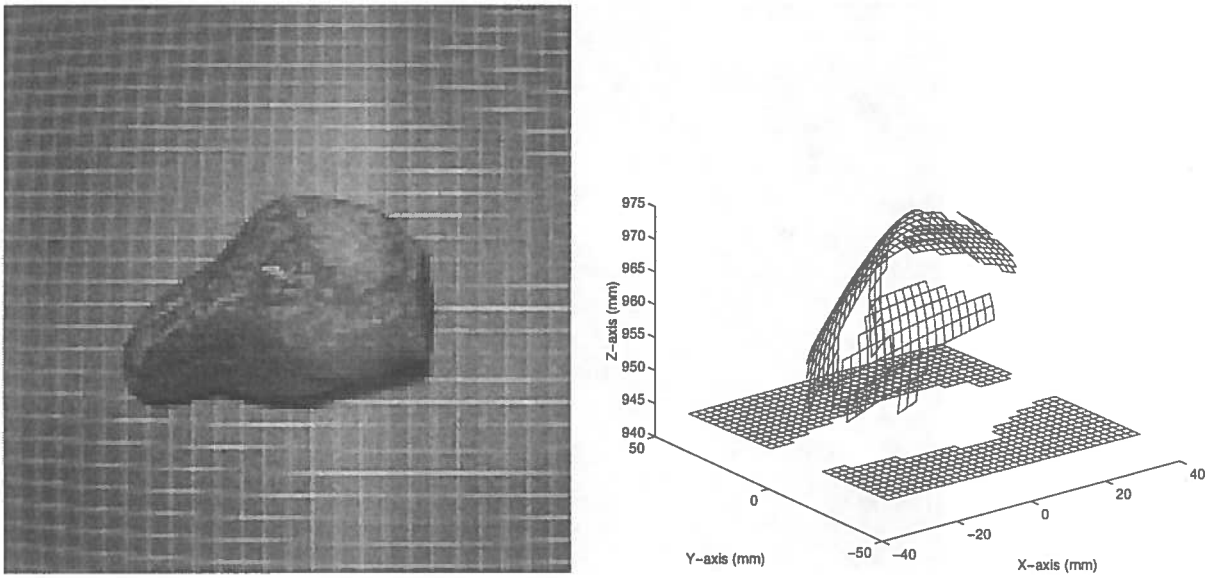



Figure 16: Grey level image and final segmentation of a indented stone

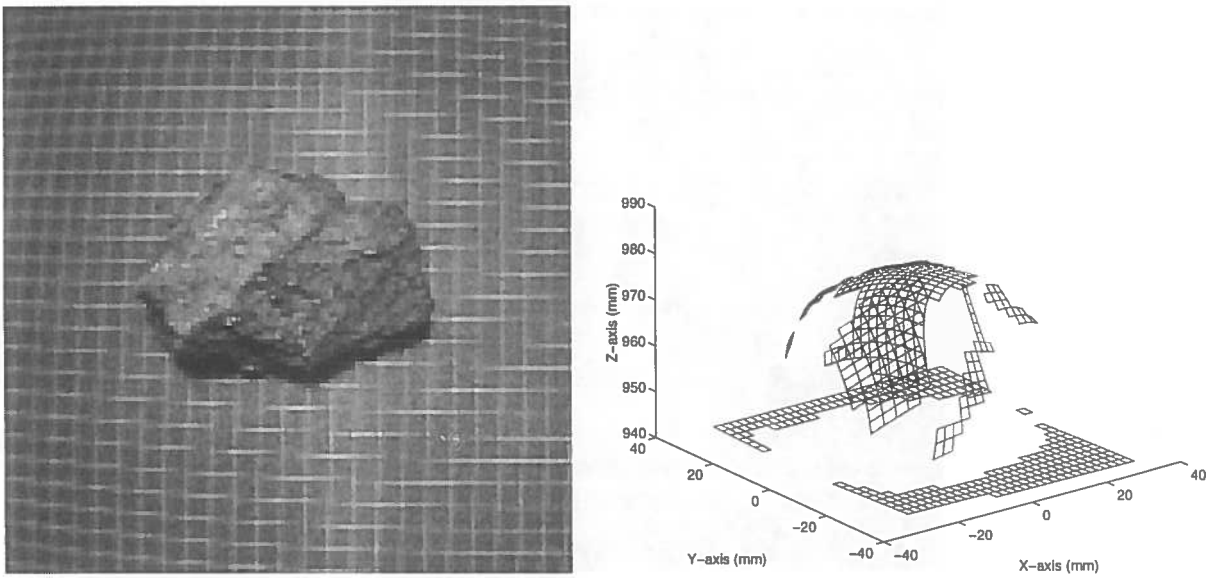



Figure 17: Grey level image and final segmentation of a rough stone

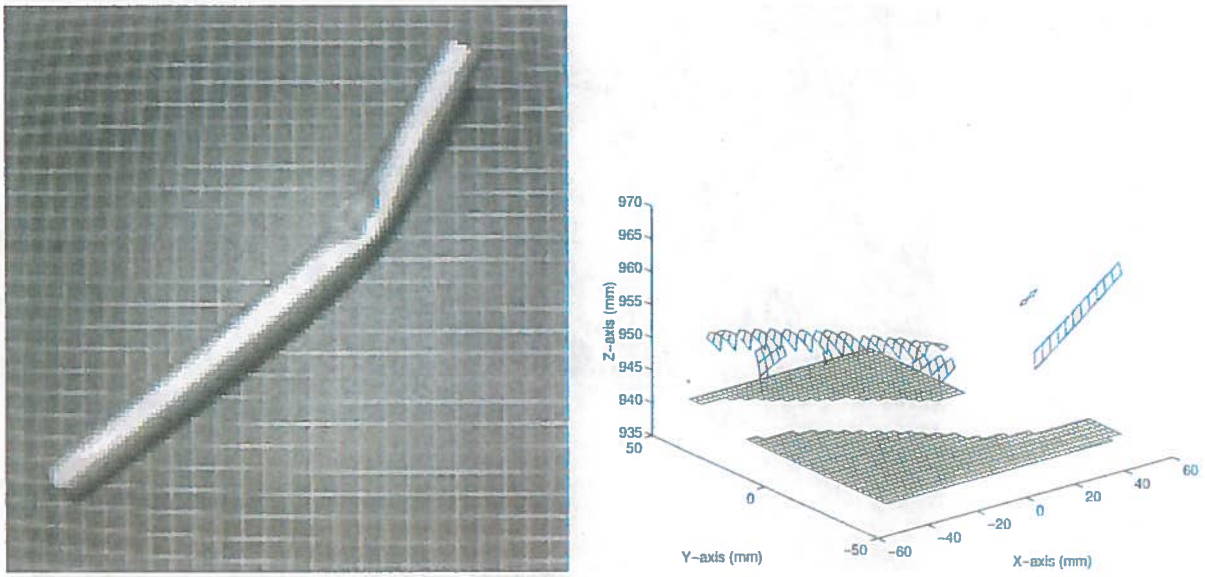



Figure 18: Grey level image and final segmentation of a broken pipe

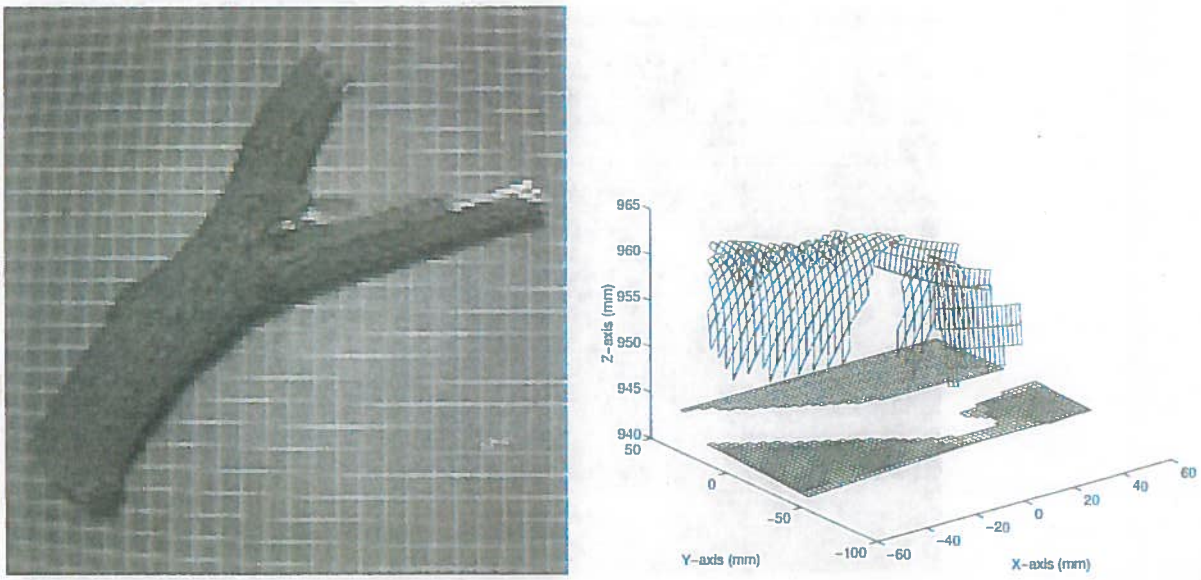



Figure 19: Grey level image and final segmentation of a forked stick