University of Edinburgh
School of Informatics

3D sensor planning for robotic grasping

4th Year Project Report
Artificial Intelligence and Computer Science

Toby Craig

May 26, 2004

Abstract: This project considers the Best Next View problem for a robot using range data to find a suitable grasp with which to pick up an object. To date, the Best Next View problem has concentrated on getting a complete range image of a scene or object, requiring many views. This project instead aims to gather enough data to allow a reasonable grasp to be found from just a few views, rather than the dozens required for a complete scene. Data capture is simulated, using real range data views of objects from different positions.

Acknowledgments

Many thanks to my supervisor, Bob Fisher, who provided me with guidance and advice throughout my project. Also to Toby Breckon of the Vision Lab, for his help with the range scanner and other assistance in collecting data. Lastly, I would like to thank my friends and family for their support.

Dedicated to the memory of my father, Tom Craig.

Contents

1. Introduction

A common problem in robotics is that of grasping an object. This is hard enough when the size, shape and orientation of the object are known; the problem is exarcebated when a robot wishes to pick up an unknown object in front of it. Range data from several views can be used to identify suitable grasping points on an object; however, range images take a long time to obtain. Previous work on selecting the Best Next View has concentrated on minimising the number of images needed to get complete and high quality coverage of an object or scene; however, this is far more than is needed simply to pick up an object. This project looks at a method for minimising the number of views needed to gather enough data to find a reasonable grasp.

1.1 Motivation

Range data scanners have been used for some time now to produce 3D representations of scenes. These range images are similar to a normal visual image, but have a depth value associated with each pixel, rather than a colour or light intensity. Typically, several range images of a scene will be combined to produce a single collection of 3D points that covers the whole scene; this will often then be converted into a mesh to allow further analysis of the objects in the scene. Complex scenes generally require many high detail views, which are time-consuming to obtain. Hence previous research has been directed at determining from which position the scene should next be observed to maximise the amount of new information captured and minimise the number of views required to capture the entire scene.

It is worth noting the difference between range images and simple 3D point sets. Range images are *structured*: they are 2D arrays like normal images. Points are stored in order, so it is a simple matter to find data points that are adjacent to a given point. This helps with tasks such as plane finding, where the problem is vastly simplified by simply having to inspect neighbouring points, which are trivial to find, instead of having to inspect every point in the set. 3D point sets, on the other hand, are simply unordered collections of 3D points. Although range images can be very useful, they lose their ordered property when two or more are registered. This project, then, can make little use of the structure of range images, and treats all range data as simple 3D point sets.

Instead, this project considers the situation of a robot that wishes merely to grasp an object, and has no interest in the range data once this task has been

accomplished. Fewer views will be required for this task, as once the robot has identified a suitable grasp, it does not need to capture any more data. Additionally, lower resolution (*i.e.* larger distances between depth samples) range data will suffice; this takes less time to capture, speeding up the process. The project finds features in a 3D point set that are promising from the point of view of grasping; these include parallel sides, opposable planar patches and concavities. With these features detected, it considers what combinations of features form part of a promising grasp. The best location to take the next range image from is then determined by guessing which view is most likely to contain features that complete the grasp, and another range image is then taken to fill in the region of the missing graspable feature(s). This process is repeated until a good enough grasp is found with the current data.

1.2 Objective and Approach

The project's aim is to develop a system that can gather enough range data to find a reasonable grasp with as few different views as possible. Rather than using a physical range scanner and robot manipulator, it will simulate data capture from previously captured (real) range images that encompass the entire object. This speeds up the testing and evaluation processes and removes reliance on hardware availability. The simulated range sensor has five degrees of freedom: translation in the x, y and z axes, and rotation about the x and z axes.

The project can be split into several stages:

- Simulated range data capture and registration
- Generation and evaluation of possible grasps
- Determining the Best Next View

The first two parts have been researched before, and the system will make use of previous work where possible. The Iterative Closest Point algorithm will be used to register (combine) new data with data that have already been captured. Much work has also been done on evaluating grasps; the system will make use of this where appropriate.

However, Best Next View research has, as has already been mentioned, concentrated on finding the Best Next View for complete model capture. This has considered factors such as the size of occluded areas that will be viewed, and the quality of all the data captured – the nearer the sensor is to being perpendicular to the object being scanned, the better the data. Objects that are scanned at an angle to the sensor have fewer data points than those that are scanned when perpendicular to the sensor, as illustrated in Figure 1.1. As the system only needs

enough data to find a reasonable grasp, high quality data is not needed – finding new data that will combine with current data to provide enough to find a good grasp is the essence of the problem. This reduces the utility of much Best Next View research; however, some useful techniques have been developed that will be used in the system.

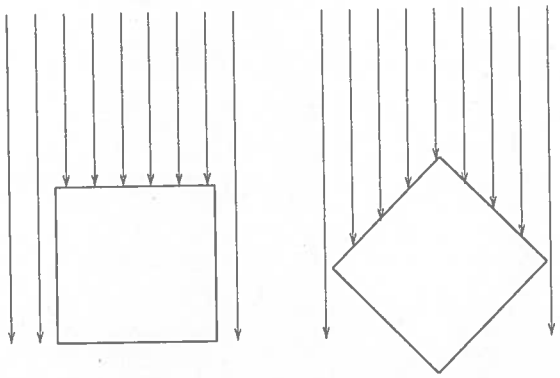

Figure 1.1: Quality of the range data depends on the angle between the object and the scanner head. The object face that is perpendicular to the scanner head yields six data points; the oblique object yields only three or four per face, and fewer if the faces and the laser are more nearly parallel. Additionally, the data obtained from the oblique faces is noisier.

A voting model will be used to decide which view to choose next. Views will be taken from the centre of each cell of an approximated hemisphere surrounding the object; each vote will be for one of the unseen cells of the hemisphere. The cell with the highest number of votes will be the one to be viewed from next.

Initially, the project will consider simple two-fingered grasps; it can then be extended to consider more complicated robot manipulators, such as the Utah/MIT dextrous hand [21]. The nature of a two-fingered grasp means that only parallel or nearly-parallel sides can be considered as a grasping option. A dextrous hand allows many more possibilities, such as a three point force closure grasp, or use of concavities; these allow much more stable grasps than the simple two contact point grasp offered by a parallel jaw gripper.

For two-fingered grasps, the voting will be quite simple: for each grasping point identified, take the dot product of the normal of the plane $\vec{p}$ fitted through it and those of all unseen viewpoints $\vec{v}_i$:

$$q_i = \vec{p} \cdot \vec{v}_i = \vec{p}_x \times \vec{v}_{i_x} + \vec{p}_y \times \vec{v}_{i_y} + \vec{p}_z \times \vec{v}_{i_z} \tag{1.1}$$

The cell that is nearest to facing in the opposite direction to the grasping point is the one most likely to view a useful grasping point (that is, one that is opposed to

the grasping point under consideration), so the grasping point will cast its vote for this cell. The dot product of the unit normal of a graspable patch and the unit normal of the viewing cell is -1 at best, and 1 at worst. -1 indicates that the normals are pointing in diametrically opposed directions, the situation where a parallel patch is most likely to be found; 1 means that the normals are pointing in exactly the same direction, so useful data will not be found when the object is scanned from this view. Additionally, votes could have more or less weight depending on the value of the dot product, or could be discounted if there is little chance of a useful patch being found from its optimal view (*e.g.* if the dot product is greater than -0.5).

More complicated manipulators require a different voting scheme. One possibility for a three fingered grasp would be to choose two grasping points at random and determine which unseen cell of the voting structure is most likely to view the optimal point to complete a three point grasp. The weighting assigned to this vote would be the expected quality of the grasp, if a suitable grasping point were found.

The entire grasp-finding process can be described by the following pseudocode:

1. Choose random position to take initial view from.
2. Capture data.
3. Combine new data with previous data.
4. Find and evaluate a number of possible random grasps.
5. If the best grasp found is good enough, return it.
6. Otherwise, determine the best next view and go to step 2.

1.3 Summary

The development and testing of a system for determining the best next view for a two-fingered robot manipulator is presented in this report. The aim of the system is to gather enough range data of an object to find a good grasp from as few views as possible. Empirical evaluation has shown that the system succeeds in finding a grasp from two or three views in 75% of cases with the test objects, outputting a pair of grasping points that result in a stable grasp. Following further views of the objects, all cases eventually resulted in a good grasp.

2. Data Capture

This chapter describes the equipment used and steps taken to capture and prepare the data for simulated range data capture.

2.1 Resources

A range scanner was used with a rotating bed to obtain the range images used by the system. The rotating bed obviates the need for the scanner head itself to rotate; a rotation of the scanner of a real robot around one axis can be perfectly simulated by rotating the bed through the same angle around the same axis, but in the opposite direction.

2.1.1 Range Scanner

A Reversa laser scanner was used to capture the range data for the system, controlled by `RISCAN` software [1]. It projects a laser stripe onto the object being scanned, and uses two cameras to determine the depth of the stripe at each point. The output is a series of x, y and z values. The scanner has an orthographic projection, and can be translated along all three axes. Range images are built up by repeatedly moving the scanner head all the way along the x axis, and moving it a small amount in the y axis between repetitions. The scanner can capture all possible parts of the object that are not occluded and face the sensor. Figure 2.1 shows that this misses both surfaces that are parallel to the laser, and surfaces that are self occluding or occluded by other objects; only one z value can be recorded for a given (x, y) coordinate. Unfortunately, the scanner cannot determine the depth of surfaces that are very dark; however, this could be used to the system's advantage to remove unwanted data points, such as the bed, that were not part of the object being scanned. These points simply do not show up in the range data (an infinite depth is recorded), so by simply covering external objects with black paper, they become invisible to the scanner. In a real physical system, if there were any chance of the robot observing itself, making it black would prevent this from causing any problems.

Unfortunately, there are other situations in which the range scanner cannot determine a depth value. If the laser stripe cannot be seen with both cameras, no value is returned, leaving blank patches in the range data. This typically occurs with surfaces that are almost vertical; the problem can be solved by rotating the object being scanned to allow both cameras to see the object.

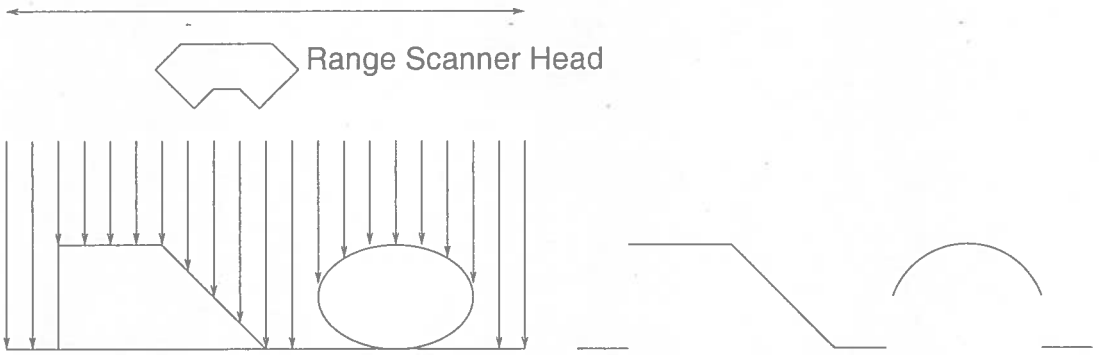

Figure 2.1: The scanner moves horizontally in this diagram, projecting its laser vertically down. The image on the right shows what the scanner captures.

This scanner can perceive objects approximately 10cm away from the scanner head. As some of the objects went outside this range, particularly when rotated, several view volumes had to be built up by performing several scans at different depths and concatenating the points found. Unfortunately, in certain positions, such as when viewing from around the horizon (*i.e.* at an elevation of 0°), the rotating bed did not allow the scanner head to approach near enough to capture all of the object. This was accepted as a limitation, as the missing parts of the object would be seen by other views. An alternative to this would have been to attempt to reconstruct the data from other views; this was rejected, as a real robot would have to deal with this problem and would not know what the object was or what it 'should' be seeing.

2.1.2 Rotating Bed

The bed used was capable of rotating through any angle around one axis in the xy plane (depending on orientation); it was set up to rotate around the x axis. A rotating plate was used to provide rotation around the z axis (see Figures 2.2 and 2.3). Although the surface normal of the rotating plate would change as the bed was rotated, it remained constant with respect to the object being scanned, and the data could easily be transformed back to represent the object in its initial orientation.

Errata

Figure 2.5, page 9: for 'range ima ge', read 'range image'.
Section 4.2.2, page 26: disregard 'although this was' at end of section.
Section 5.3.2, page 31: after 'more surface area', read 'is in contact with the gripper'.
Section 5.3.4, page 33: for 'distance from the lines is desired', read 'distance from the line is desired'.
Section 6.2, page 41: for 'divide its number of views by $n + 1$', read 'divide its number of votes by $n + 1$'.
Section 6.2, page 41: for 'acos(d)¡45', read 'acos(d) $< 45°$'.
Section 7.1, page 53: for 'foce closure', read 'force closure'.
Section 7.1, page 53: for 'closer then 5mm', read 'closer than 5mm'.
Section 7.1, page 53: for 'rotating However', read 'rotating. However'.
Section 7.1, page 54: for 'Table 5.3', read 'Table B.3'.

Tables 6.1 + 6.3 ?

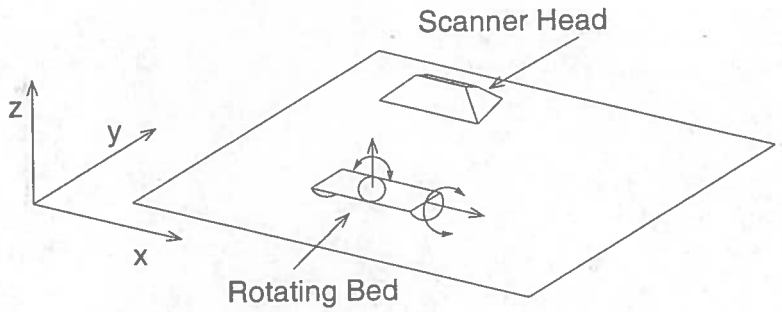

Figure 2.2: The bed can be rotated about the x axis, passing roughly through the base of the object being scanned. The rotating plate allows the object to be rotated about its local z axis.

2.2 Test Data Acquisition

2.2.1 Scanning

Several objects were scanned, to give a range to test the system with. They ranged from a simple wooden block to a small toy cow. It was decided to use steps of 45° for both the azimuth and the elevation (see Figure 2.4 for a definition of azimuth and elevation), resulting in 17 views for each object (eight from an elevation of 0°, eight from an elevation of 45°, and one looking straight down from an elevation of 90° – the azimuth angle does not matter at this elevation; 0° was used). This converts the hemisphere of points from which the object can be viewed from a continuous representation into a set of discrete cells. This simple method of discretising a hemisphere was suggested by Horn [13] and Connolly [8].

The objects were scanned with 1mm between samples in the x and y directions, which was found to be adequate for this task, though model acquisition typically requires a higher resolution (*e.g.* 0.1mm between samples). The range scanner finds depth values with great accuracy; the error has been found experimentally to be approximately 15 μm.

2.2.2 Extraneous Data Removal

Areas of the scene such as the rotating bed – the floor for a real robot – must be removed before any analysis of the data can proceed. For a real robot this is a trivial problem – the data would simply be thresholded, and data points with a depth value below a certain threshold (the z value of the floor) would be discarded as floor data. For this project, data was examined using the `RISCAN` software and points corresponding to the bed were removed manually. The data

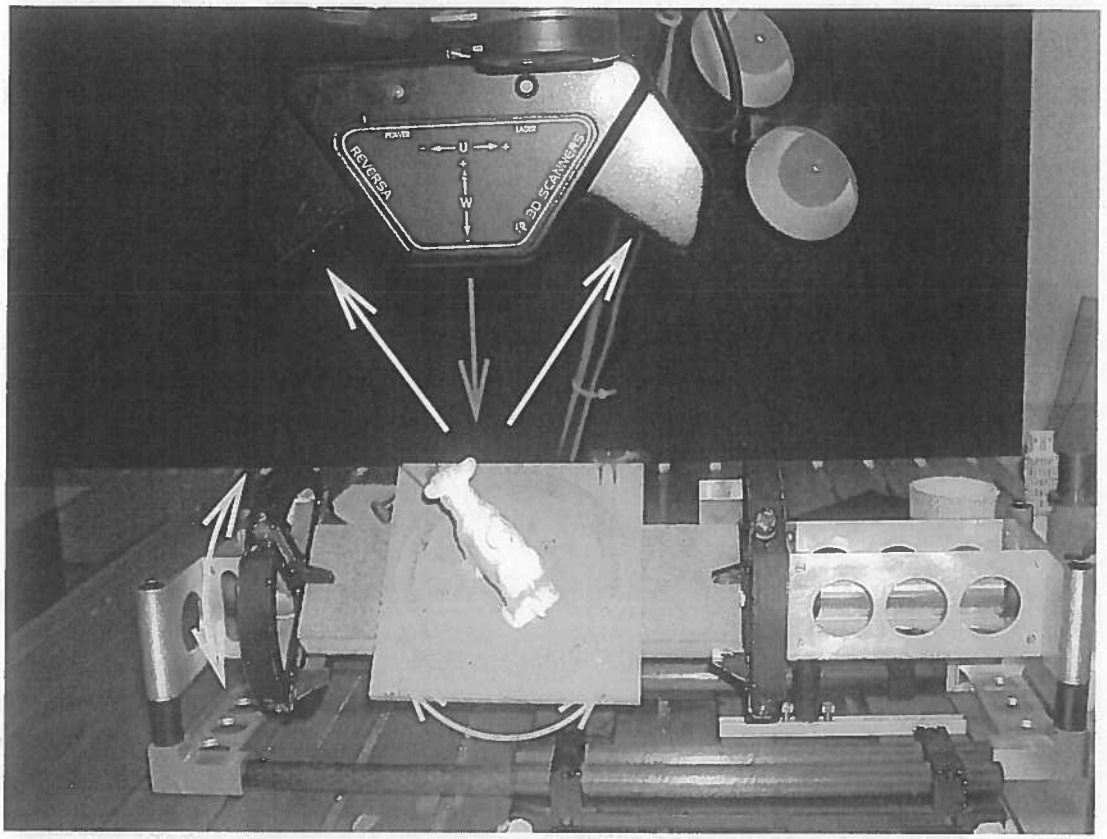

Figure 2.3: This picture shows the experimental setup for gathering range data. The range scanner's laser is emitted in the direction of the red arrow; its two cameras observe the object from the directions of the white arrows. The rotating bed can revolve the object roughly around its x axis, indicated by the yellow arrow; while the object can be rotated about its z axis by the revolving plate. The cow on the platform was used as one of the data sets.

was not cleaned up too perfectly, though, as a real robot would have to deal with some uncertainty and would not have a human available to aid it.

2.3 Results

17 views were taken of each object with 45° steps for both the azimuth and elevation (see Figure 6.1). Table 2.1 summarises the data captured; Figure 2.5 shows a typical range image from the Stuttgart Range Image Database [24].

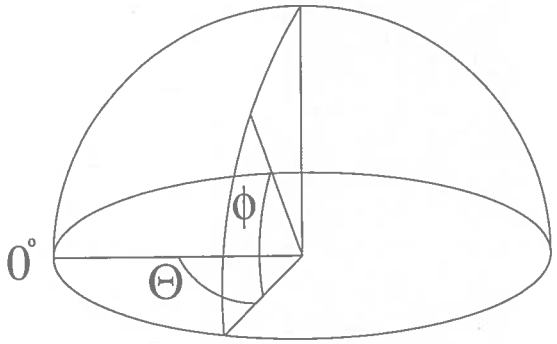

Figure 2.4: The azimuth is the angle θ around the horizon from the zero point, from 0° to 360°. It corresponds to rotation around the z axis. The elevation is the angle ϕ above the xy plane, from 0° to 90°.

Object	Min size	Max size	Mean size
Cow	915	2045	1652
Stone	83	784	410
Wooden Block	1886	4054	3159

Table 2.1: Number of points captured for each test object.

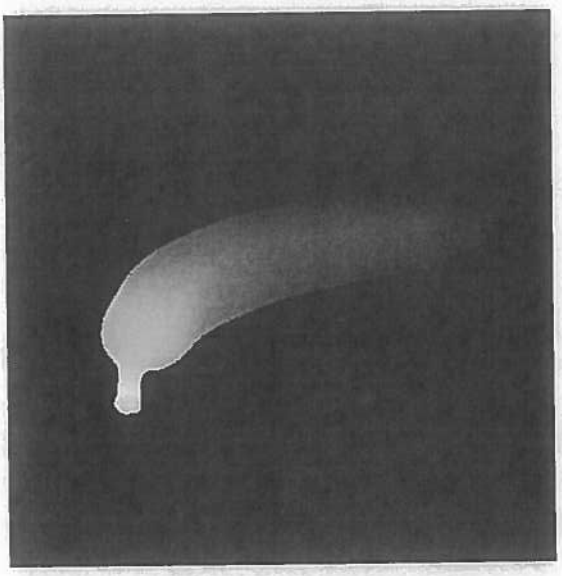

Figure 2.5: A range image of a banana.

3. Range Data Registration and Fusion

Range data registration is the process of taking two views of a scene, taken from different positions and/or orientations, and finding a transformation that maps corresponding points from one set of data onto another. This allows the system to consider all of the data that it has collected so far as a single data set. The Iterative Closest Point (ICP) algorithm was used to register the data. This requires an approximate transformation to match the two data sets into roughly the same position, then uses a least-squares approximation to get the best possible match between the two data sets. Once the two data sets are aligned, fusion is carried out; this is the process of converting the two aligned data sets into one. Ultimately, a single data set that covers both input data sets is returned.

3.1 Transformation

A reasonably accurate initial transformation is essential for the ICP algorithm to successfully register range data. This was found manually, although it would be simple for a calibrated robot equipped with a range scanner to carry it out automatically. The reasons that this process was carried out manually were that the exact origin of rotation about the z axis was not known, and the axis of rotation about the x axis did not intersect the plane of the rotating plate, meaning that the scanned objects were rotated about an axis slightly below their bases. A real robot would not move the object, but move its range scanner head to known positions, making the initial estimates of the data sets' positions easy to find.

The point sets were all transformed to a standard position prior to running the ICP algorithm. The 90° elevation, 0° azimuth position (*i.e.* looking straight down on top of the object) was chosen as the standard position; all views not taken from this position were transformed, then aligned with this data set by hand to prepare them for use with ICP.

The first step in transforming the data to the standard orientation (rotation) was to translate the data in each set to its centroid (the centre of mass of the object, found by taking the mean x, y and z values of its points), in order to keep the data in roughly the same position once it had been rotated. This involves simply finding the mean x, y and z values of the data points, and subtracting them from each point. The data was then rotated about the x and/or z axes by

the negation of the angle that the object had been rotated through during the scanning process. Next, the data was translated back to its initial position by adding the mean values back on. A short Matlab script was written to carry out this task.

With the data in the standard orientation, the next step was to translate each point set to align it with the standard position. The tool used was `handalign`, by Neil McCormick, formerly of the University of Edinburgh Machine Vision Unit. This allows two or more point sets to be loaded, each of which can be independently rotated and translated to align it with another point set. The reason that the data was rotated independently of this program was that it can be quite unwieldy for rotation, and the angles that the objects had been rotated by were known quite precisely. Thus it was easier and less error-prone to rotate the objects separately, and `handalign` was used for translation, which it handles well, and any minor rotations that inaccuracies in setting the rotating bed introduced. The output from `handalign` is a 4×4 transformation matrix that can be multiplied by a $4 \times n$ matrix of points to effect the complete transformation to the standard position and orientation; this was used in the main Matlab program to transform all loaded data to the standard position, ready for the ICP algorithm.

3.2 Iterative Closest Point Algorithm

3.2.1 Literature Review

The ICP algorithm was introduced by Besl and McKay in 1992 [3]. The original algorithm took two point sets P and Q, where P was a subset of Q, and an approximate transformation between them, outputting a precise transformation to map P on to Q. The following steps are repeated until the calculated transformation converges:

1. *Find nearest point:* For each point $\vec{p}_i$ in P, find the nearest point $\vec{q}_j$ in Q.
2. *Compute registration:* Find transformation T that minimises the sum of the squared distances between all pairs of closest points $\vec{p}_i$ and $\vec{q}_j$.
3. *Perform transformation:* Apply transformation T to P.

These steps are repeated until the difference in total squared distance error between two iterations falls below a threshold. The algorithm converges quickly to the nearest local minimum to the initial position of P.

The squared distance error can be defined in a number of ways. Typically, the Euclidean distance between two points is used; however, other functions can be

used – Chen and Medioni [6] minimised the squared difference in the surface normal direction. The Euclidean distance was used in this project for its simplicity and good performance.

Much work has been done to extend and improve this algorithm. These improvements have included: faster rates of convergence; different cost functions (as noted above); and the ability to register point sets which are partially overlapping, but that each have unique points not present in the other set. This last improvement is of great importance to this project, as it relies on being able to register a point set with another that has some, but not all, points in common. It has been implemented in different ways by many researchers, including Chen and Medioni, Zhang [27], and Turk and Levoy [25]. Turk and Levoy and Chen and Medioni both also proposed methods for pairwise registration of more than two sets of points; this is also an important part of this project, in case more than two views of an object are necessary to find a good grasp.

It has been noted that pairwise registration of several overlapping point sets into a common coordinate frame does not necessarily provide optimal registration – registering two sets A and B, followed by the registration of another set C with A, will minimise the squared distance between sets A and B, and between sets A and C, but not necessarily between sets B and C. Several papers, including Bergevin *et al.*'s, [2], Eggert *et al.*'s [9] and Blais and Levine's [4], have proposed an extended version of the ICP algorithm that considers many range images simultaneously, and minimises the sum of the squared distances for all views. This approach solves this problem in registering multiple range images; however, in practice, registering one pair of images at a time was not found to be a significant problem in this project, due to the low expectations made of the range data, and made the implementation simpler. Additionally, Blais and Levine highlight the fact that this problem is most severe when the first and last views of a complete object to be registered are compared; as the system aims to stop after just a few views, the issue becomes even less of a problem.

The ICP algorithm has been found to be very effective, given a reasonable initial registration. However, it can have problems with regular solids, such as cubes and cylinders. For example, two views of a cube, taken from different angles, may only have points on one face in common. These points may 'slide' along the face of the cube, and any position where some of the common points overlap will succeed (see Figure 3.1). Similarly, a cylinder could be rotated through any angle about its axis. However, with a good initial estimate of the registration, the transformation will 'lock in' to a suitable local minimum of squared distance error. Objects that have not been precision engineered do not typically exhibit this behaviour, as there are enough imprecisions (*e.g.* small surface details) in the data to match data sets precisely. Brujic and Ristic [5] empirically found this to be true using a Monte Carlo simulation, and also that registration error

converges to zero as the number of points increases. 'Sliding' was not found to be a problem with the system, as the initial registration estimates were sufficiently accurate.

Another problem that was encountered was in the case that two data sets share no points, *e.g.* data sets from diametrically opposing views located on the horizon. Such views will almost certainly share no data points. In this case, the system carried out the next view decision process again, but excluded the previously chosen view from the process.

Similarly, two data sets that share just a few points were also treated as being entirely distinct. This is because a few outliers in both sets that happened to be near each other could cause the sets to be incorrectly registered. The transformation mapping one group of outliers onto another would be unlikely to represent the true transformation mapping one set's points on to the corresponding points in the other set. As the system dealt with data sets ranging from about 350 to 4000 points, a simple single value threshold was not suitable. Instead, the number of points in the data set to be registered was divided by 30, and this number was used as a minimum number of corresponding points, without which the data set was rejected as being distinct.

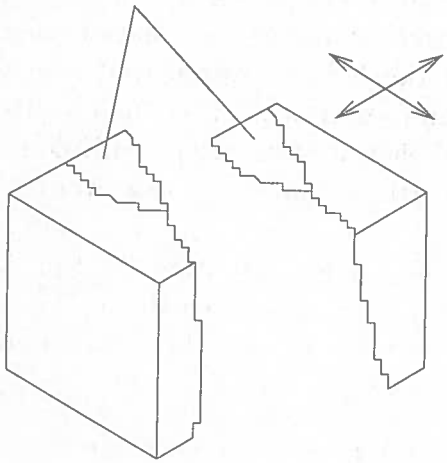

Figure 3.1: The two 3D point sets, sharing the indicated areas, can be translated in the directions shown. Any translation could be accepted by ICP, provided that some points overlap.

3.3 Fusion

Range data fusion, or integration, is the process of taking two sets of range data that have been registered and producing a single set that is composed of both

input sets, but does not contain any redundant points. No vastly superior method for carrying this task out has yet been found, but several have been developed using different methods. Those applicable to 3D point sets like those used in the project include: measuring the distance from a point to a surface consisting of the k nearest points and updating it to fit the surface (Masuda [15], Hoppe *et al.* [12]); and using Venn diagrams of overlapping surfaces to decompose the data into subsets, followed by constructing a canonical grid (with vertices x_i and y_i) and computing the average z value of all overlapping patches at all values of x and y (Soucy and Laurendeau [23]).

As this is not the focus of the project, a far simpler approach was taken. The ICP code returns the registered data set, as well as a list of corresponding points found in both the input newly-registered data set and the data set it was registered with. The fusion system simply removes the corresponding points from the registered data set, and concatentates the new subset of unique points with the data set with which these points are now registered, *i.e.* the new set $R = Q \cup TP$. As previously mentioned, a more sophisticated method is essential when many views are to be registered with great accuracy; however, for the purposes of this project, this uncomplicated approach suffices.

One slight complication was that each point had the angle from which it was captured associated with it. This information was used in the grasping point detection stage to determine the directions of the surface normals of grasping patches – all normals should point away from the object, *i.e.* towards the range scanner head. For each fitted plane equation $\vec{p}$, found in a data set with an associated viewing angle $\vec{v}$, $\vec{p} \cdot \vec{v}$ should be positive; if it is not, the plane normal is pointing the wrong way, and the plane parameters are negated to fix this problem.

This raises the issue of what should be done when points from two different data sets (that have different viewing angles) are fused. The approach decided upon was to sum the viewing angles for fused points. A simple average was not used, as fusion of a fused point with a third data point would skew the viewing angle in favour of the third point – it would have 50% influence, while the first two points would each have 25% influence over the viewing angle, as shown in Figure 3.2. Simply summing the viewing angles keeps the fused viewing angle balanced. It is not necessary to divide the viewing angle by its magnitude (to normalise it), as only the sign of the calculation $\vec{p} \cdot \vec{v}$ is important; a magnitude of more than 1 does not affect this.

The reason why the original viewing angle is not simply kept is that a point could conceivably be fused with another point that had a very different viewing angle (*e.g.* a point along the edge of a cube). If this point is then incorporated into a patch whose points were seen from a different angle, it could have an effect on the average viewing direction of the points. This determines the direction of the

plane normal, so failure to rectify the point's viewing angle could cause a patch's normal to point into the object instead of out of it.

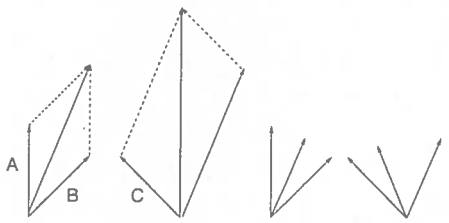

Figure 3.2: On the left, when the viewing angles A and B are fused, a vector equivalent to twice their average is the result. When C is also fused, the result is a vector whose direction is the average of all three viewing angles. On the right, however, fusing A and B simply takes their average. When this average is fused with C, a vector biased in favour of C is the result.

3.4 Implementation

Because developing an ICP implementation was not within the scope of the project, a publicly available implementation of ICP in Matlab was used [17]. This code accepts two sets of points in approximate registration and a tolerance value for specifying how far apart two points can be, while still representing the same point on the object. A value of 1mm was chosen for this, as this was the resolution of the range data; larger values would allow unrelated points to be treated as being the same point, while smaller values could prevent related points from being matched. It returns the rotation and translation matrices found, a list of indices of the corresponding points found amongst the two data sets, and the second data set transformed to be registered with the first. A small amount of glue code was written to interface this with the rest of the system, as well as a simple method for fusing the two data sets (described above).

The method used by the algorithm for registering a data set P that was not a subset of the other set Q was as follows:

1. Find nearest point $\vec{q}_j$ in Q to every point $\vec{p}_i$ in P
2. If the Euclidean distance from $\vec{p}_i$ to $\vec{q}_j$ is greater than the tolerance, ignore $\vec{p}_i$. The remaining points are the non-unique points that are present in both P and Q.
3. Find the transformation T that minimises the squared distance between corresponding points.

4. Apply T to all points in P, including those that have no corresponding point in Q.

As previously mentioned, it is possible for two data sets to have no or too few points in common with the given tolerance value. When this occurred, the system carried out the voting for the next view again, but excluded the view that had previously been chosen from selection. This was repeated until a view was found that could be registered.

3.5 Results

The tables in the Appendix show, for each object, the number of points in each view, the mean distance error when two views were registered, and the number of points returned when two views were fused. A dash in the table indicates either that the views are the same, or the pair of views could not be registered, as they did not share enough data points.

The approximate dimensions of the objects are:

- Cow: $76 \times 26 \times 46$mm
- Wooden block: $67 \times 44 \times 43$mm
- Stone: $46 \times 26 \times 16$mm

The maximum errors for each object were:

- Cow: 0.6717mm
- Wooden block: 0.6455mm
- Stone: 0.6774mm

3.6 Evaluation

The implementation was found to work well for some pairs of views, typically registering and fusing two sets of roughly 1800 and 2050 points, of which 550 were shared, in about a second and a half on a 2.6GHz Pentium 4 DICE PC with 512MB of RAM. (The same specification of PC was used for all tests.) The error was of the order of magnitude of approximately one hundredth of the dimensions of the objects; this was considered to be more than accurate enough for the purposes of the system. Data sets that shared too few points were rejected, so pairs of data sets with this property, such as two faces of a cube that only shared

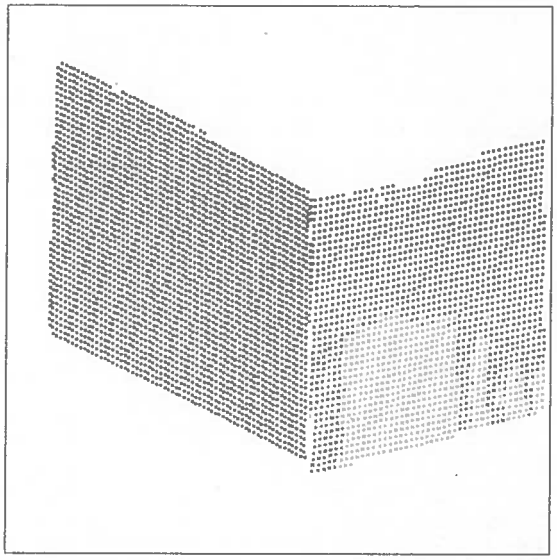

Figure 3.3: These two data sets cannot be registered because they share too few points.

an edge, were not registered. Figure 3.3 shows this situation. Figure 3.4 hows how two sets of range data are registered and fused.

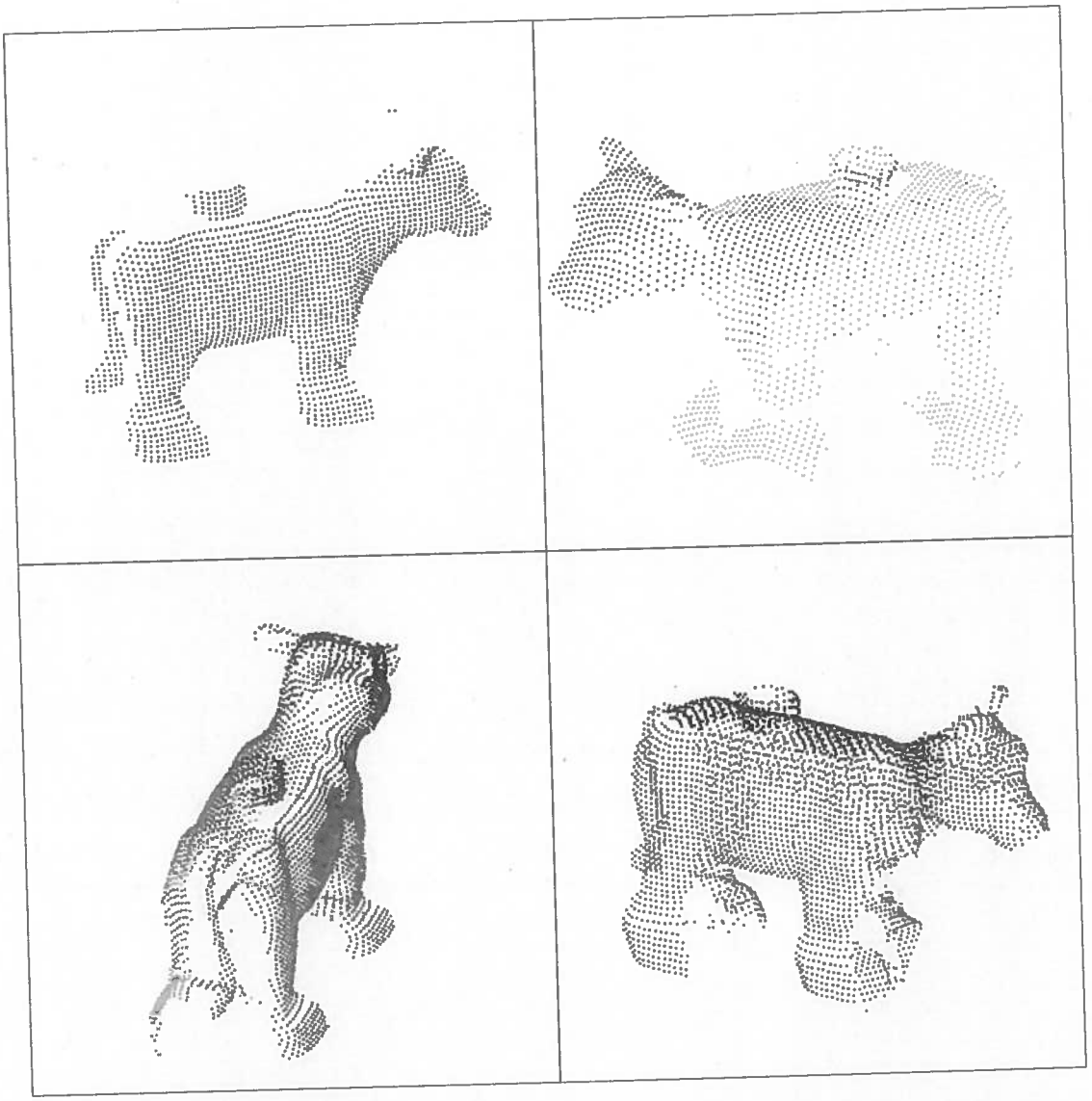

Figure 3.4: The image at the top left was taken from the 0° azimuth, 0° elevation position; the image at the top right was taken from the 135° azimuth, 45° elevation position. The bottom left image shows the two images in approximate registration, having been rotated and translated. The bottom right image shows the final data set once the two views have been registered and fused.

4. Grasping Point Detection

The system treats any near-planar set of points over a certain circular area as a potential grasping point, provided that the plane consists of a reasonably large number of points. Plane equations centred on random points are found using a least squared distance solution; the normals and centres of mass of suitable planar patches are used to evaluate potential grasps. This part of the system aims to find as many good graspable patches as possible, and to determine the quality of the planes fitted through their centres.

4.1 Algorithm

Two methods were tried for this process. The first was as follows:

1. Choose a random data point.
2. Find the number of neighbouring points within a specified distance of the data point.
3. If there are too few nearby points, try again with another data point.
4. Otherwise, fit a plane to the nearby points. Ensure that the plane normal is pointing in the correct direction.
5. If the plane is a poor fit, try again with another data point.
6. Otherwise, remove the points from the data set and save the plane's equation and centre of mass.
7. Repeat until there are no more points, or the number of consecutive failures has exceeded a specified threshold.

This method was found to work fairly well, although it exhibited some behaviours which could be regarded as flaws, namely: it often fails to find all possible grasping patches, due to the maximum number of failures occurring before all grasping patches have been selected (see Figure 4.1); and many possible grasping points are eliminated because another one overlaps it (see Figure 4.2).

The first issue was the main problem with this algorithm, as it was hard to determine a suitably high threshold for the maximum number of failures in a row. This was attempted by setting the threshold to a supposedly unrealistically high value (10000), and recording the largest number of failures in a row (excluding the threshold), resulting in a very long runtime. The maximum number of consecutive failures reached around 1150 for data sets ranging in size from about 900 to 2000

points, and took up to ninety seconds to terminate. It was therefore decided simply to check all points once each (removing points that were found to be part of a plane, regardless of whether or not they had been checked), as the reason the first method failed was because the same points were being checked many times. Checking every point was found not to be intractable, taking from eight to 19 seconds to find from 68 to 150 patches in data sets containing between 900 and 2000 points.

The second algorithm was identical, except for the first and seventh steps:

1. Find the first point that has not already been tried and set it to tried.

7. Repeat until all remaining points have been tried.

Although this method does not necessarily maximise the number or quality of patches, the set of rejected points will not contain any patches; it represents an improvement over the first method in both speed and number of patches found.

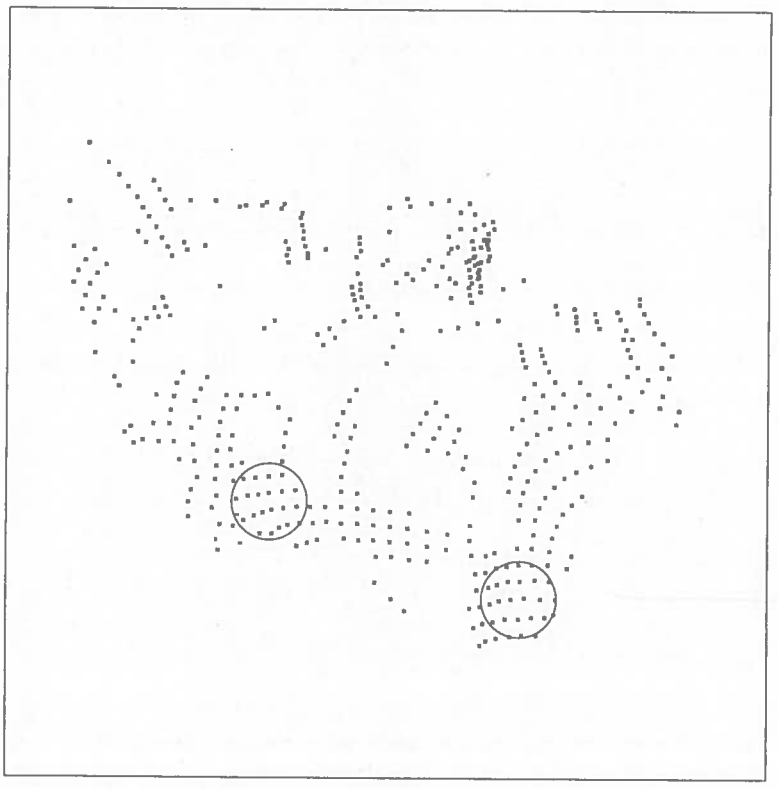

Figure 4.1: This is an example set of points rejected by the first point finding algorithm, where the maximum consecutive failures threshold was set too low. As can be seen, not all possible grasping patches were found; at least two reasonable, small, roughly planar regions are in this set.

The second issue was also found to be relatively unimportant (for both methods), and in fact beneficial in some ways. As can be seen in Figure 4.2, the removal of the points constituting a found patch prevents unfound patches from making use of these points; only a few of the points that they need remain. However, this works to the system's advantage in certain respects. Were a patch's points to remain under consideration after the patch was found, the patch could be found again and again (with the first method, though not with the second), wasting time. The patches that would otherwise remain unfound could be considered for grasping; however, at the scale of the robot's gripper, a grasping point just a millimetre away from another is little different. Additionally, the vastly greater quantity of grasping points to be considered would involve much more computation, slowing the system down in order to compare grasps that were almost identical. The relatively small number of grasping points generated allows all possible grasps to be evaluated within a reasonable amount of time.

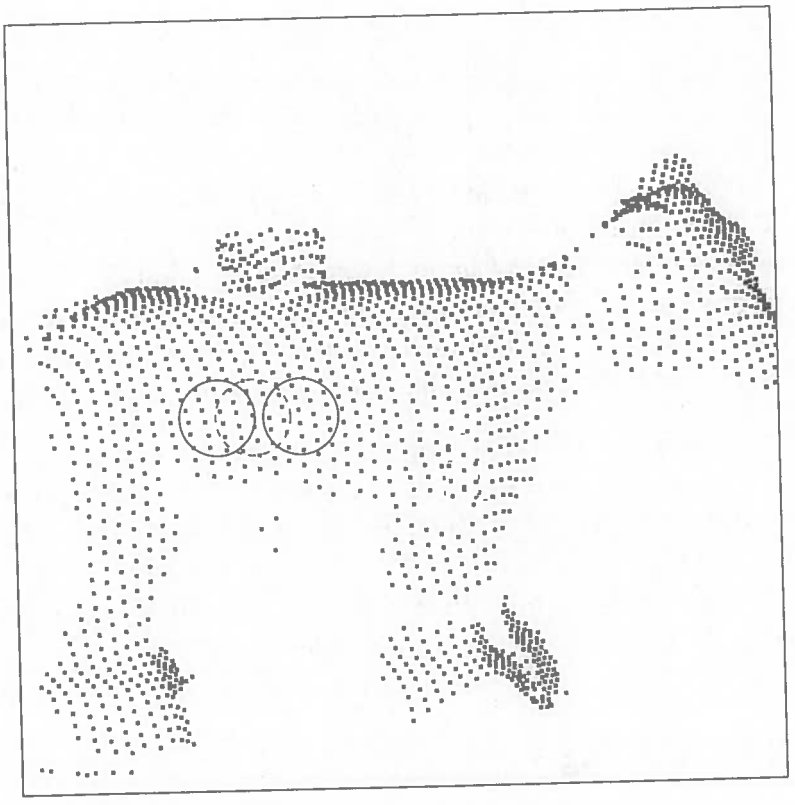

Figure 4.2: Finding either of the two patches outlined with a solid circle, or any other overlapping patches, precludes the possibility of finding the one outlined with a dashed line, or any others that are overlapped.

4.2 Implementation

This part of the system can be divided into three parts: finding candidate grasping patches; fitting planes through candidate patches' points; and determining the quality of the planes.

4.2.1 Finding Dense Patches

With a point chosen, the Euclidean distances from this point $\vec{x}_p$ to all other points in the data set $\vec{x}_i$ were found with the following equation:

$$d_i = \sqrt{(\vec{x}_p - \vec{x}_i)^2 + (\vec{y}_p - \vec{y}_i)^2 + (\vec{z}_p - \vec{z}_i)^2} \tag{4.1}$$

The gripper contact was assumed to be a circle of radius 2.5mm (though this could easily be modified) giving a contact area of $\pi \times 2.5^2 \approx 20\text{mm}^2$. All points $\vec{x}_i$ with $d_i < 2.5$ were incorporated into the patch. As the scanning resolution was 1mm between points in both the x and y axes, the patch can consist of up to roughly 20 points. A good grasping patch, however, may be on the side of an object that was not scanned perpendicular to the range scanner's head; as noted in Section 1.2, it will have fewer data points in the x and/or y axes. Additionally, dark spots on the object will not be picked up, so perfectly good graspable patches may consist of many fewer than 20 points. Patches were found that were visually good, but that were not perpendicular to the scanner head when scanned, resulting in as few as 9 points. For these reasons, 9 was used as the lower limit for the number of points in a graspable patch.

Good grasping patches that were scanned at an extremely low resolution from a certain view, and so have fewer then 9 points, can be 'filled in' from another view that scanned the area at a more perpendicular angle (see Figure 4.3). As the grasping patches are 're-discovered' in every iteration, patches that were discarded in a previous iteration can be used when range data from another view has filled them in.

4.2.2 Plane Fitting

With a reasonably dense patch, the next step was to fit a plane through its points. A plane fitted through a set of points can be defined by the equation $\vec{p} = Ax + By + Cz + D = 0$. The distance d_i from one of N homogeneous points $\vec{v}_i = (x_i, y_i, z_i, 1)$ to a plane can be calculated by plugging the point's x, y and z values into the plane's equation:

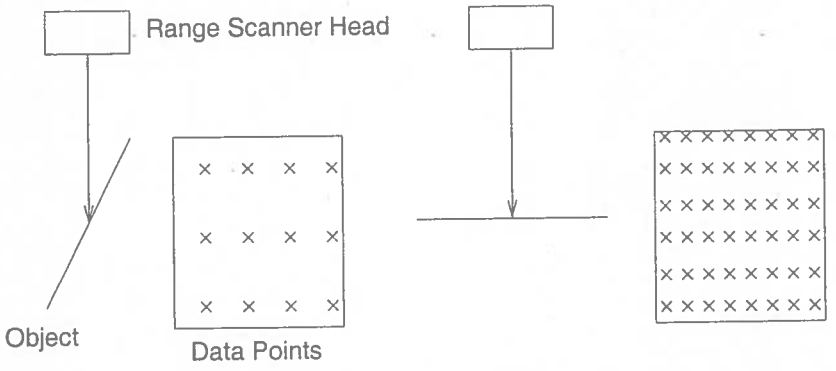

Figure 4.3: If a patch is scanned when it is almost parallel to the direction of the range scanner's laser, few points are recorded. The patch can be 'filled in' by registering it with data taken from a better view.

$$d_i = Ax_i + By_i + Cz_i + D \tag{4.2}$$

assuming that the plane is normalised, *i.e.* $A^2 + B^2 + C^2 = 1$.

The best-fitting plane through a set of points minimises Q, the sum of the squares of the distances from each point to the plane, *i.e.* $Q = \Sigma_{i=1}^{N} d_i^2$. This can be written as:

$$Q = \sum_{i=1}^{N} (\vec{v}_i^T \vec{p})^2 = \sum_{i=1}^{N} \vec{p}^T \vec{v}_i \vec{v}_i^T \vec{p} = \vec{p}^T (\sum_{i=1}^{N} \vec{v}_i \vec{v}_i^T) \vec{p} = \vec{p}^T M \vec{p} \tag{4.3}$$

The plane equation can be found using Singular Value Decomposition (SVD). SVD decomposes any $m \times n$ matrix, where $m \geq n$, into the form UDV^T, where U is an $m \times n$ matrix, V is an $n \times n$ matrix, and D is an $n \times n$ diagonal matrix [16]. V contains the eigenvectors of M, and D their associated eigenvalues. In this case, the three eigenvectors represent the best, an intermediate and the worst plane equations through the points in M (in terms of squared error), in increasing order of the associated eigenvalue.

It is essential for both grasp evaluation and next-view voting that the normals of the planes point away from the object. As the view that the object was scanned from was included with each point, this can easily be checked. First, the average normal of the plane's points is calculated. It is quite possible that the plane consists of points taken from different views, or that the points have been fused from two or more points taken from different views, so an average view angle is essential. Once this has been found, the dot product of the plane normal and the average view angle is found. If this is negative, the plane normal is facing the wrong way. This is rectified by negating the plane's parameters.

The plane finding code was substantially based on the code from the Advanced Vision module at the University of Edinburgh [10], with some task specific modifications. With a typical data set of around 2500 points, this step took about ninety seconds to complete execution, ~~although this was~~ .

4.2.3 Plane Quality

Of course, the best plane through a set of points may not be a particularly good fit; if the points are not approximately coplanar, there simply is no good plane. The quality of a plane equation found with this method can be found from the eigenvector, which is related to the average squared distance from the points to the plane. Initially, the eigenvalue threshold was set quite high (0.6), resulting in many grasping points being found. However, it was later found that this lenient policy caused problems when determining which view each grasping point voted for; the inaccurate plane normals caused inappropriate views to be voted for. By setting the threshold to a lower value (0.2), fewer, but higher quality, planes were found. This helped with the voting process and ensured that better decisions were made on which view should be captured next.

Because the eigenvalue only represents an average of the points' fits to the plane, it is possible that, despite a low average, one or more points are present that render the plane useless for grasping. For example, a single point representing a spike could make it impossible to use the patch as a grasping point (see Figure 4.4). To detect and eliminate planes like this, the distance from every point in a grasp to the plane ~~was~~ calculated using Equation 4.2 was found. If any point was more than one millimetre away from the plane, the plane was rejected.

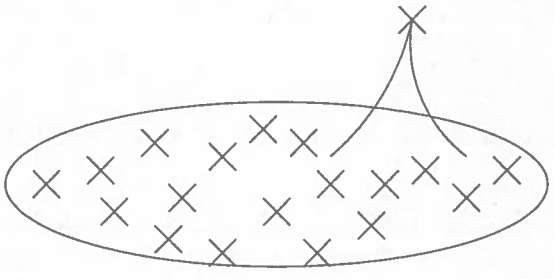

Figure 4.4: Although the high quality of the points on the plane ensure a low eigenvalue, the spike makes the point unusable. The plane can be ruled out when the distance from the point on the spike to the plane is found to be too large.

4.3 Results

Table 4.1 shows what grasps were found from which views for each object.

Azimuth/°	Elevation/°	Cow	Wooden block	Stone
0	0	92	228	57
0	45	150	367	40
0	90	149	248	35
45	45	134	293	48
45	90	139	300	11
90	45	68	265	49
90	90	70	151	4
135	45	133	260	57
135	90	139	304	11
180	45	150	303	48
180	90	149	225	32
225	45	126	254	41
225	90	114	288	23
270	45	107	294	22
270	90	64	148	12
315	45	127	318	22
315	90	114	306	13

Table 4.1: Number of grasping points found from each view for each object.

4.4 Evaluation

The algorithm may not maximise the number or quality of grasping points, and is not particularly efficient; indeed, apart from the range data capture, finding grasping points is the bottleneck in the system. However, it was found to work well in practice, and successfully detected suitable grasping points, while rejecting plane fits that were problematic for the voting process, and were too non-planar for a robot manipulator. The set of rejected points returned by the second version of the algorithm is guaranteed not to contain possible grasping points, unlike the initial version; once the algorithm had finished running, the only points that had not been incorporated in grasping patches either formed non-planar patches, or were the left-over points in between detected grasping patches.

5. Grasp Generation and Evaluation

With graspable points determined, the next step for the system is to generate and test possible grasps. Grasp generation is simple; evaluating a grasp to differentiate between the quality of different grasps is more involved.

5.1 Two Finger Grasps

As the number of grasping patches the system generates is quite small, it is not unfeasible or intractable to evaluate every single combination of two grasping patches. This generates $n^2 - n$ grasps over n grasping points. Each grasp can then be evaluated for possibility – whether or not the grasp could actually be used to pick the object up – and quality – how good the grasp is compared to others. It is assumed that the two fingers are dextrous (*i.e.* they have six degrees of freedom and can be applied at any position and orientation) rather than being a simple parallel jaw gripper, whose two jaws must be directly opposite and parallel to each other.

5.2 Grasp Possibility

5.2.1 Physically Impossible Grasps

The robot's manipulator will have a maximum distance that it that can span; it may also have a minimum distance. These are clearly the first properties to check of a grasp, as they are simple to calculate, and failure will minimise wasted computation time. The length of the *grasp axis* connecting the two grasping points is compared to minimum and maximum values.

It may also be that the object obstructs the robot's manipulators; this is *path planning*, as opposed to grasp planning. As the system is not designed for a particular robot, this check is not carried out; however, it would be simple to incorporate it into the system and fail grasps that do not meet this condition.

5.2.2 Force Closure

The force closure property determines whether or not a grasp will hold an object in equilibrium. It is a binary condition; as such, further measures must be made to determine the quality of the grasp.

For a two fingered grasp, the force closure property [19] can be determined using *friction cones*. These are imaginary cones that have the apex at the grasping point and the vertex in the opposite direction to the grasping point normal; that is, they point into the object. A friction cone is determined by the angle α between its vertex and surface. This is derived from the coefficient of friction μ between the gripper and the object: $\alpha = atan(\mu)$. For the force closure property to hold, the grasp axis must lie inside both cones. This can be checked for each grasping point by taking the dot product of the normalised grasp axis and the plane normal at the grasping point, giving the cosine of the angle between them. Subtracting this angle from 180 gives the angle ϕ_i between the grasp axis and the vertex of the friction cone; if this is less than α_i for both grasping points, the grasp is force closure (see Figure 5.1). Non-force closure grasps can be discarded immediately; those that satisfy this property are evaluated for grasp quality.

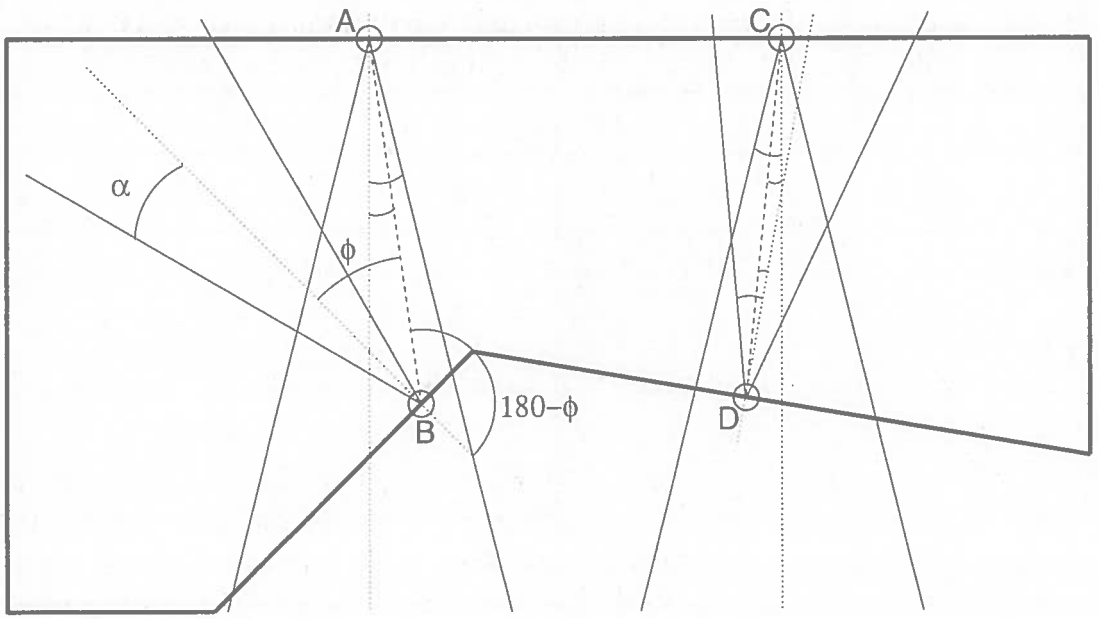

Figure 5.1: The grasp on the left (A+B) is not force closure; the dashed line connecting the grasping points only falls inside only one of the friction cones. The right hand grasp (C+D) is force closure; ϕ_i is smaller than α_i for both friction cones.

5.3 Grasp Quality

Of the many grasps that the system evaluates, it is possible that at least several will exhibit the force closure property once enough data has been gathered. The next step is to decide which of these grasps is the best. Desirable qualities for a grasp include stability and robustness to positioning errors and noise.

5.3.1 Literature Review

Two-fingered grasping has not come under such scrutiny since dextrous hands have become widespread; however, there is still some recent literature dealing with this problem. Smith *et al.* [22] examine parallel jaw grasps on two dimensional slices through the centre of mass of a polygonal three dimensional object. Five criteria are considered, although three of these are binary conditions that must be satisfied, or the grasp will not be considered. These are: minimum distance from corners of grasp points; the force closure property; and accessibility for the grippers. The two quality criteria are dependence on friction and torque required to prevent the object from rotating. Both of these criteria were included in the system developed in the project.

Montana [18] uses some other properties to evaluate grasp quality. These include: object curvature, gripper shape, and gripper and object viscoelasticity ('softness' or 'squishiness'). Curvature makes use of the fact that the more concave a surface is, the better it is for gripping (within limits), and that the nearer a convex surface is to being planar, the more surface area it has in contact with the gripper. However, these properties were not implemented, as the gripper properties were assumed to be constant, the object viscoelasticity was unknown, and calculating object curvature was considered to be beyond the scope of the project.

5.3.2 Implementation

As determining grasp quality is not the focus of this project, only a simple (though reasonably effective) method for determining grasp quality was used, though it would be simple to 'plug in' a more sophisticated method. It consists of two criteria, which are merged to produce a single value for grasp quality. These are dependence on friction and distance from the grasp axis to the centre of mass of the object.

The dependence on friction can be found by calculating how near the grasp axis is to the limits of each of the friction cones. All other factors being equal, a grasp that uses two parallel, opposed grasping points (*i.e.* the grasp axis is aligned

with the friction cones' orientations) is better than one whose grasping points are more nearly perpendicular (see Figure 5.2). Given the angles between the grasp axes and the friction cones ϕ_1, ϕ_2 and the friction angle α, the independence from friction Q_f is calculated thus:

$$Q_f = \frac{\phi_1 + \phi_2}{2\alpha} \tag{5.1}$$

This gives a score in the range 0 to 1, where 1 represents grasping points that are perfectly parallel, and 0 represents a grasp that is on the point of slipping from the robot's fingers. The *grasp zone* is the area where the finger may be placed while retaining the force closure property; the higher Q_f is, the larger the grasp zone. It is determined by finding the points where the grasp axis coincides with the outside of the friction cones. A better grasp gives a robot more leeway in the exact placing of its fingers, making the grasp more likely to succeed.

Of course, if a finger is misplaced within its grasp zone, the grasp zone for the other finger will change. The 'true' grasp zone is the area in which a finger can be placed anywhere while still guaranteeing force closure. This is half the size of the grasp zone.

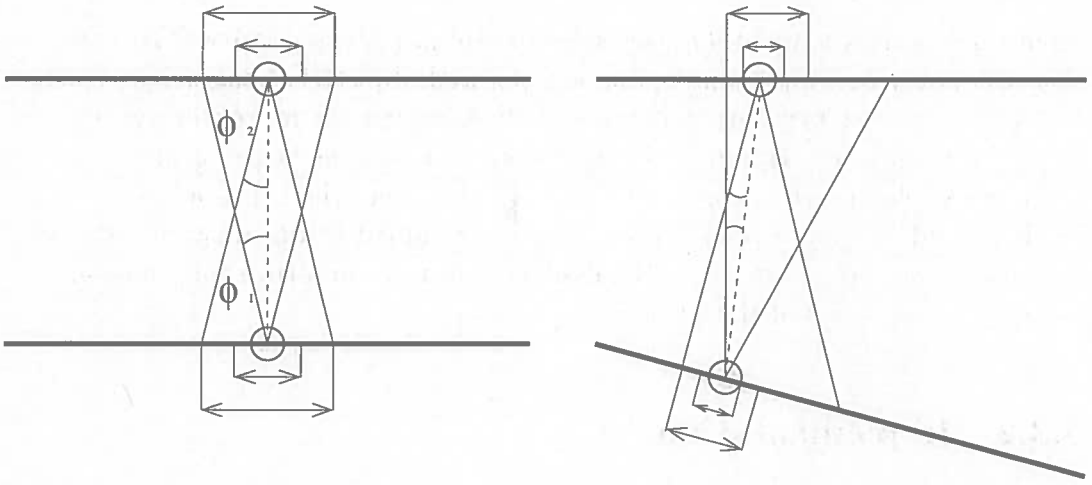

Figure 5.2: The left hand grasp is better than the right hand one, as the sum of angles between the grasp axes and the outsides of the friction cones is greater. The left grasp zones are larger, giving more leeway in placing the fingers. The 'true' grasp zones are shown inside the main grasp zones.

The second measure of quality that the system uses is the distance from the centre of mass to the grasp axis, illustrated in Figure 5.3. Minimising this distance results in a more stable grasp; the object is less likely to rotate around the grasp axis, as the weight is more evenly distributed in front of and behind the grasping

points. This can also be seen as minimising the amount of torque that the fingers must apply to prevent the object from rotating.

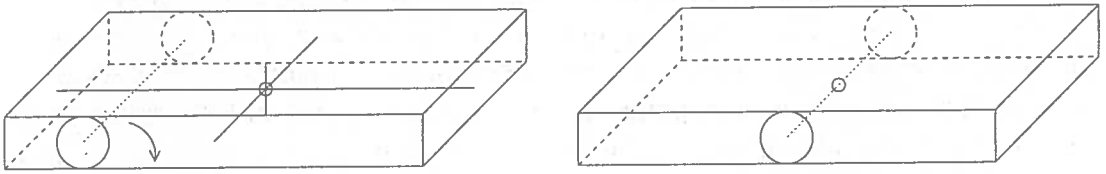

Figure 5.3: The left grasp, denoted by the two large circles, is likely to cause the object to rotate in the indicated direction, as the grasp axis is far away from the centre of mass, denoted by the small circle. The axis of the right grasp, however, passes straight through the centre of mass, so the object will not rotate when it is lifted.

Of course, it is not possible to know the centre of mass of an unknown object. However, assuming that the object's mass is evenly distributed, it is possible to make an estimate using the captured range data. The centroid of the data is simply the mean x, y and z value of all the data points; this is an approximation of the object's centre of mass. Not having a complete set of data points means that this value will probably be less accurate than in the ideal case, particularly with data from only one view. However, because the system does not need to calculate the centre of mass until it is evaluating grasps, and it needs at least two roughly opposing views to produce force closure grasps, it will have a reasonable representation of the object. This approach was found to work well in determining the approximate centre of mass of the object.

The smallest distance from a point to a line can be found with the following equation from Mathworld [16], where $\vec{l}_1$ and $\vec{l}_2$ are two points on the line (the centres of mass of the two grasping patches were used), and $\vec{p}$ is the point whose distance from the lines is desired (the centre of mass of the object):

$$d = \frac{|(\vec{l}_2 - \vec{l}_1) \times (\vec{l}_1 - \vec{p})|}{|\vec{l}_2 - \vec{l}_1|} \tag{5.2}$$

The cross product $\vec{u} \times \vec{v}$ is calculated as follows:

$$\vec{u} \times \vec{v} = \hat{x}(\vec{u}_y\vec{v}_z - \vec{u}_z\vec{v}_y) - \hat{y}(\vec{u}_x\vec{v}_z - \vec{u}_z\vec{v}_x) + \hat{z}(\vec{u}_x\vec{v}_y - \vec{u}_y\vec{v}_x) \tag{5.3}$$

where $\hat{x}$, $\hat{y}$ and $\hat{z}$ are unit vectors along each axis.

It is desirable for this distance to be converted into a measure between 0 and 1, so that it can be easily merged with the other criterion. Of course, the distance d should be minimised, so it should be subtracted from some maximum allowable

distance m. However, using a constant for m would treat differently sized objects inconsistently, *i.e.* using a value for m consistent with a large object would mean that all distances in a small object would become insignificant, while a small value of m would give a negative quality for all but the smallest distances in large objects. For this reason, m was determined dynamically by finding the largest distance from a data point (not necessarily a point in a grasping patch) to the object's centre of mass. The quality measure for distance to the object's centre of mass, taking values between 0 (bad) and 1 (good), is:

$$Q_d = \frac{(m - d)}{m} \tag{5.4}$$

Finally, the overall quality measure is simply the average of Q_f and Q_d.

5.4 Results

The tables in the Appendix show how different pairings of views produce different numbers and qualities of grasps. For each object, the number of grasps found when two views were registered is shown, followed by the number of these that satisfied the force closure property, and the maximum quality of grasp that these two views offered. A dash indicates that the two views could not be registered together, or that no grasps were found. The best grasp is shown graphically.

Figure 5.4 shows a grasp of quality 0.82575, found after five views had been taken of the stone. The other grasping patches and their surface normals are also shown.

5.5 Evaluation

A value of 0.75 was chosen as the threshold for a good enough grasp. This ensured that both criteria were at least reasonably good; scores of 0.5 and 1 represent the worst case for one criterion. As the results show, the cow and the wooden block have many combinations of two views that result in a good enough grasp. The stone has considerably fewer force closure grasps, and all of them score below the threshold. The result of this is that the stone will always require at least three views before a good enough grasp can be found, while good grasps for the wooden block and the cow will often be found from just two views.

The centre of mass detection could be slightly inaccurate in its estimation; this caused grasps to tend away from the true centre of mass of the object. For example, given four faces of the wooden block (three sides and the top), the top

face and the side face whose opposing face had not been seen caused the estimated centre of mass to be shifted in their direction (see Figure 5.8). This sometimes caused grasps to gravitate away from the optimal grasping position; however, it was not a major problem.

In practice, a typical 'best grasp' scored from roughly 0.8 to 0.9 and had similar friction and distance qualities. Generally, several grasps of this quality were found after a few iterations; the highest scoring one of these was returned as the final grasp.

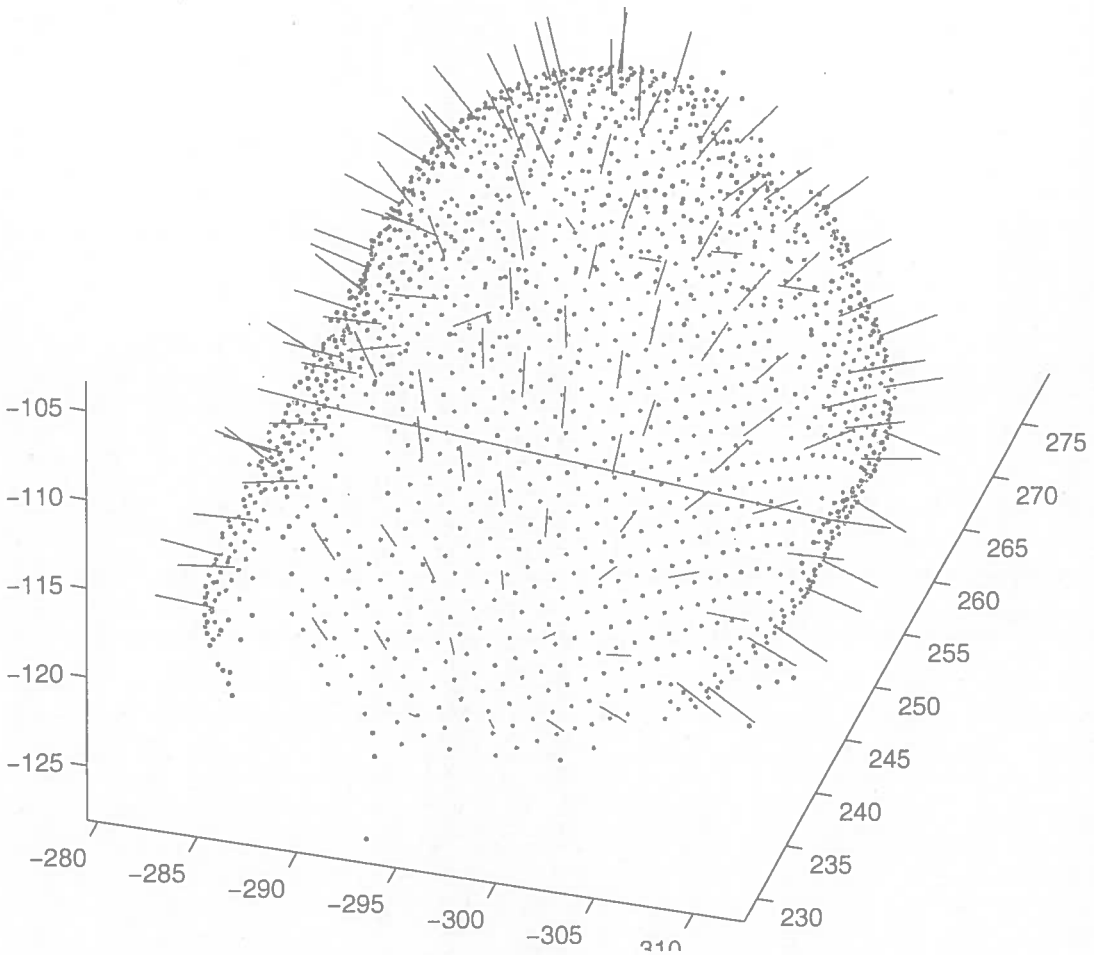

Figure 5.4: The green points are those that form part of a grasping patch; the rejected points are in blue. The short red lines are the normals of the grasping patches. The long red line links together the two patches that formed the successful grasp. The red dot is the stone's estimated centre of mass.

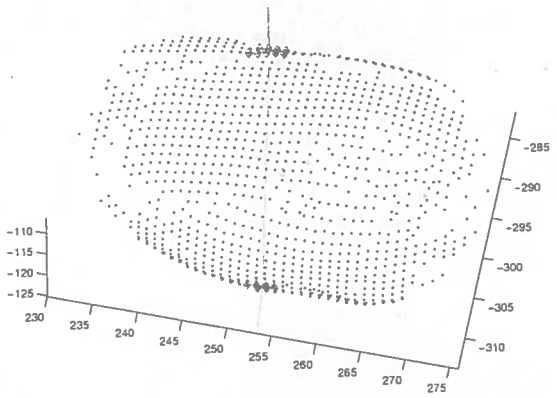

Figure 5.5: This is the best grasp that can be found from two views from the stone data set. The two view positions are 0° elevation and 180° azimuth, 90° elevation. This grasp scored 0.5889.

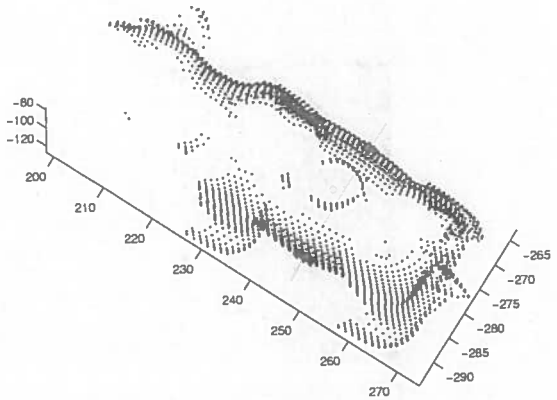

Figure 5.6: This is the best grasp that can be found from two views from the cow data set. The two view positions are 0° azimuth, 90° elevation and 225° azimuth, 90° elevation. This grasp scored 0.92221.

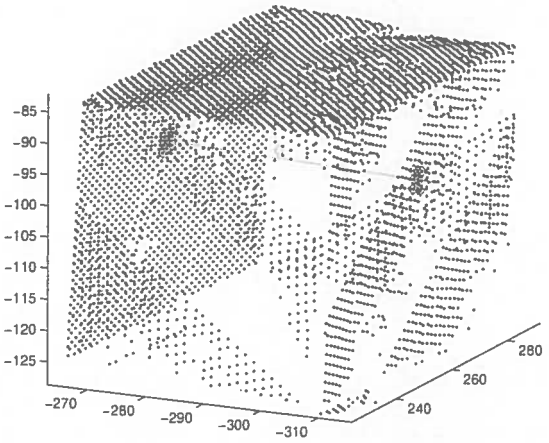

Figure 5.7: This is the best grasp that can be found from two views from the wooden block data set. The two view positions are 225° azimuth, 45° elevation and 45° azimuth, 45° elevation. This grasp scored 0.96703.

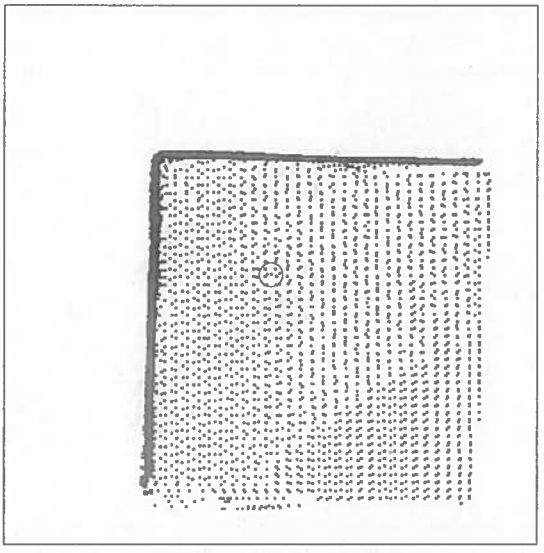

Figure 5.8: The green centre of mass (circled) was shifted by roughly one sixth of the side length towards the side and top faces.

6. Next Best View Determination

The goal of this part of the system is to decide, given the grasping points found in the current data, from which position the system should next capture data. A simple voting structure was used, which, based on angles between grasping patches and cell directions, decided which cell would be the best to examine next.

The azimuth of the initial view was chosen randomly. However, the elevation was always set to 45° for the first view. This helped with registration of subsequent views; an elevation of 0° or 90° would have made it difficult to find views that could be registered with data from the initial view.

6.1 Voting Structure

The hemisphere surrounding the object was divided into 17 cells, spaced at 45° intervals. Figure 6.1 shows the front 11 of the 17 views used. This choice of relatively coarse discretisation was made to reduce the number of scans required per object for the simulation; for example, just reducing the step size to 30° would have increased the number of scans required per object to 37. However, a real robot would not be using pre-scanned views, but would be scanning to order, allowing views from any azimuth and elevation. In this case, the step size might be 5°, resulting in 1297 cells ($360 \div 5 = 72$ cells per level, $90 \div 5 = 18$ levels, 1 cell at the top of the hemisphere; $72 \times 18 + 1 = 1297$ cells). This model was proposed by Horn [13] and Connolly [8].

As discussed in Garcia *et al.*'s paper[11], this is not an ideal method for discretising a hemisphere. The cells are not of uniform size – those at the top have less area than those nearer the horizon. This has the result that votes for an area around the top of the hemisphere are likely to be split among several cells, while a single cell representing the same area further down will collect all of the votes for that area; the net effect is that cells nearer the horizon will have a better chance of being voted for. A spherical discretisation map, as proposed by Garcia *et al.* [11], or a tesselation based on a regular polyhedron, suggested by Horn [13], would be superior. However, it was considered that given the relatively coarse level of discretisation, neither of these more sophisticated approaches would provide greatly improved results. Additionally, cells around the horizon are more likely to be those of interest, as graspable points will often have normals corresponding

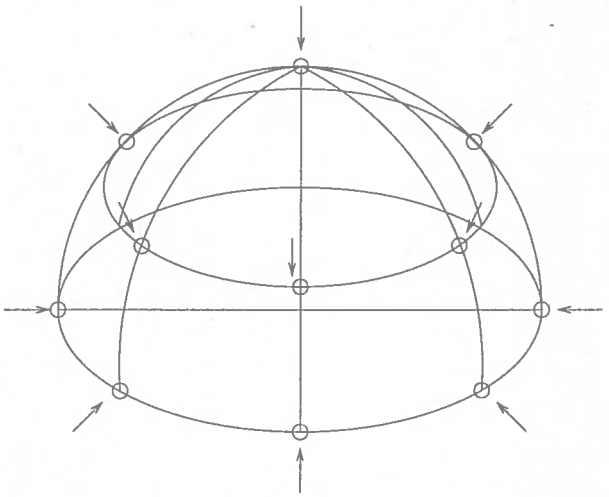

Figure 6.1: Views were taken from the indicated points on the hemisphere towards its centre (rear arrows and circles omitted for clarity).

to areas found in those cells, so this bias could be useful.

6.2 Face Quality Determination

Initially, the decision about which viewing cell each grasping point should vote for was simple, as all that is required for a two-fingered grasp is a pair of parallel, opposed grasping patches. The dot product of the normals of each unseen viewing cell and each grasping patch was calculated (see Equation 1.1), and the one vote allocated to each grasping patch was cast for the viewing point that resulted in the lowest dot product; this corresponded to the patch that was most nearly opposite it. The next view chosen was the one whose corresponding cell had the most votes.

This simple algorithm was found to work well. However, some refinements were made to improve its performance with some objects.

When the largest dot product of a grasping patch's normal and every unseen viewing cell is 0 or greater, it is highly improbable that any more information will be found that will help form a grasp. The only situation in which this could occur is that of a patch with a normal pointing straight up (see Figure 6.2), but even then the number of points found would probably be too small for a patch to be found.

Because finding a patch in this way was extremely unlikely, and in the vast majority of cases a patch that had a dot product of 0 or more would nominate a

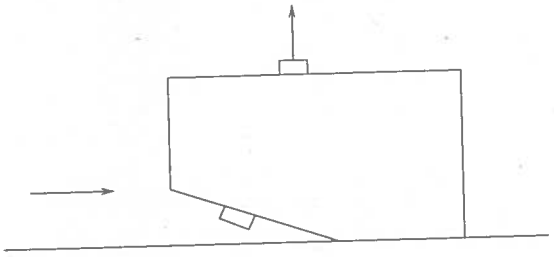

Figure 6.2: The dot product of the top patch's normal and the viewing direction is zero; however, the patch on the underside of the object could be used as part of a grasp.

patch that would not be any use, it was decided to discount any votes that did not have a good enough score, *i.e.* 0 was the maximum threshold.

A second improvement involved increasing the probability that a new view contained significant new information, *i.e.* it was a reasonable distance from previous views. If a grasp was not found after two iterations, or if corresponding points could not be found for ICP, a view very close to the previous one was often chosen. This is undesirable, as little new data will be captured from this view. Figure 6.3 shows one situation where this causes a problem. The initial view 1 (azimuth 45°, elevation 0°) sees faces A and B. As B has a larger area than A, more grasping points are found on it, and its favoured view 2 (azimuth 270°, elevation 0°) is chosen. Unfortunately, the face seen by view 2 is irregular and offers no new grasping points. The second round of voting acts on the same data as the first round, but cannot this time vote for view 2. Instead, it will typically choose view 3 (azimuth 270°, elevation 45°). This offers information on the top face, but this is of little use. A preferable view would be 4 (azimuth 180°, elevation 0°). This captures grasping points that form grasps with those on face A, completing the process.

Given this initial view, the smallest number of views possible is three (*e.g.* 1, 2 and 4). The system must be dissuaded from voting for views that are close to other views that have already been seen. The process used was to find the number of previously seen views within 45° in any direction of the view under consideration n, and to divide its number of votes by $n+1$. This reduces the influence over the choice of the next view of cells that are in a similar region to cells that have already been viewed. In the example above, view 4 would be chosen over view 3, as the importance of view 3 would be diminished by its proximity to view 2. The dot product d was used to determine which views were within 45° of the view being considered; if $\text{acos}(d) < 45$, the view is near.

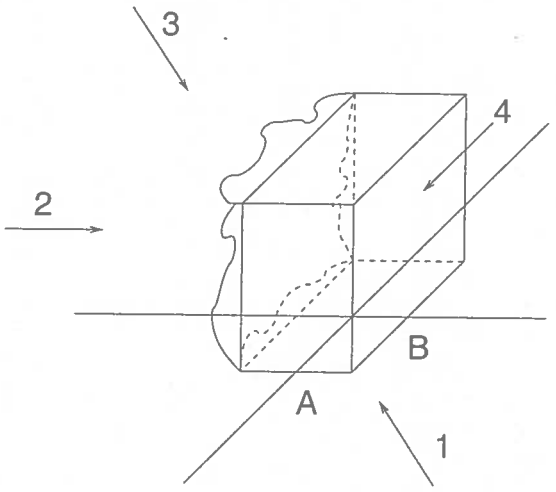

Figure 6.3: View 3 would normally be chosen after view 2 is found not to be useful, though view 4 would be preferable.

6.3 Example 1

The following example used the stone data set; the initial view was from the 0° azimuth, 45° elevation position. A grasp was found after three views had been taken.

40 patches constituting 440 points were found in the first view's data; these are shown alongside the 101 points that were rejected in Figure 6.4. The votes that were cast given these patches are in Table 6.1.

The view chosen was at the 180° azimuth, 45° horizon position. Its 615 points were registered with the first view to give a new set of 1118 points. 88 patches were found in this; 156 points were rejected. These are shown in Figure 6.5. 44 grasps (see Figure 6.6) were found at this stage; the best of these had a quality of 0.64512. As this grasp quality was not good enough, a second round of voting was carried out. The results are in Table 6.2.

The 180° azimuth, 90° horizon position provided 100 patches (see Figure 6.7) following registration, and a good enough grasp (of quality 0.79824) was found. This is shown in Figure 6.8.

6.4 Example 2

This example used the cow data set and an initial view from the 225° azimuth, 45° horizon position. The initial set of 126 patches, drawn from 1858 points, is

shown in Figure 6.9. The votes are shown in Table 6.3; the view from the 0° azimuth, 45° horizon position was chosen.

Following registration, the data set grew to 3354 points, from which 209 patches were found. 580 force closure grasps were found; the best one had a quality of 0.87793 (see Figures 6.10 and 6.11).

6.5 Evaluation

The system successfully chose good next views for some objects, allowing grasps to be found after typically two views. Generally, these were about 180° apart, though this depended on what could be seen from the initial view. In cases where the second view offered no useful information, further views from this area were successfully avoided to allow views from more useful angles. However, the stone typically required about five views before a sufficiently good grasp could be found, although possible grasps were often found after just two or three views.

Tables of all results are in Appendix C.

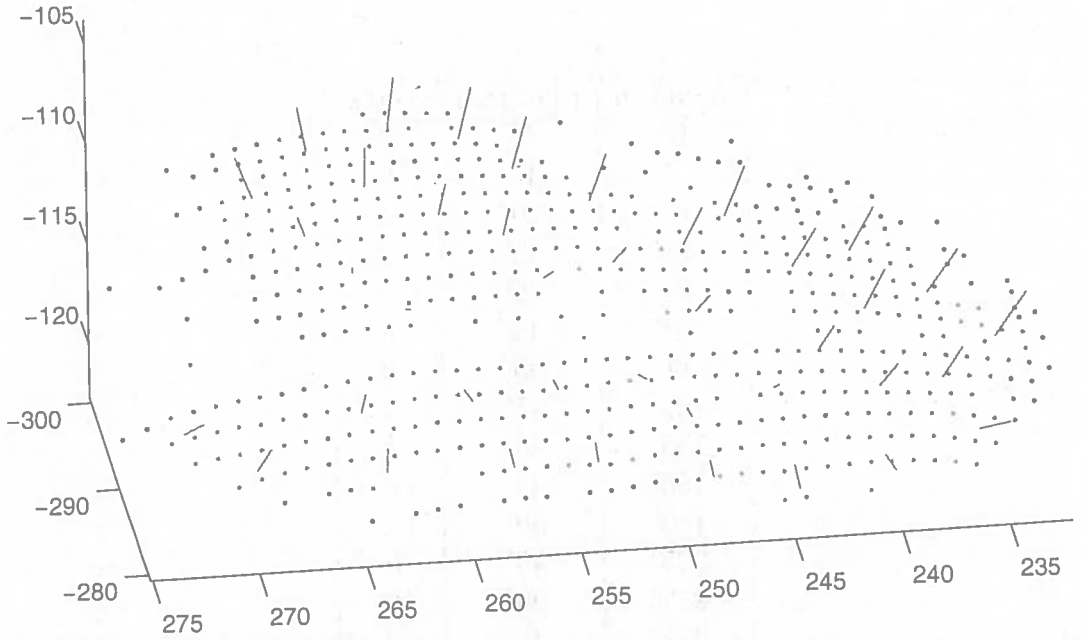

Figure 6.4: The initial stone data set. The points of the found planes are green, the rejected points are blue. The red lines are the normals of the planes.

Azimuth	Elevation	Votes
0°	0°	0
0°	45°	0
0°	90°	0
45°	45°	0
45°	90°	0
90°	45°	0
90°	90°	0
135°	45°	0
135°	90°	3
180°	45°	0
180°	90°	26
225°	45°	0
225°	90°	11
270°	45°	0
270°	90°	0
315°	45°	0
315°	90°	0

Table 6.1: First round of voting for the stone.

Azimuth	Elevation	Votes
0°	0°	0
0°	45°	0
0°	90°	12
45°	45°	0
45°	90°	7
90°	45°	0
90°	90°	0
135°	45°	0
135°	90°	2
180°	45°	0
180°	90°	14.5
225°	45°	0
225°	90°	9
270°	45°	0
270°	90°	6
315°	45°	0
315°	90°	11

Table 6.2: Second round of voting for the stone.

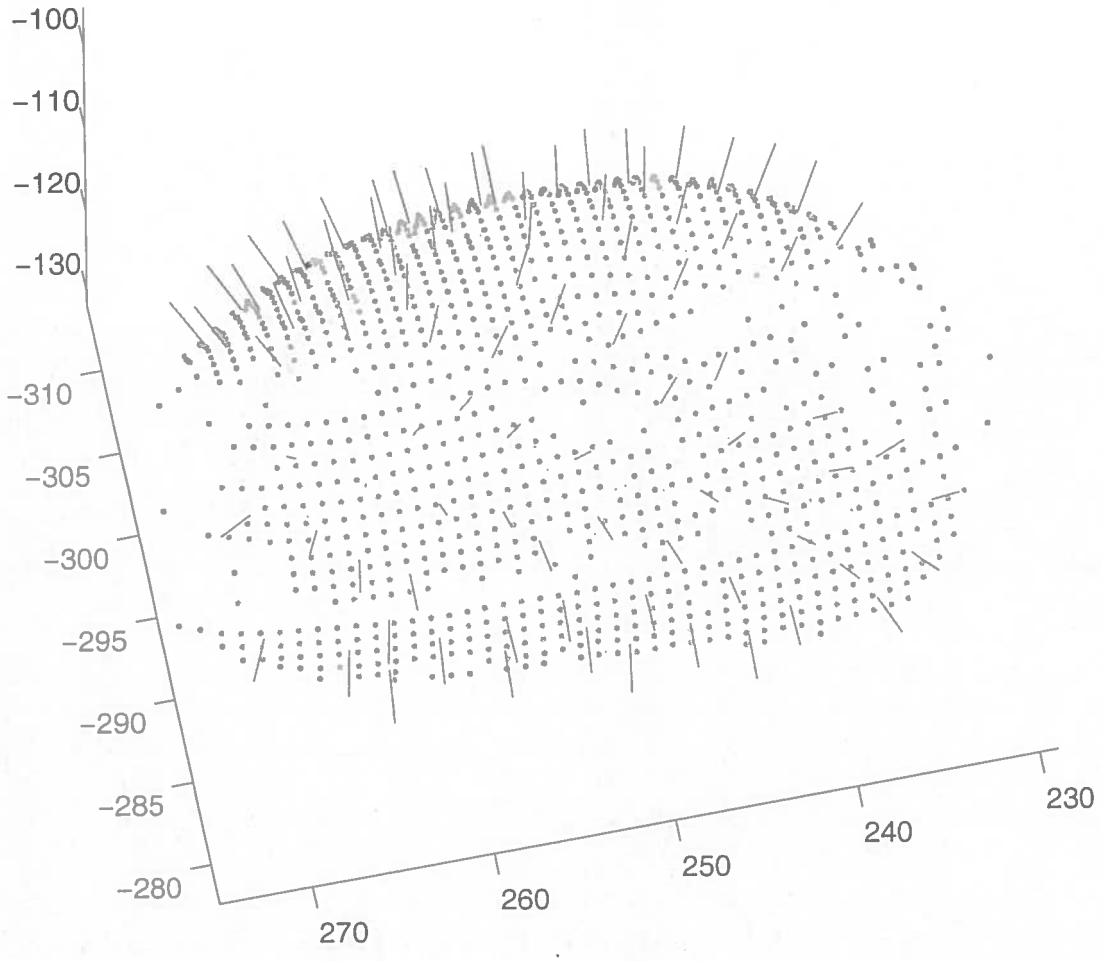

Figure 6.5: The stone data set following the registration of the second view. As before, the points of the found planes are green, the rejected points are blue, and the red lines are the normals of the planes.

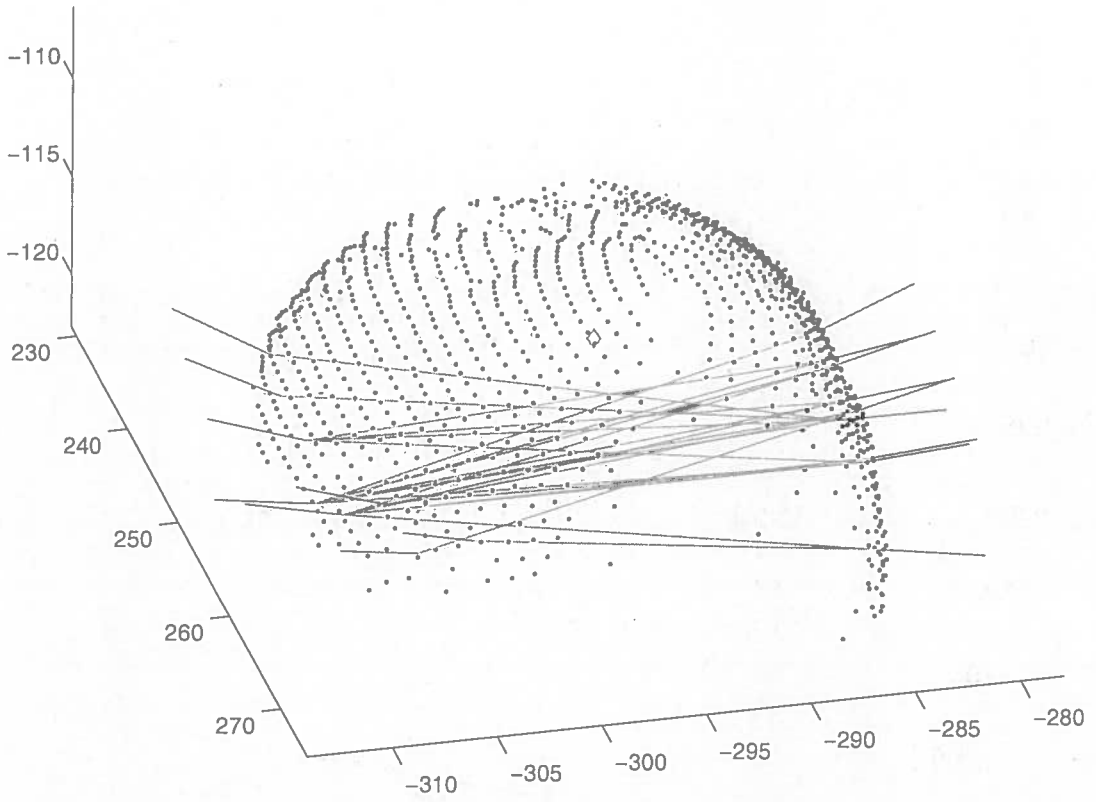

Figure 6.6: The red lines are the normals of the patches used in feasible grasps, the green lines connect the constituent patches of the grasps, and the red diamond is the estimated centre of mass of the object.

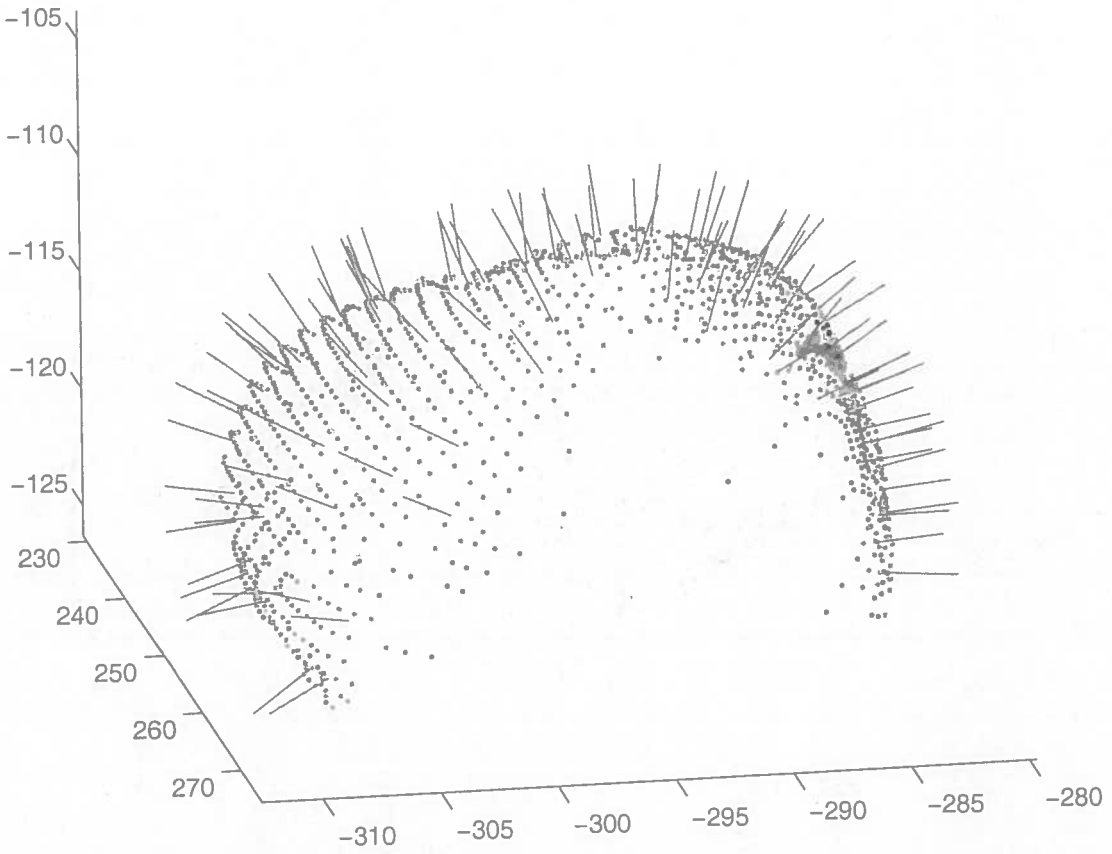

Figure 6.7: The stone data set following the registration of the third view. As before, the points of the found planes are green, the rejected points are blue, and the red lines are the normals of the planes.

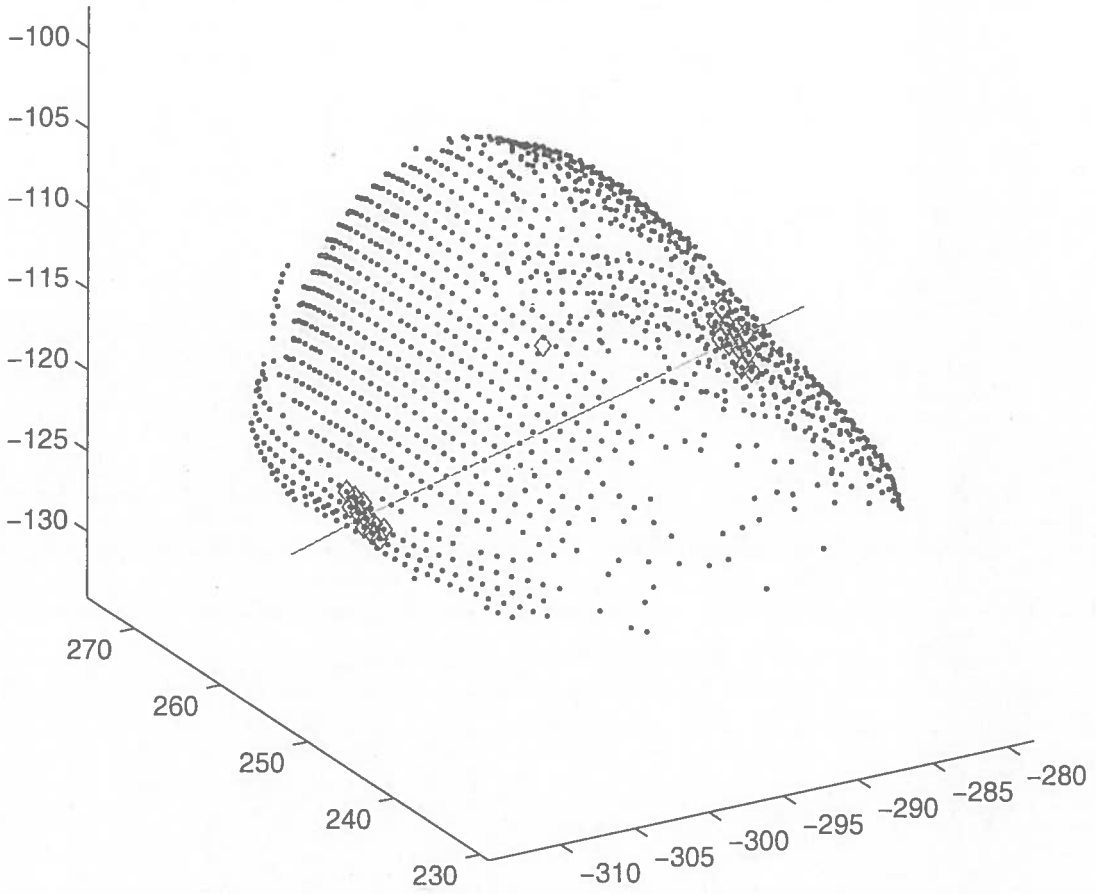

Figure 6.8: The red points are the planes to be used in the grasp, the green line contains the grasping patch normals and the grasp axis, and the red diamond is the estimated centre of mass.

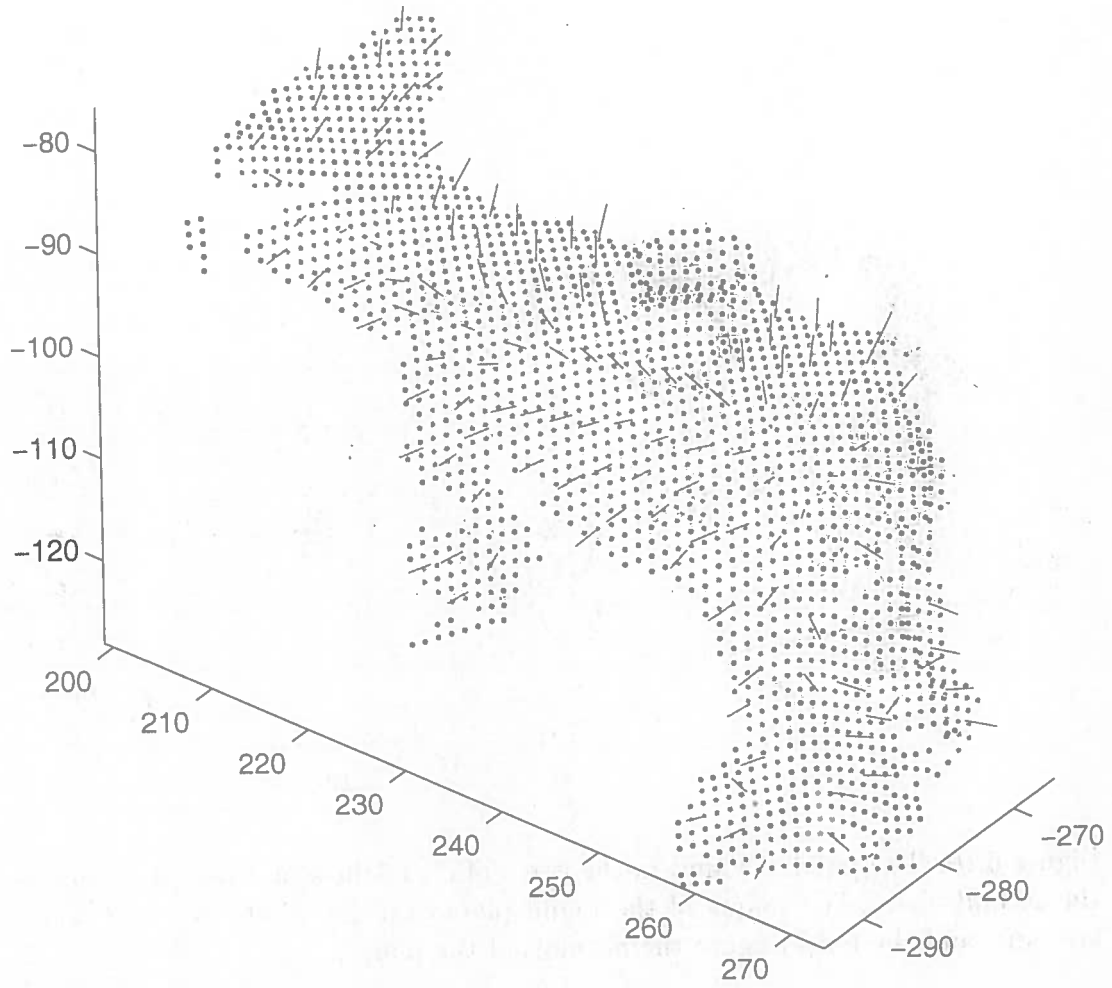

Figure 6.9: The initial cow data set. The points of the found planes are green, the rejected points are blue, and the red lines are the normals of the planes.

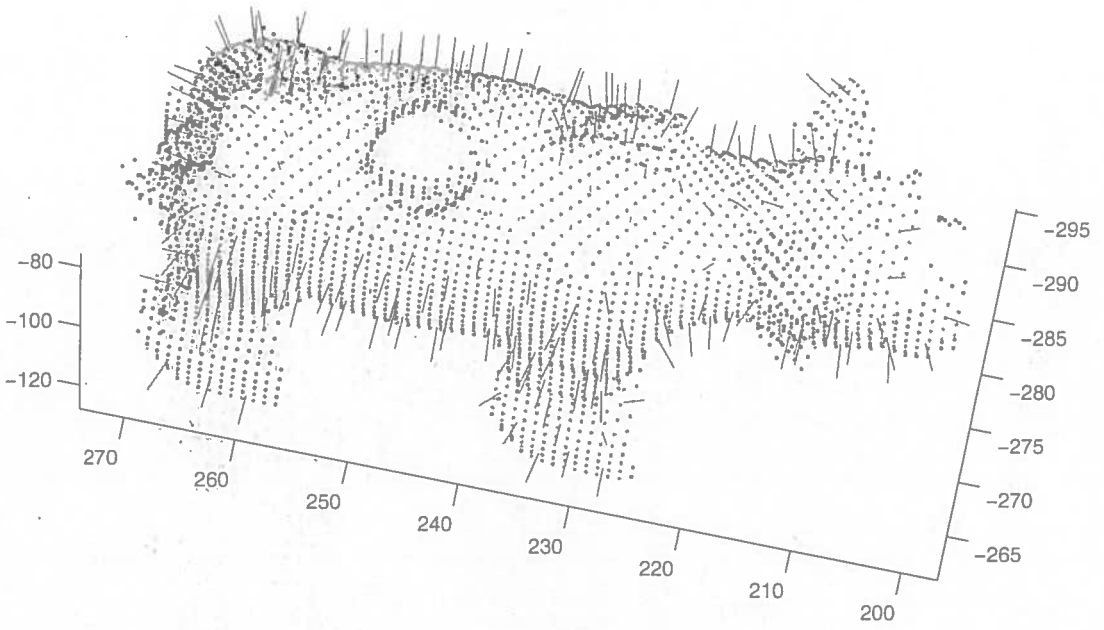

Figure 6.10: The patches found in the cow data set following the registration of the second view. The points of the found planes are green, the rejected points are blue, and the red lines are the normals of the planes.

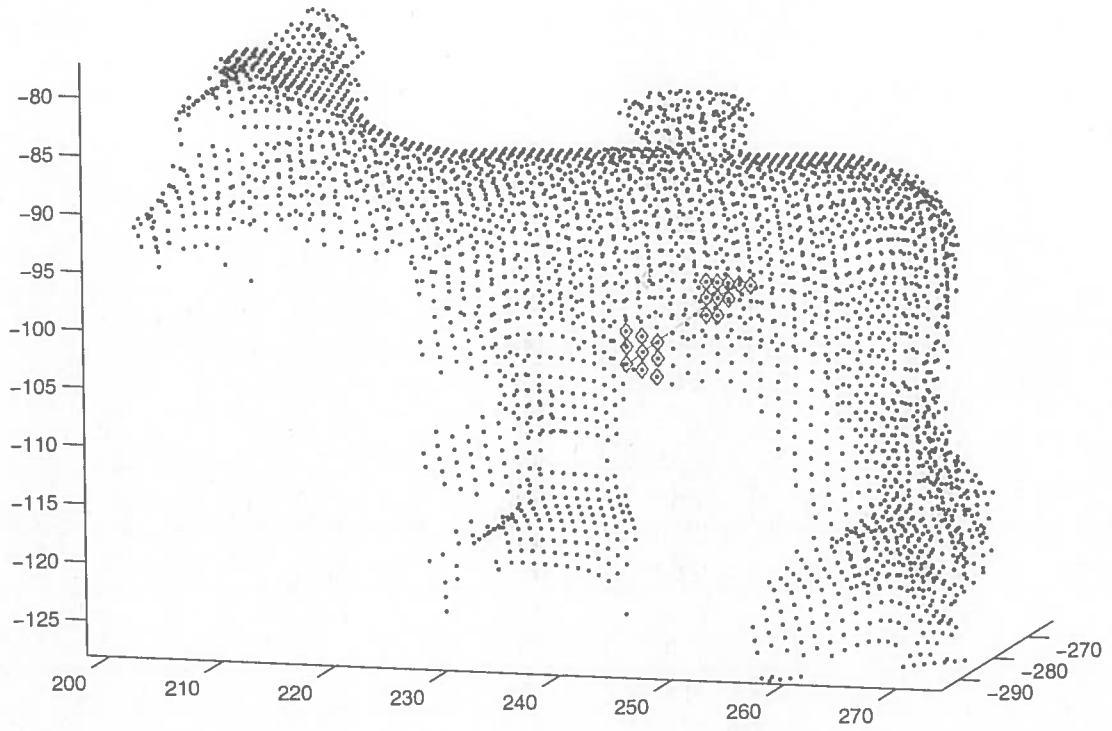

Figure 6.11: The red points are the planes to be used in the grasp, the green line contains the grasping patch normals and the grasp axis, and the green diamond is the estimated centre of mass.

Azimuth	Elevation	Votes
0°	0°	0
0°	45°	0
0°	90°	66
45°	45°	0
45°	90°	12
90°	45°	0
90°	90°	28
135°	45°	0
135°	90°	6
180°	45°	0
180°	90°	9
225°	45°	0
225°	90°	1
270°	45°	0
270°	90°	0
315°	45°	0
315°	90°	4

Table 6.3: First round of voting for the cow.

7. Conclusions

7.1 Experimental Conclusions

The aim of finding a good two-fingered grasp with just two or three views was achieved in 75% of cases. This was dependent on the object being grasped: the cow was grasped with data from only two views in all cases; a grasp was found for the wooden block with two or three views in all cases except for one anomalous instance; while the stone required three or five views in most cases. However, in many cases where a good grasp was not found within two or three views, a usable grasp was found. These grasps exhibited the force closure property, but their quality was not above the threshold of 0.75. The emphasis of the system could be changed by altering this threshold; reducing it to 0.6, for example, would emphasise speed rather than grasp quality, perhaps suiting a robot capable of very precise finger placement, obviating the need for large grasp zones. Using this value would increase the success rate to 83% over all objects. Conversely, a value of 0.9 would suit an application that demanded very stable grasps, and would require more views in most cases.

Range data registration and fusion was carried out successfully and with little error. Registration of data sets that were too dissimilar (*i.e.* they had too few common points) was rejected, preventing points from being registered where they should not have been.

The grasping patch finding strategy was the slowest part of the system, but was guaranteed not to miss out any feasible grasping patches, and had safeguards in place to prevent unsuitable patches from being accepted. However, the removal of a patch's constituent points from the pool of potential patches meant that patch centres could be no closer than 5mm to each other. When picking up large objects this is of little import; however, a small object may require more flexibility in finger placement to find a good grasp, as the shorter the grasp axis is, the smaller the grasp zones are.

Although it is hard to objectively define a quality measure for a grasp, it was felt that the system delivered good, perhaps even conservative, decisions on a grasp's quality. The slight inaccuracies in determining the objects' centres of mass, as well as the assumption that the objects' masses were evenly distributed, could make the grasps more reliant on torque to prevent the object from rotating. However, the distances from the grasping patches to the ideal points (those in line with the true centre of mass) was small. The assumption of a fairly high coefficient of friction was not altogether unwarranted; robot fingers are frequently rubber

tipped to provide much-needed friction.

The best next view decision worked well in general, in particular for the wooden block and the cow, allowing good grasps to be found with just two or three views. Its performance on the stone was not quite as good; however, as can be seen in Table 6.3, there are very few good immediate next views for the stone, so this is not a simple problem. In this case, it is more of a matter of luck in randomly choosing an initial view that has suitable partners for good grasps. One reason for this poor performance is that the rounded, graspable sides of the stone require views from near the horizon to find enough points to form grasping patches whose normals are almost parallel to the ground plane. These views are hard to register with other, similar but opposed views, as they share few or no points. Experimentation with different step sizes in discretising the hemisphere surrounding the object could help: a view with an elevation of about 20° might be able to see enough points to form a grasping patch, while also seeing enough of the top of the stone to allow registration with a similar but diametrically opposed vies (see Figure 7.1).

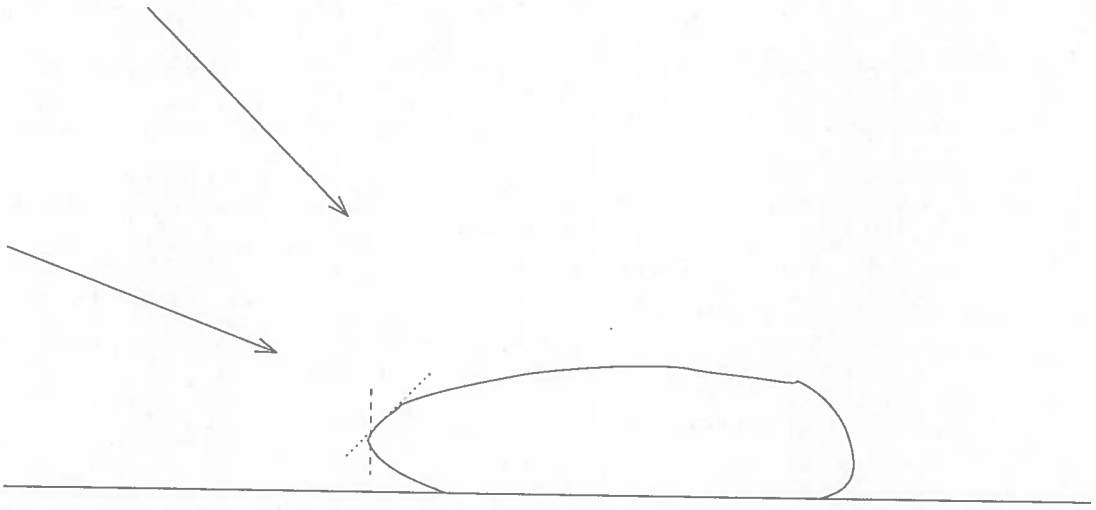

Figure 7.1: The view from an elevation of 45° can see only the grasping patch represented by the dotted line, which is useless for grasping. The view from an elevation of 20°, on the other hand, can see the edge of the stone that approximates the vertical grasping patch represented by the dashed line, which can be used as part of a grasp.

7.2 Further Work

7.2.1 Further Research

A major improvement to the system would be to allow three- or more-fingered grasps. This would allow much more flexibility in both picking up and manipulating objects, as well as more stable grasps. Several heuristics could be employed in choosing the best next view. For example, Chinellato *et al.* [7] state that a three-fingered grasp ideally has the three fingertips at the points of an equilateral triangle to improve the balance of the grip. With this in mind, two grasping patches could be picked at random, and the view observing their ideal companion voted for, if it had not already been observed. The weighting of the vote could be commensurate with the area of the triangle the grasp would form, a large grasping triangle also rendering the grip more stable.

7.2.2 Implementational Improvements

Improvements could be made in several areas of the system:

- If it was necessary to register many views to obtain a good grasp, a more sophisticated version of ICP could be used for simultaneous re-registration of the captured data for more accuracy in this process.
- Grasping patch finding could allow patches to be closer together, and could also evaluate the curvature of the patches to admit advantageously curved areas. Concave patches are better than convex ones for grasping; this could be used in grasp evaluation. This process could also be made more efficient by attempting to retain the structure of the range images in areas that shared no points with other views to speed up plane finding.
- Further measures could improve the assessment of a grasp's quality; for example, it is preferable for the grasp axis to be above rather than below the object's centre of mass, or the object may rotate when it is grasped.
- A less coarsely discretised hemisphere would allow finer control over the next view. As pointed out above, this could have helped with finding grasping patches on the edge of the stone. Also, a structure such as an icosahedron would distribute votes more evenly, rather than concentrating them around the horizon.

Appendix A. Range Data Registration and Fusion Results

Index	Azimuth/°	Elevation/°	Cow	Wooden block	Stone
1	0	45	2045	4054	541
2	0	90	1952	2950	431
3	45	45	1804	3674	599
4	45	90	1878	3276	180
5	90	45	1052	3039	616
6	90	90	1000	1900	83
7	135	45	1804	3360	676
8	135	90	1878	3253	174
9	180	45	2045	3436	615
10	180	90	1952	2861	433
11	225	45	1858	3306	533
12	225	90	1571	3220	324
13	270	45	1588	3307	313
14	270	90	915	1886	151
15	315	45	1858	3961	311
16	315	90	1571	3336	209
17	0	0	1309	2892	784

Table A.1: Number of points for each object and view angle for each view.

	1	2	3	4	5	6	7	8	9
1	–	0.36301	0.48674	0.50959	0.55389	0.62238	0.56462	–	0.482
2	0.34462	–	0.51452	0.49889	0.6084	0.50926	0.62268	–	0.561
3	0.52869	0.57204	–	0.38768	0.52039	0.53105	0.53412	0.53946	0.604
4	0.53424	0.5325	0.3523	–	0.55721	0.52055	0.56295	0.41126	0.539
5	0.55789	0.63942	0.52445	0.54644	–	0.54739	0.50518	0.56888	0.541
6	0.62437	0.52617	0.53002	0.51823	0.50272	–	0.56972	0.44945	0.578
7	0.57062	0.62809	0.53016	0.561	0.50736	0.56022	–	0.41272	0.520
8	0.60456	–	0.57018	0.42554	0.56274	0.4801	0.41137	–	0.540
9	0.48225	0.66415	0.59439	0.56544	0.54745	0.59312	0.48065	0.52358	–
10	0.5737	–	0.62382	–	0.53797	0.51433	0.53138	0.38714	0.304
11	0.6048	0.67059	0.52655	0.65439	0.57339	0.63459	0.55329	0.58517	0.513
12	0.6218	0.59841	–	–	0.60904	–	0.56188	0.46709	0.506
13	0.54285	0.63279	0.57439	0.6168	0.43345	0.61931	0.56856	0.61324	0.564
14	0.62606	0.57494	0.64888	0.57895	0.53667	–	0.65043	0.64239	0.671
15	0.50684	0.53381	0.56183	0.57213	0.57597	0.63358	0.53221	0.59792	0.614
16	0.52229	0.39856	0.58336	0.53363	0.65038	0.59548	–	–	0.664
17	0.3783	0.51519	0.50385	0.57604	0.41654	0.56349	0.49844	0.62983	0.445
	10	11	12	13	14	15	16	17	
1	0.58286	0.60772	0.64622	0.54756	0.60857	0.48653	0.50218	0.42813	
2	–	0.65121	0.62683	0.60996	0.5804	0.47058	0.33837	0.48463	
3	0.62109	0.52215	–	0.61558	0.65581	0.56802	0.60681	0.55278	
4	–	0.63174	–	0.62149	0.57675	0.57142	0.51454	0.56909	
5	0.56505	0.57935	0.5758	0.4517	0.57958	0.60002		0.4528	
6	0.52005	0.63709	–	0.61732		0.65258	0.63722	0.54861	
7	0.55424	0.56063	0.58494	0.56612	0.6339	0.55831	–	0.5386	
8	0.43784	0.60514	0.48551	0.6224	0.62141	0.65532	–	0.6309	
9	0.34684	0.48294	0.49762	0.55311	0.67005	0.60914	0.66727	0.46884	
10	–	0.46465	0.34256	0.5975	0.62957	0.64913	0.61654	0.49173	
11	0.52334	–	0.36516	0.52001	0.60386	0.59479	0.64365	0.5412	
12	0.38612	0.33085	–	0.54443	0.5358	0.63145	0.51866	0.59429	
13	0.63655	0.51697	0.58229	–	0.46218	0.50471	0.55703	0.42353	
14	0.63537	0.51338	0.52946	0.41801	–	0.51283	0.41185	0.50313	
15	0.65496	0.59323	0.65668	0.50355	0.5734	–	0.37956	0.51991	
16	0.60968	0.63981	0.52025	0.5499	0.42353	0.36016	–	0.61595	
17	0.4891	0.53406	0.59619	0.41562	0.52436	0.51077	0.61709	–	

Table A.2: Mean distance error for registering pairs of views from the cow data set.

units? precision?

	1	2	3	4	5	6	7	8	9
1	–	2300	2305	2643	2422	2754	3332	–	3558
2	2514	–	2701	2478	2718	2722	3628	–	3873
3	2107	2305	–	2312	2013	2302	2938	3397	3308
4	2529	2178	2393	–	2427	2123	3428	3367	3807
5	2345	2656	1939	2430	–	1610	2009	2508	2407
6	2772	2704	2327	2163	1652	–	2453	2176	2782
7	3327	3629	2929	3445	2077	2392	–	2299	2097
8	3775	–	3353	3342	2481	2119	2379	–	2526
9	3558	3840	3328	3799	2510	2762	2306	2653	–
10	3866	–	3619	–	2769	2725	2715	2458	2533
11	3354	3661	3252	3597	2389	2723	2472	2876	2253
12	3495	3433	–	–	2534	–	2633	2588	2477
13	2746	3086	2732	3224	2139	2501	2774	3210	2763
14	2707	2651	2594	2726	1932	–	2590	2707	2777
15	2244	2439	2516	2874	2392	2738	3237	3606	3355
16	2474	2115	2650	2616	2486	2504	–	–	3504
17	2421	2874	2260	2861	1667	2138	2345	2893	2512
	10	11	12	13	14	15	16	17	
1	3860	3341	3446	2747	2701	2359	2545	2350	
2	–	3674	3426	3146	2640	2763	2374	2903	
3	3612	3245	–	2719	2580	2410	2571	2164	
4	–	3613	–	3220	2722	2916	2650	2848	
5	2737	2338	2530	2111	1928	2339	–	1525	
6	2722	2729	–	2506	–	2720	2498	2153	
7	2289	2393	2553	2743	2601	3224	–	2246	
8	2133	2960	2596	3210	2710	3600	–	2883	
9	2318	2360	2550	2793	2768	3330	3504	2415	
10	–	2769	2370	3217	2711	3620	3423	2990	
11	2455	–	2215	2252	2243	2950	3105	2233	
12	2108	2354	–	2552	1850	3144	2793	2658	
13	3101	2260	2545	–	1722	2228	2476	1863	
14	2654	2291	2004	1862	–	2271	1848	2055	
15	3626	2953	3095	2197	2125	–	2209	2175	
16	3429	3145	2791	2519	1766	2332	–	2587	
17	2986	2301	2657	1963	2042	2227	2587	–	

Table A.3: Number of points returned from fusion of two views from the cow dataset.

	1	2	3	4	5	6	7	8	9
1	–	0.34233	0.48565	0.46555	0.49303	0.64289	0.4964	0.63214	0.37407
2	0.25788	–	0.39634	0.27222	–	–	–	–	–
3	0.52376	0.53438	–	0.49994	0.52126	0.58924	0.52203	0.60147	0.50742
4	0.483	0.37277	0.43599	–	0.50224	0.38047	0.52571	0.41483	–
5	0.49331	–	0.48454	0.49886	–	–	0.51443	0.49364	0.50211
6	0.63028	–	0.42728	0.27373	–	–	0.40173	0.27416	0.63947
7	0.4782	–	0.4944	0.56898	0.49928	0.54439	–	0.58204	0.49756
8	–	–	0.51585	0.38817	0.46602	0.35512	0.50508	–	0.49352
9	0.3427	–	0.47339	0.59895	0.47651	0.62281	0.47352	0.50703	–
10	–	–	–	–	–	–	0.42328	0.26452	0.27043
11	0.47635	–	0.37992	–	0.47685	0.64547	0.48325	0.59473	0.50251
12	–	–	–	–	–	–	0.52247	0.36598	0.48586
13	0.47813	–	0.47527	–	0.35286	–	0.47578	–	0.47862
14	–	–	–	–	–	–	–	–	–
15	0.51701	0.54475	0.50915	0.53901	0.48453	–	0.35252	–	0.48044
16	0.48701	0.3848	0.48786	0.36372	–	–	–	–	–
17	0.26762	–	0.39726	–	0.26955	–	0.39183	–	0.26888
	10	11	12	13	14	15	16	17	
1	0.51256	0.49701	–	0.49761	0.618	0.4905	0.46958	0.38186	
2	–	–	–	–	–	0.4097	0.28784	–	
3	–	0.41384	–	0.5027	–	0.52984	0.54342	0.50516	
4	–	–	–	–	–	0.48284	0.36811	0.63689	
5	0.62719	0.49551	–	0.37998	–	0.49436	–	0.38649	
6	0.34717	0.60684	–	–	–	–	–	–	
7	0.59054	0.48407	0.60691	0.47266	–	0.34938	–	0.4705	
8	0.36997	0.49262	0.3701	–	–	–	–	–	
9	0.39973	0.47523	0.50655	0.47034	–	0.47107	–	0.36163	
10	–	0.40917	0.26986	–	–	–	–	–	
11	0.58567		0.58999	0.50071	0.57407	0.48567	0.48612	0.46664	
12	0.36734	0.49526		0.4812	0.36765	0.37441	0.29447	–	
13	–	0.47889	0.48986	–	0.38291	0.48074	0.49307	0.36247	
14	–	0.40563	0.273	0.27723	–	0.3996	0.28984	–	
15	–	0.4956	0.47104	0.51318	0.56652	–	0.57513	0.47737	
16	–	0.41447	0.29946	0.49011	0.3759	0.51614	–	–	
17	–	0.39298		0.26342		0.39295	–	–	

Table A.4: Mean distance error for registering pairs of views from the wooden block dataset.

	1	2	3	4	5	6	7	8	9
1	–	4170	4479	5304	5219	5817	5361	7172	5428
2	4864	–	5154	4287	–	–	–	–	–
3	3798	3876	–	4372	4252	4583	4615	6137	5119
4	5177	3476	5020	–	5063	3290	5705	5171	–
5	5258	–	4530	5096	–	–	3668	4907	4434
6	5843	–	5093	3884	–	–	4401	3768	5233
7	5486	–	4863	5435	3528	3610	–	4830	4104
8	–	–	6438	5286	5045	3307	5372	–	5249
9	5602	–	5331	6555	4587	5208	4330	5161	–
10	–	–	–	–	–	–	5629	4224	4842
11	5455	–	5202	–	4471	5090	4320	5244	3762
12	–	–	–	–	–	–	5996	4678	5251
13	5430	–	5196	–	4509	–	4634	–	4674
14	–	–	–	–	–	–	–	–	–
15	4233	4387	4548	5477	5193	–	5315	–	5375
16	5294	3447	5590	4721	–	–	–	–	–
17	5015	–	4780	–	4089	–	4211	–	4247
	10	11	12	13	14	15	16	17	
1	6806	5311	–	5327	5803	4716	5424	4097	
2	–	–	–	–	–	5570	4270	–	
3	–	5038	–	5027	–	4444	5120	3809	
4	–	–	–	–	–	5924	4704	5994	
5	5742	4357	–	4410	–	5136	–	3254	
6	2823	5128	–	–	–	–	–	–	
7	4954	4316	5678	4654	–	5363	–	3404	
8	3427	5697	4672	–	–	–	–	–	
9	4109	4061	5170	4723	–	5456	–	3490	
10	–	5316	4143	–	–	–	–	–	
11	4319	–	4642	3897	4191	4910	5899	3345	
12	3364	5253	–	5242	3281	6474	5248	–	
13	–	4108	5217	–	3436	4640	5375	3358	
14	–	4717	3792	3965	–	5170	3879	–	
15	–	4878	6187	4330	4445	–	4619	3993	
16	–	6104	5232	5390	3359	5339	–	–	
17	–	4142	–	4173	–	4851	–	–	

Table A.5: Number of points returned from fusion of two views from the wooden block dataset.

	1	2	3	4	5	6	7	8	9
1	–	0.35409	0.48738	0.43409	0.50202	0.54301	0.55004	–	0.5934
2	0.32342	–	0.47769	0.38269	0.52564	–	–	–	–
3	0.44333	0.50363	–	0.3883	0.46299	0.45608	0.47748	0.49768	0.55782
4	0.56986	0.56142	0.61549	–	0.55694	0.39794	–	–	–
5	0.47122	0.52959	0.45566	0.48776	–	0.32507	0.4656	0.50066	0.5
6	0.59764	–	0.56175	0.4862	0.52134	–	0.61583	0.52376	0.51685
7	0.49503	–	0.45352	–	0.45145	0.49397	–	0.38053	0.4679
8	–	–	0.50958	–	0.5368	0.41617	0.47105	–	0.5613
9	0.50624	–	0.54157	–	0.50687	0.5654	0.48115	0.51924	–
10	–	–	–	–	0.60697	–	0.56234	0.32887	0.42772
11	–	–	0.5508	–	0.56485	–	0.57276	0.57891	0.49742
12	–	–	–	–	0.52019	–	0.56956	–	0.51632
13	0.50483	–	0.53721	–	0.46042	–	0.57461	–	0.57374
14	–	–	–	–	–	–	–	–	0.67737
15	0.50173	0.53605	0.54587	0.49105	0.54913	–	0.48003	–	0.6382
16	0.55721	0.56833	0.49579	0.31927	–	–	–	–	–
17	0.4035	0.52557	0.4868	0.51545	0.32162	0.49963	0.44576	0.54575	0.36563
	10	11	12	13	14	15	16	17	
1	–	–	–	0.55572	–	0.47239	0.51772	0.41898	
2	–	–	–	–	–	0.50749	0.41381	0.33641	
3	–	0.58152	–	0.52104	–	0.48089	0.52401	0.48231	
4	–	–	–	–	–	0.56842	0.44053	0.55548	
5	0.54554	0.54186	0.6096	0.43577	–	0.52071	–	0.3498	
6	–	–	–	–	–	–	–	0.47547	
7	0.52787	0.53457	0.55091	0.45894	–	0.39851	–	0.45399	
8	0.43209	0.61003	–	–	–	–	–	0.58422	
9	0.38357	0.47527	0.45747	0.47741	0.45752	0.49114	–	0.35978	
10	–	0.56099	0.43794	0.59216	–	–	–	0.47246	
11	0.54724	–	0.37803	0.50972	0.51966	0.52308	–	0.55655	
12	0.46791	0.41232	–	0.52736	0.47778	–	–	0.52987	
13	0.58285	0.5816	0.54371	–	0.33076	0.49021	0.48147	0.42399	
14	–	0.56703	0.51688	0.42127	–	0.57028	0.51175	0.47343	
15	–	0.56083	–	0.48936	0.51431	–	0.34145	0.55034	
16	–	–	–	0.488	0.49125	0.44535	–	0.55373	
17	0.46495	0.54206	0.55583	0.33239	0.52786	0.49917	0.59439	–	

Table A.6: Mean distance error for registering pairs of views from the stone dataset.

	1	2	3	4	5	6	7	8	9
1	–	621	681	631	879	588	1080	–	1118
2	646	–	826	528	993	–	–	–	–
3	731	800	–	665	758	621	951	756	1050
4	576	481	608	–	707	234	–	–	–
5	898	944	769	718	–	625	757	703	901
6	585	–	593	225	577	–	669	231	682
7	1097	–	974	–	777	704	–	715	795
8	–	–	754	–	683	234	667	–	669
9	1112	–	1056	–	906	685	778	687	–
10	–	–	–	–	1003	–	914	508	722
11	–	–	1056	–	989	–	885	667	686
12	–	–	–	–	925	–	936	–	729
13	788	–	828	–	842	–	849	–	750
14	–	–	–	–	–	–	–	–	749
15	612	590	683	461	817	–	931	–	881
16	660	508	786	374	–	–	–	–	–
17	1012	1092	907	911	914	830	912	902	920
	10	11	12	13	14	15	16	17	
1	–	–	–	778	–	643	669	1016	
2	–	–	–	–	–	605	549	1138	
3	–	1056	–	841	–	726	786	902	
4	–	–	–	–	–	454	370	902	
5	1008	1006	921	847	–	825	–	874	
6	–	–	–	–	–	–	–	832	
7	931	934	940	895	–	940	–	894	
8	491	664	–	–	–	–	–	894	
9	0769	733	771	809	760	888	–	933	
10		635	541	665	–	–	–	1050	
11	652	–	584	667	641	801	–	944	
12	515	557	–	518	400	–	–	988	
13	670	634	497	–	376	451	444	860	
14	–	634	391	359	–	412	292	918	
15	–	792	–	458	421	–	406	863	
16	–	–	–	445	292	398	–	972	
17	1057	965	982	906	914	900	968	–	

Table A.7: Number of points returned from fusion of two views from the stone dataset.

Appendix B. Grasp Generation and Evaluation Results

	1	2	3	4	5	6	7	8	9
1	–	2256	2652	2070	4556	1640	6642	–	7656
2	2162	–	4160	1482	5700	–	–	–	–
3	3306	3782	–	2450	3660	2352	5700	3192	6972
4	1482	1190	2162	–	2756	156	–	–	–
5	5256	5550	3906	2652	–	2352	3540	2862	5700
6	1722	–	2070	30	2070	–	2550	210	2862
7	7310	–	6006	–	4290	3192	–	3306	3906
8	–	–	2970	–	2352	182	2162	–	2550
9	8556	–	7140	–	5256	2652	3422	2862	–
10	–	–	–	–	6806	–	5402	1332	2756
11	–	–	6162	–	5256	–	4422	2450	2256
12	–	–	–	–	5256	–	5700	–	3306
13	3660	–	4032	–	4032	–	4290	–	3192
14	–	–	–	–	–	–	–	–	3306
15	1806	1892	2352	812	3660	–	5256	–	4830
16	2450	1560	3540	552	–	–	–	–	–
17	6006	7310	5256	4160	4970	3192	4692	4422	4830
	10	11	12	13	14	15	16	17	
1	–	–	–	3192	–	2450	2352	6480	
2	–	–	–	–	–	2256	1892	7832	
3	–	6806	–	3906	–	3192	3660	5700	
4	–	–	–	–	–	702	462	3782	
5	6320	6162	4830	4290	–	3660	–	4422	
6	–	–	–	–	–	–	–	3540	
7	5700	5256	5700	4830	–	5256	–	4556	
8	1482	2450	–	–	–	–	–	4422	
9	2862	2652	3422	3540	3422	4290	–	5112	
10	–	2162	1332	2162	–	–	–	6480	
11	2352	–	1722	2256	2070	3422	–	4422	
12	1122	1406	–	1122	702	–	–	5700	
13	2162	1560	1190	–	600	756	650	3660	
14	–	1980	702	420	–	650	156	4692	
15	–	3782	–	930	812	–	600	4160	
16	–	–	–	650	272	650	–	4556	
17	6162	4830	5700	4422	4422	5112	5112	–	

Table B.1: Number of grasps considered for each pair of views from the stone data set.

	1	2	3	4	5	6	7	8	9
1	–	–	–	–	–	–	16	–	44
2	–	–	–	–	2	–	–	–	–
3	–	2	–	–	–	–	–	–	–
4	–	–	–	–	–	–	–	–	–
5	–	2	–	–	–	–	–	–	–
6	–	–	–	–	–	–	–	–	–
7	12	–	–	–	–	–	–	–	–
8	–	–	–	–	–	–	–	–	–
9	56	–	–	–	–	–	–	–	–
10	–	–	–	–	–	–	–	–	–
11	–	–	2	–	2	–	–	2	–
12	–	–	–	–	–	–	–	–	–
13	–	–	–	–	2	–	–	–	–
14	–	–	–	–	–	–	–	–	–
15	–	–	–	–	–	–	–	–	–
16	–	–	–	–	–	–	–	–	–
17	–	10	–	–	–	–	–	–	–

	10	11	12	13	14	15	16	17	
1	–	–	–	–	–	–	–	–	
2	–	–	–	–	–	–	–	14	
3	–	–	–	–	–	–	–	–	
4	–	–	–	–	–	–	–	–	
5	2	–	6	6	–	–	–	–	
6	–	–	–	–	–	–	–	–	
7	–	–	2	–	–	–	–	–	
8	–	–	–	–	–	–	–	–	
9	–	–	–	–	–	–	–	–	
10	–	–	–	–	–	–	–	10	
11	–	–	–	–	–	–	–	–	
12	–	–	–	–	–	–	–	–	
13	–	–	–	–	–	–	–	–	
14	–	–	–	–	–	–	–	–	
15	–	–	–	–	–	–	–	–	
16	–	–	–	–	–	–	–	–	
17	8	–	4	–	–	–	–	–	

Table B.2: Number of grasps exhibiting the force closure property for each pair of views from the stone data set.

	1	2	3	4	5	6	7	8	9
1	–	–	–	–	–	–	0.39151	–	0.33811
2	–	–	–	–	0.10084	–	–	–	–
3	–	0.26585	–	–	–	–	–	–	–
4	–	–	–	–	–	–	–	–	–
5	–	0.20138	–	–	–	–	–	–	–
6	–	–	–	–	–	–	–	–	–
7	0.47739	–	–	–	–	–	–	–	–
8	–	–	–	–	–	–	–	–	–
9	0.3067	–	–	–	–	–	–	–	–
10	–	–	–	–	–	–	–	–	–
11	–	–	0.23799	–	0.21261	–	–	0.34626	–
12	–	–	–	–	–	–	–	–	–
13	–	–	–	–	0.35006	–	–	–	–
14	–	–	–	–	–	–	–	–	–
15	–	–	–	–	–	–	–	–	–
16	–	–	–	–	–	–	–	–	–
17	–	0.5254	–	–	–	–	–	–	–
	10	11	12	13	14	15	16	17	
1	–	–	–	–	–	–	–	–	
2	–	–	–	–	–	–	–	0.23319	
3	–	–	–	–	–	–	–	–	
4	–	–	–	–	–	–	–	–	
5	0.3681	–	0.53941	0.51499	–	–	–	–	
6	–	–	–	–	–	–	–	–	
7	–	–	0.1487	–	–	–	–	–	
8	–	–	–	–	–	–	–	–	
9	–	–	–	–	–	–	–	–	
10	–	–	–	–	–	–	–	0.5889	
11	–	–	–	–	–	–	–	–	
12	–	–	–	–	–	–	–	–	
13	–	–	–	–	–	–	–	–	
14	–	–	–	–	–	–	–	–	
15	–	–	–	–	–	–	–	–	
16	–	–	–	–	–	–	–	–	
17	0.46989	–	0.32221	–	–	–	–	–	

Table B.3: Maximum grasp quality for each pair of views from the stone data set. The best two view grasp is found when views 17 and 10 are registered.

	1	2	3	4	5	6	7	8	9
1	–	27060	23562	31862	24180	36672	54990	–	62750
2	32942	–	36672	29070	33306	36290	75350	–	82082
3	18360	27722	–	22350	17556	26082	43890	62250	55932
4	30800	20592	29070	–	27060	24492	62750	60270	69960
5	23256	32220	13340	28392	–	9312	15750	31506	26732
6	37830	35910	26082	25760	11130	–	28730	25122	39402
7	56882	74256	44732	59780	18632	28730	–	24806	18360
8	77006	–	60270	63756	29412	23870	24492	–	27390
9	62750	78120	55460	74256	28056	31862	24492	31152	–
10	82656	–	68906	–	36290	37056	33672	29756	32580
11	43472	62250	37442	59292	22952	32580	22650	30800	19182
12	64262	64262	–	–	32220	–	31862	27722	33306
13	34782	43056	34040	51302	17556	27722	32942	41412	33306
14	33672	32220	28056	37830	14762	–	32580	32580	33306
15	20592	25440	20592	32580	20592	31506	45156	58322	39800
16	28056	20022	29070	28056	28730	31506	–	–	59292
17	29070	37830	23256	41412	10100	19740	23870	41006	27390
	10	11	12	13	14	15	16	17	
1	80940	44732	60762	31506	34782	22952	34410	26732	
2	–	58806	62750	46010	36672	36290	31152	43890	
3	72092	38220	–	31862	31152	17822	29070	20592	
4	–	60762	–	47306	38220	36290	30102	39402	
5	35532	21462	32220	17292	15252	18906	–	8190	
6	35156	34410	–	28730	–	32580	32580	20022	
7	25760	21756	29070	35532	32942	40200	–	22650	
8	19182	28730	24492	51302	33306	60762	–	39402	
9	30102	22350	34782	34410	32220	43056	62750	26082	
10	–	40602	30102	48180	30102	61752	62250	45582	
11	27390	–	22650	17822	17822	26732	39006	19740	
12	21462	30102	–	29412	13572	46440	36672	35910	
13	41412	23256	30102	–	11342	21170	32220	15006	
14	34040	22052	12656	12656	–	23256	16002	18360	
15	62250	28392	45156	16770	20592	–	23562	18632	
16	58322	42230	37442	29412	14280	29070	–	34040	
17	43472	19182	34410	16002	18090	18360	32942	–	

Table B.4: Number of grasps considered for each pair of views from the cow data set.

	1	2	3	4	5	6	7	8	9
1	–	–	–	14	–	34	702	–	1074
2	–	–	2	4	28	16	1078	–	1390
3	–	4	–	4	–	4	398	730	618
4	6	–	4	–	16	2	994	1314	1174
5	4	24	2	18	–	2	2	22	10
6	30	12	4	4	–	–	2	4	46
7	684	1014	388	918	2	10	–	–	–
8	1068	–	754	1368	22	14	–	–	26
9	1074	1456	654	1324	–	46	–	16	–
10	1448	–	822	–	34	30	2	4	–
11	580	932	374	854	70	218	40	158	–
12	1038	1362	–	–	338	–	252	372	22
13	22	58	202	510	298	790	278	490	20
14	22	30	312	774	388	–	316	746	34
15	–	4	42	100	42	216	426	782	540
16	10	–	212	470	342	632	–	–	832
17	4	40	2	58	–	28	2	32	–
	10	11	12	13	14	15	16	17	
1	1434	462	1014	6	30	–	26	–	
2	–	734	1408	52	26	4	2	50	
3	858	386	–	224	302	48	234	–	
4	–	912	–	552	820	162	398	44	
5	36	62	402	322	406	50	–	–	
6	26	192	–	910	–	248	586	32	
7	2	64	232	276	316	436	–	2	
8	6	110	376	642	780	794	–	64	
9	–	–	30	6	26	576	958	–	
10	–	4	8	38	32	802	1182	54	
11	–	–	–	4	–	322	548	8	
12	4	–	–	14	2	626	974	88	
13	54	6	14	–	–	–	16	30	
14	28	–	–	–	–	2	–	36	
15	816	320	622	–	2	–	–	2	
16	1106	544	922	16	2	–	–	64	
17	40	–	66	10	48	16	58	–	

Table B.5: Number of grasps exhibiting the force closure property for each pair of views from the cow data set.

	1	2	3	4	5	6	7	8	9
1	–	–	–	0.64415	–	0.59242	0.84641	–	0.87147
2	–	–	0.42482	0.45315	0.6746	0.53243	0.85513	–	0.8964
3	–	0.5034	–	0.40295	–	0.5873	0.81849	0.80084	0.85142
4	0.49919	–	0.4422	–	0.5849	0.51361	0.80887	0.86062	0.86832
5	0.51044	0.58342	0.54115	0.55896	–	0.50138	0.59144	0.62123	0.50931
6	0.62045	0.48245	0.51864	0.61194	–	–	0.5195	0.57492	0.62086
7	0.86425	0.86653	0.83805	0.81723	0.56094	0.63741	–	–	–
8	0.87164	–	0.84147	0.86594	0.59179	0.62004	–	–	0.60759
9	0.86869	0.87666	0.8548	0.87387	–	0.59174	–	0.54238	–
10	0.89216	–	0.83803	–	0.66293	0.63364	0.43846	0.44498	–
11	0.87139	0.90283	0.88697	0.90059	0.86929	0.82622	0.7058	0.73614	–
12	0.86553	0.85008	–	–	0.86036	–	0.78543	0.7535	0.49882
13	0.61496	0.74205	0.8587	0.86144	0.91883	0.84094	0.90022	0.85519	0.43061
14	0.30469	0.55062	0.86108	0.77759	0.90626	–	0.84219	0.78418	0.31982
15	–	0.37109	0.72258	0.75244	0.83214	0.83574	0.82232	0.8537	0.86686
16	0.36172	–	0.81116	0.80422	0.8285	0.78024	–	–	0.89299
17	0.45511	0.70185	0.51554	0.68006	–	0.50492	0.46024	0.60643	–
	10	11	12	13	14	15	16	17	
1	0.89982	0.856	0.88623	0.40718	0.6219	–	0.63252	–	
2	–	0.86052	0.92221	0.77223	0.68283	0.43738	0.4583	0.67979	
3	0.85536	0.79161	–	0.8556	0.87549	0.73072	0.8511	–	
4	–	0.83036	–	0.83606	0.85175	0.70193	0.81492	0.64336	
5	0.57058	0.87259	0.82976	0.87692	0.91038	0.85681	–	–	
6	0.58199	0.83064	–	0.86461	–	0.84521	0.83628	0.52495	
7	0.46784	0.8059	0.84223	0.88577	0.83949	0.88422	–	0.4043	
8	0.44557	0.7137	0.84581	0.87068	0.83384	0.82798	–	0.63767	
9	–	–	0.60912	0.42919	0.52691	0.87619	0.91715	–	
10	–	0.43884	0.62082	0.79683	0.63055	0.88393	0.89929	0.64434	
11	–	–	–	0.47082	–	0.87253	0.9207	0.45027	
12	0.48677	–	–	0.63898	0.47832	0.8377	0.80992	0.68899	
13	0.68679	0.49112	0.73039	–	–	–	0.72802	0.56123	
14	0.59692	–	–	–	–	0.20742	–	0.32163	
15	0.8781	0.86297	0.91587	–	0.41953	–	–	0.41392	
16	0.8869	0.83013	0.79576	0.71573	0.39336	–	–	0.64164	
17	0.66141	–	0.70004	0.57568	0.65683	0.58637	0.71329	–	

Table B.6: Maximum grasp quality for each pair of views from the cow data set. The best two view grasp is found when views 12 and 2 are registered.

	1	2	3	4	5	6	7	8	9
1	–	141752	149382	218556	214832	242556	197580	390000	215760
2	164430	–	182756	122850	–	–	–	–	–
3	93942	89102	–	121452	115940	130682	121452	270920	167690
4	212060	98910	194040	–	211140	87320	227052	225150	–
5	218556	–	142506	203852	–	–	83232	175980	144780
6	240590	–	178506	109230	–	–	120062	104652	193160
7	206570	–	146306	199362	74256	77006	–	131406	105300
8	–	–	305256	222312	197580	92720	192282	–	234740
9	239610	–	180200	336980	163620	196692	135056	202950	–
10	–	–	–	–	–	–	202950	115260	159600
11	197580	–	150156	–	122850	159600	107912	190532	95172
12	–	–	–	–	–	–	257556	184470	223256
13	230880	–	185330	–	158802	–	142506	–	171810
14	–	–	–	–	–	–	–	–	–
15	122150	118680	129240	198470	176820	–	156420	–	193160
16	217622	93330	224202	173472	–	–	–	–	–
17	167690	–	128522	–	105300	–	99540	–	116622
	10	11	12	13	14	15	16	17	
1	325470	182756	–	216690	241572	175142	227052	129960	
2	–	–	–	–	–	204756	127092	–	
3	–	138756	–	165242	–	120756	171810	90902	
4	–	–	–	–	–	256542	177662	249500	
5	228962	116622	–	148610	–	175980	–	73712	
6	51756	154842	–	–	–	–	–	–	
7	148610	100806	218556	135056	–	159600	–	69432	
8	97032	235710	178506	–	–	–	–	–	
9	126380	113232	204756	171810	–	191406	–	86142	
10	–	177662	120756	–	–	–	–	–	
11	115260	–	148610	88506	113906	141752	243542	64770	
12	85556	202050	–	219492	85556	303050	214832	–	
13	–	112560	220430	–	93942	152490	217622	88506	
14	–	141752	105300	109230	–	177662	107912	–	
15	–	133590	271962	127092	131406	–	132132	102720	
16	–	273006	234740	234740	93942	226100	–	–	
17	–	89700	–	113906	–	131406	–	–	

Table B.7: Number of grasps considered for each pair of views from the wooden block data set.

	1	2	3	4	5	6	7	8	9
1	–	–	–	–	–	–	2352	16806	11466
2	–	–	–	–	–	–	–	–	–
3	–	–	–	–	–	–	1200	10284	6850
4	–	–	–	–	–	–	1948	16536	–
5	–	–	–	–	–	–	–	–	–
6	–	–	–	–	–	–	–	–	–
7	2196	–	1258	1890	–	–	–	–	–
8	–	–	10686	16098	–	–	–	–	–
9	12476	–	6570	11196	–	–	–	–	–
10	–	–	–	–	–	–	–	–	–
11	4546	–	3466	–	3728	5288	2038	4738	–
12	–	–	–	–	–	–	7706	19360	–
13	–	–	3760	–	15626	–	9094	–	–
14	–	–	–	–	–	–	–	–	–
15	–	–	1684	7796	7350	–	4880	–	5844
16	–	–	3478	16288	–	–	–	–	–
17	–	–	–	–	–	–	–	–	–
	10	11	12	13	14	15	16	17	
1	21108	4680	–	–	–	–	–	–	
2	–	–	–	–	–	–	–	–	
3	–	3694	–	3582	–	1820	3646	–	
4	–	–	–	–	–	7972	17010	–	
5	–	3716	–	14972	–	7330	–	–	
6	–	5110	–	–	–	–	–	–	
7	–	2226	8128	7934	–	5078	–	–	
8	–	4242	18348	–	–	–	–	–	
9	–	–	–	–	–	5592	–	–	
10	–	–	–	–	–	–	–	–	
11	–	–	–	–	–	2264	4202	–	
12	–	–	–	–	–	8468	15446	–	
13	–	–	–	–	34	–	8	–	
14	–	–	–	–	–	–	–	–	
15	–	2144	8774	–	–	–	–	–	
16	–	4236	16338	–	–	–	–	–	
17	–	–	–	–	–	–	–	–	

Table B.8: Number of grasps exhibiting the force closure property for each pair of views from the wooden block data set.

	1	2	3	4	5	6	7	8	9
1	–	–	–	–	–	–	0.86596	0.92765	0.90769
2	–	–	–	–	–	–	–	–	–
3	–	–	–	–	–	–	0.88237	0.919	0.91337
4	–	–	–	–	–	–	0.83594	0.94471	–
5	–	–	–	–	–	–	–	–	–
6	–	–	–	–	–	–	–	–	–
7	0.81587	–	0.88593	0.85438	–	–	–	–	–
8	–	–	0.92198	0.93433	–	–	–	–	–
9	0.90378	–	0.89251	0.90313	–	–	–	–	–
10	–	–	–	–	–	–	–	–	–
11	0.94865	–	0.92209	–	0.91504	0.89341	0.88119	0.89515	–
12	–	–	–	–	–	–	0.92279	0.93498	–
13	–	–	0.84881	–	0.93839	–	0.93156	–	–
14	–	–	–	–	–	–	–	–	–
15	–	–	0.85866	0.94389	0.95661	–	0.92401	–	0.84607
16	–	–	0.72972	0.93523	–	–	–	–	–
17	–	–	–	–	–	–	–	–	–
	10	11	12	13	14	15	16	17	
1	0.93685	0.90756	–	–	–	–	–	–	
2	–	–	–	–	–	–	–	–	
3	–	0.96703	–	0.83956	–	0.85854	0.84485	–	
4	–	–	–	–	–	0.93753	0.9501	–	
5	–	0.95888	–	0.93524	–	0.9084	–	–	
6	–	0.92387	–	–	–	–	–	–	
7	–	0.89571	0.94647	0.90286	–	0.90614	–	–	
8	–	0.89646	0.95808	–	–	–	–	–	
9	–	–	–	–	–	0.83908	–	–	
10	–	–	–	–	–	–	–	–	
11	–	–	–	–	–	0.923	0.95099	–	
12	–	–	–	–	–	0.88249	0.93514	–	
13	–	–	–	–	0.54098	–	0.37195	–	
14	–	–	–	–	–	–	–	–	
15	–	0.92102	0.89569	–	–	–	–	–	
16	–	0.91971	0.92149	–	–	–	–	–	
17	–	–	–	–	–	–	–	–	

Table B.9: Maximum grasp quality for each pair of views from the wooden block data set. The best two view grasp is found when views 11 and 3 are registered.

Appendix C. Overall Results

As the system is entirely deterministic once the initial viewing position has been chosen, and there are only eight initial viewing positions (the initial elevation was set to 45°), each object can only be tested eight times. The results are shown below.

Initial azimuth	Total views	Best grasp after two views	Best grasp after three views	Final grasp quality	Number of patches in initial view	
0°	2	0.91397	–	0.91397	367	
45°	2	0.91764	–	0.91764	293	
90°	3	0	0.93783	0.93783	265	
135°	2	0.84368	–	0.84368	260	
180°	2	0.91747	–	0.91747	303	
225°	2	0.95023	–	0.95023	254	
270°	2	0.94127	–	0.94127	294	
315°	11	0	0	0.92004	318	
	Second view	Number of votes	Number of patches in second view	Third View	Number of votes	Number of patches in third view
0°	180°/45°	205/299	465	–	–	–
45°	180°/45°	125/283	410	–	–	–
90°	0°/45°	122/263.5	468	180°/90°	202/403.5	677
135°	0°/45°	144/256.5	455	–	–	–
180°	0°/45°	249/301.5	490	–	–	–
225°	0°/45°	150/254	445	–	–	–
270°	90°/45°	126/293	399	–	–	–
315°	0°/45°	122/310	350	180°/90°	121/282	n/a

Table C.1: Summary of results for the wooden block data set.

Initial azimuth	Total views	Best grasp after two views	Best grasp after three views	Final grasp quality	Number of patches in initial view	
0°	3	0.64512	0.79824	0.79824	40	
45°	5	0	0.74447	0.83714	48	
90°	6	0.61699	0.63898	0.79376	49	
135°	5	0	0	0.82575	57	
180°	3	0.64443	0.80140	0.80140	48	
225°	12	0	0	0.85057	41	
270°	3	0	0.79577	0.79577	22	
315°	5	0	0	0.84321	22	
	Second view	Number of votes	Number of patches in second view	Third View	Number of votes	Number of patches in third view
0°	180°/45°	26/40	88	180°/90°	14.5/61.5	100
45°	180°/45°	15/48	84	0°/45°	23/72.5	100
90°	270°/45°	17/49	66	225°/90°	15/54.5	79
135°	315°/45°	20/57	73	270°/90°	18/60.5	80
180°	0°/45°	24/48	93	180°/90°	14/66.5	99
225°	0°/45°	19/41	n/a	45°/90°	13/31.5	n/a
270°	45°/45°	7/22	64	180°/90°	15/54	87
315°	135°/45°	10/22	73	270°/90°	19/60.5	80

Table C.2: Summary of results for the stone data set.

Initial azimuth	Total views	Best grasp after two views	Best grasp after three views	Final grasp quality	Number of patches in initial view	
0°	2	0.87764	–	0.87764	150	
45°	2	0.89818	–	0.89818	134	
90°	2	0.90454	–	0.90454	68	
135°	2	0.89993	–	0.89993	133	
180°	2	0.87764	–	0.87764	150	
225°	2	0.87793	–	0.87793	126	
270°	2	0.92538	–	0.92538	107	
315°	2	0.87027	–	0.87027	127	
	Second view	Number of votes	Number of patches in second view	Third View	Number of votes	Number of patches in third view
0°	180°/45°	101/143	251	–	–	–
45°	180°/45°	73/133.5	237	–	–	–
90°	270°/45°	20/67.5	132	–	–	–
135°	0°/45°	72/132.5	239	–	–	–
180°	0°/45°	101/146.5	251	–	–	–
225°	0°/45°	66/126	209	–	–	–
270°	90°/45°	52/107	133	–	–	–
315°	180°/45°	63/125.5	200	–	–	–

Table C.3: Summary of results for the cow data set.

Bibliography

[1] 3D Scanners (UK), http://www.3dscanners.co.uk

[2] Bergevin, R., Soucy, M., Gagnon, H. and Laurendeau, D. "Towards a general multi-view registration technique", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 18, no. 5, pp 540–547, May 1992

[3] Besl, P. J., and McKay, N. D. "A Method for Registration of 3-D Shapes", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 14, no. 2, pp 239–256, Feb 1992

[4] Blais, G. and Levine, M. D. "Registering Multiview Range Data to Create 3D Computer Objects", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 17, no. 8, pp 820–824, Aug 1993

[5] Brujic, D. and Ristic, M. "Analysis of Free Form Surface Registration", Proc. IEEE Int. Conf. on Image Processing, vol. 2, pp 393–396, 1996

[6] Chen, Y. and Medioni, G. G. "Object modelling by registration of multiple range images", Image and Vision Computing, vol. 10, no. 3, pp 145–155, Apr 1992

[7] Chinellato, E., Fisher, R.B., Morales, A. and del Pobil, A.P. "Ranking planar grasp configurations for a three-finger hand", Proc. IEEE Int. Conf. on Robotics and Automation, Taipei, pp 1133–1138, September 2003

[8] Connolly, C. J. "The determination of the next best views", Proc. IEEE Int. Conf. on Robotics and Automation, St. Louis, pp 432–435, October 1985

[9] Eggert, D., Fitzgibbon, A. W. and Fisher, R. B. "Simultaneous registration of multiple range views for use in reverse engineering", Proc. Int. Conf. on Pattern Recognition, Vienna, pp 243–247, Aug 1996

[10] Fisher, R. B. Advanced Vision, http://www.inf.ed.ac.uk/teaching/modules/av/MATLAB/TASK3/

[11] Garcia, M. A., Velazquez, S. and Sappa, A. D. "A Two-Stage Algorithm for Planning the Next View From Range Images", Proc. British Machine Vision Conference BMVC98, Southampton, pp 720–729, September 1998

[12] Hoppe, H., DeRose, T., Duchamp, T., McDonald, J. and Stuetzle, W. "Surface reconstruction from unorganized points", Proc. SIGGRAPH '92, Chicago, pp 71–78, Jul 1992

[13] Horn, B. K. P. "Extended Gaussian Images", Proc. IEEE, vol. 72, no. 12, pp 1671–1686, December 1984

[14] Massios, N. A. and Fisher, R. B. "A Best Next View Selection Algorithm Incorporating a Quality Criterion", Proc. British Machine Vision Conference BMVC98, Southampton, pp 780–789, September 1998

[15] Masuda, T. "Object Shape Modelling from Multiple Range Images by Matching Signed Distance Fields", Proc. 1st Int. Symposium on 3D Data Processing, Visualisation and Transmission, Padova, pp 439–448, Jun 2002

[16] MathWorld, `http://mathworld.wolfram.com/`

[17] Mian, A. S. "Automated 3D Model-Based Free-Form Object Recognition", `http://www.csse.uwa.edu.au/~ajmal/`

[18] Montana, D. J. "Contact Stability for Two-Fingered Grasps", IEEE Transactions on Robotics and Automation, vol. 8, no. 4, Aug 1992

[19] Nguyen, V.-D. "Constructing Force Closure Grasps", Int. Journal of Robotic Research, vol. 7, no. 3, pp 3–16, 1988

[20] Sanchiz, J. M. and Fisher, R. B. "A next-best-view algorithm for 3D scene recovery with 5 degrees of freedom", Proc. British Machine Vision Conference BMVC99, Nottingham, pp 163–172, September 1999

[21] Sarcos, Inc., University of Utah, MIT. Utah/MIT Dextrous hand, `http://www.sarcos.com/telespec.dexhand.html`

[22] Smith, G., Lee, E., Goldberg, K., Bohringer, K., Craig, J. "Computing Parallel-Jaw Grip Points", IEEE Int. Conf. on Robotics and Automation (ICRA), Detroit, pp 1897–1903, May 1999

[23] Soucy, M. and Laurendeau, D. "Surface Modelling from dynamic integration of multiple range views", Proc. 11th Int. Conf. on Pattern Recognition, The Hague, pp 449–452, Sep 1992

[24] Stuttgart Range Image Database, `http://range.informatik.uni-stuttgart.de/`

[25] Turk, G. and Levoy, M. "Zippered Polygon Meshes from Range Images", Proc. SIGGRAPH '94, Orlando, pp 311–318, Jul, 1994

[26] Wren, D. and Fisher, R. B. "Dextrous hand grasping strategies using preshapes and digit trajectories", Proc. IEEE Int. Conf. on Systems, Man and Cybernetics, Vancouver, Vol 1, pp 910–915, October 1995

[27] Zhang, Z. "Iterative Point Matching for Registration of Free-Form Curves and Surfaces", Int. Journal of Computer Vision, vol. 13, no. 2, pp 119–152, Oct 1994