Kinematic Enveloping Grasp Planning Method for Robotic Dexterous Hands and Three-Dimensional Objects

Kinematic Enveloping Grasp Planning Method for Robotic Dexterous Hands and Three-Dimensional Objects

By

SHAHRAM SALIMI, M.S.

A Thesis

Submitted to the School of Graduate Studies

in Partial Fulfillment of the Requirements

for the Degree

Master of Applied Science

McMaster University

Masters of Engineering (2005) McMaster University

Mechanical Engineering Hamilton, Ontario

TITLE: Kinematic Enveloping Grasp Planning Method for Robotic Dexterous Hands and Three-Dimensional Objects.

AUTHOR: Shahram Salimi

(University of Tehran, Iran)

SUPERVISOR: Dr. Gary M. Bone, Associate Professor.

NUMBER OF PAGES:

Abstract

Three-dimensional (3D) enveloping grasps for dexterous robotic hands possess several advantages over other types of grasps. However, their innate characteristics such as the several degrees of freedom of the dexterous hand, complexity of analyzing the 3D geometry of the object to be grasped or detecting the 3D contact points between the object and the hand make planning them automatically a very challenging problem. This thesis describes a new method for kinematic 3D enveloping grasp planning for a three-fingered dexterous hand.

The required inputs are the geometric models of the object and hand; and the kinematic model of the hand. The outputs are the position and orientation of the palm and the angular joint positions of the fingers. The method introduces a new way of processing the 3D object. Instead of considering the object as a whole, a series of 2D slices (vertical and horizontal) of the object are used to define its geometry. This method is considerably simpler than other methods of object modeling and its parameters can be easily setup.

A new idea for grading the object's 3D grasp search domain is proposed. The grading system analyzes the curvature pattern and thickness of the object and grades object regions according to their suitability for grasping. The proposed method is capable of eliminating most of the ungraspable areas of the object from the grasp search domain at the early stages of the search. This improves the overall efficiency of the search for a grasp.

In modeling a dexterous hand a new method is proposed to model the fingers. In this model each finger is modeled by three articulated line segments, representing the top, centre and bottom of the finger. This model has significant benefits that it is efficient and does not need the exact coordinate of the 3D contact point between the finger and the object to analyze the feasibility of the grasp.

The new grasp planning method was implemented by writing a 4300 line MATLAB program. The program has been run successfully with several 3D objects. These results are documented.

To my family, Khosrow, Efat, Anahita and Keyvan,

whom my heart beat is mere an echo of theirs.

Acknowledgements

My greatest sincere thanks to my research supervisor, Dr. Gary M. Bone, whose invaluable guidance and moral support were always on my side in all the hardships in two years of my study in McMaster. I am so grateful to have this opportunity of sharing this time with him.

I also would like to thank all the professors, colleagues and staff at Mechanical Engineering Department, specially Zhihong Rao for his invaluable friendship.

Table of Content

List of Figures

List of Tables

Chapter 1: Introduction

It has not been a long time since the interdisciplinary field of robotics was introduced to the world[1]. In this short period robotics has shown its huge potential for handling a variety of tasks. Now robots not only belong to the physical world around people but they have even opened the way to improved forms of medicine and surgery. It is hard to think of any kind of automated manufacturing system without robots used in some part of it. Although robots have already established their value and practicality in many fields, they are still far from what the researchers are dreaming to achieve, a world of intelligent robots and humanoid robots.

One principal unsolved problem is how a robot interacts with its outside world. Creating some ways of understanding the world around it (i.e. seeing, hearing) and interacting appropriately, (i.e. moving, touching and grasping) are still huge challenges for researchers. In the former, researchers have developed more advanced and practical sensors; and faster and more accurate signal

[1] The first industrial robot was bought by GM in the 1960s and the subject of robotics did not appear at most universities until the 1980s.

processing algorithms. A robot with sophisticated sensing capabilities but without the ability to act to change its environment is not a satisfactory solution. One of the main duties of an autonomous robot would be to touch, grasp objects and manipulate objects. But researchers have not yet found a universal answer to what is the best method to implement this autonomous behaviour.

1.1 Grasping

Grasping is a difficult problem in robotics[1]. In general, identifying suitable contact locations, hand pose, and force exertion strategies requires satisfying three main sets of constraints: constraints due to limited capabilities of the gripper or dexterous hand; constraints due to object geometric features and material characteristics; and constraints due to the task requirements. In analyzing a grasp it is hard to separate these constraints from each other. A successful grasp is typically accomplished by reasonably satisfying all of these constraints together, which is not always possible. The grasping procedure should decide between many tradeoffs. For instance the best contact locations do not always lead to the best force conditions.

A human mind is able to analyze the slightest details of an object and decide between many possibilities to select the best condition of the grasp according to the understanding it has of the hand and object's physical properties. With

[1] Even humans sometimes have difficulty grasping some objects, for example a wet bar of soap.

existing engineering knowledge this is impossible to duplicate with a robot. So in order to design the systems, researchers are forced to do many simplifications. Most of the grasping systems are task based and are focused on the range and type of the objects the robot should grasp. The simplest strategies that cover the minimum needs of the grasping task are chosen, which in the end limits the robot to highly structured environments requiring minimal autonomous behaviour.

1.2 Jaw Grippers

A lot of gripper types have been developed for robot grasping but the most popular type are termed jaw grippers. Fig.1.1 shows a set of common versions of this type.

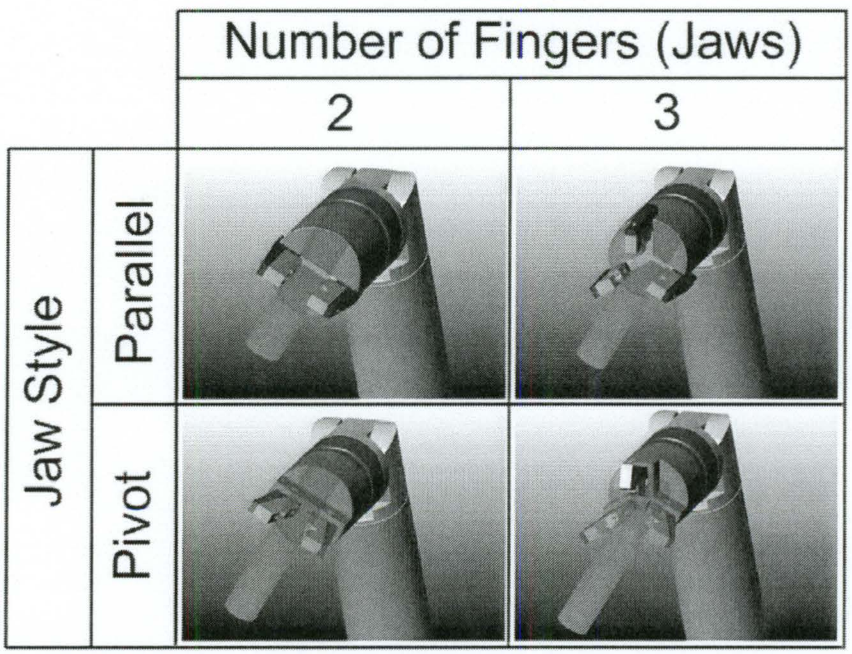

Fig.1.1. Jaw grippers [1]

These grippers have the advantage of being simple in design, making them inexpensive and reliable. However they usually have only one degree of freedom (DOF). This limits the variety of grasps they can execute. Furthermore, they usually require the shape of the jaw to be custom designed for grasping a particular object. This means that multiple grippers and tool changers must be used to allow the robot to grasp several different objects. This approach loses its reliability and cost-effectiveness when the number of objects increases. The fixed jaw shape, and single DOF, makes jaw grippers a poor design choice for intelligent robots.

1.3 Dexterous Hands

To overcome the limitations of jaw grippers, researchers introduced dexterous hands. This type of gripper includes a wide variety of designs but has some common characteristics:

- They are designed to do 3D grasping and often dexterous manipulation as well.
- The goal is to make them capable of gripping almost any generic object.
- They can have many DOF, depending on their mechanism.
- Designers have often tried to make them similar to the human hand. Usually they have fingers similar in shape to the human hand, but that may vary in the number of the fingers per hand and number of phalanges for each finger.

Two common types of grasps implemented with dexterous hands are fingertip grasps (see Fig.1.2) and enveloping grasps (see Fig.1.3).

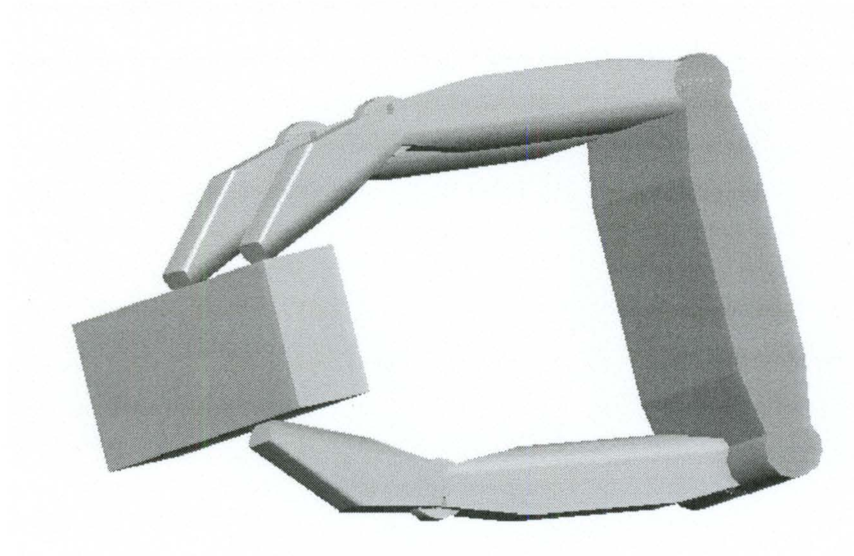

Fig.1.2. Fingertip grasp using a dexterous hand.

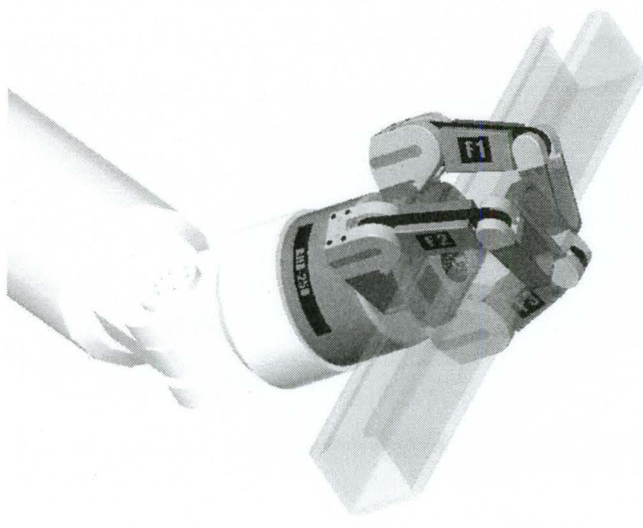

Fig.1.3 Enveloping grasp using a dexterous hand [1].

Between these two types, the enveloping grasp is more desirable because:

- It is more robust to errors in positioning of the fingers or the object and force control errors.
- It relies less on friction to constrain the object.
- It is compatible with simple robotic hands.

However enveloping grasps are particularly difficult to plan for 3D objects for the following reasons:

- Locating the palm and fingers of an N DOF hand for an enveloping grasp requires searching a 6+N dimensional space.
- The physical limits such as the length of the finger phalanges, or the range of motion for joint, add complex constraints to the search.
- The contact between the object surface and the surfaces of the palm and fingers is much more complex to model and analyze than fingertip contact.
- The force and moment analysis is a statically indeterminate problem.

This thesis is focused on enveloping grasp planning with dexterous hands. Only the kinematic aspects (i.e. not the force and moment aspects) are investigated. The thesis has been laid out as follows:

- In Chapter 2, a literature review is presented with an emphasis on the kinematic aspects of grasp planning.
- In Chapter 3, the theory and method of the grasp planning is presented.

- In Chapter 4, a numerical example for a complex shaped object is presented in detail.
- In Chapter 5, additional numerical examples are presented to demonstrate the generality of the grasp planning method.
- Chapter 6 includes the conclusions and recommendations for future work.

Chapter 2: Literature Review

In this chapter the relevant literature from the field of robotic grasp planning is reviewed. The emphasis is on the kinematic aspects of grasp planning.

2.1. Related work

Cutkosky [2] created a grasp taxonomy to categorize particular grasps and trace how they have been derived from generic human grasp types. Grasps are classified by the task requirements (e.g. forces and motions that must be imparted), properties of the grasped object (e.g. the shape, and slipperiness) and constraints from the hand or gripper (e.g. the maximum grasp force). The paper tries to classify different grasps the way the human mind does. Based on human grasping patterns, a concept is proposed for grasp planning within a flexible fully automated manufacturing system. The work presents an interesting idea, but for common manufacturing systems the type of object and grasp type are already known and the challenge is more focused on how to do that particular grasp more efficiently and effectively.

Wren and Fisher [3] used preshape patterns for grasp planning with a dexterous hand. They defined a group of grasp strategies from task-specified sets of preshapes. Basically they divided the grasp patterns in four categories of preshapes: precision, lateral, manipulation and hook. For each preshape, a pattern of movement for the joints of the hand is reserved. Based on the object to grasp, the type of the grasp should be decided. This method has this benefit that it can simplify the grasp planning, but it provides no information on how the wrist should approach the object. They also do not discuss how an enveloping grasp can be implemented by this routine.

Asada and Kitagawa [4] introduced a kinematical formulation for form closure grasps. With a form closure grasp the object is restrained by the hand regardless of the squeezing forces or coefficient of friction. In other words it is kinematically restrained. In their algorithm the form closure characteristic of the grasp is formulated as a set of constraints which should be solved along with a set of piecewise smooth surface equations assigned to object surface considered as potentially grasped surfaces. Their routine is useful for locating the object contact points in a fixture design. The output of their program is the number and estimated position of the contact points. With their method the contact point positions are totally dependent on the object shape and there is no geometrical constraint between them. Therefore it can not be applied to the enveloping grasp planning problem where geometric constraints relating the fingers and palm together are required.

Mirtich and Canny [5] introduced a set of criteria to find the optimum force closure 2D or 3D grasps. The property known as force closure relies on the squeezing forces and friction to restrain an object. When compared to form closure its reliance on friction is a disadvantage but it has the advantage of requiring much fewer contact points between the hand and object. They based their method on having a simplified smooth surfaced object. The finger positioning in their method is completely independent of each other and all the points on the object surface are considered potentially graspable. The result of their algorithm is a geometrical relationship between the finger contacts that could lead to an optimum force closure grasp.

T. Yoshikawa [6] analyzed the two well known grasp classifications form closure and force closure from a different point of view. He classified them as passive and active closures which both could have form or force closure as sub-categories. By formulating the reaction wrench and infinitesimal translational displacements for an arbitrary object, he derived a set of relational conditions between them to describe each of the grasp closure categories under consideration. He also analyzed the effect of friction in his classification results.

Based on the human enveloping grasping routine, Kaneto, Hino and Tsuji [7] divided the procedure of grasping into three phases: approach, lifting and grasping. Using the object dimensions and weight they introduced a feasibility space for each of these stages. The routine they proposed can be very helpful in estimating the relative placement of the hand and object before hand approach. It can also estimate the joint torque needed to complete the enveloping grasp. However it models the grasping as lifting a cylindrical shape. This shape factor can make the results quite inaccurate, but still the procedure might be helpful as a first approximation.

Smith, Lee, Goldberg, Bohringer and Craig [8] offered a relatively fast approach to find the contact points for a parallel-jaw gripper grasping a prismatic object. In their algorithm they optimized the grasp relative to the friction force exerted and the resultant torque. They require the location of the centre of the mass of the object to be known. The calculations are performed on the 2D object contour created by horizontally slicing the 3D object model. Based on the 2D slice their procedure checks the accessibility of the gripper jaws to the calculated contact points. Their approach can give a grasp planning result quickly but the need for the centre of mass position makes it somewhat difficult to implement on objects with complex volumes. Their method is also limited to prismatic objects grasped by a parallel-jaw gripper.

Hwang, Takano and Sasaki [9] solved the contact problem between a robot hand and an object as a contact between a B-Spline surfaced object and a finger modeled by a cylinder and an ellipsoid. They also presented an algorithm for the kinematics of grasp and manipulation of a B-spline surface modeled object and solved the finger joint displacements and contact points for a given position and orientation of an object, allowing multiple contact on the same finger. Their approach is complex since for any contact solution the entire modeled finger coordinates and assumed surface coordinates should be set in advance. Since the model is solved numerically in order to have a good result a reasonable initial condition must be given as an input. This algorithm could be quite useful in handling delicate or fragile objects, since in any step of the manipulation, an accurate estimate of the contact point is calculated and as a result the forces applied by the joints could be more precisely controlled. For normal objects, because the system has to solve recursive equations, it requires a lot of unnecessary calculations that will slow down the grasping system. Also since the surface of the object is modeled by B-Splines the sharp edges of the object, if any, would be estimated with smoother profiles.

In a work on force closure grasping, Borst, Fischer and Hirzinger [10] put their focus on finding a stable grasp by looking at the friction effect in distributing the force over the final grasp. In order to analyze the quality of the grasp they used a task wrench space which represents the wrenches expected to occur for a given task. Their method tries to determine the finger positioning using a simplified geometric model of the object. In their work they included the finger constraints imposed by the hand geometry and used the model of a four fingered dexterous hand. Their method is useful for fingertip grasps but it has no ability to deal with the other types of grasp such as enveloping grasps.

By using the animation software package MAYA, Moussa and Serban [11] developed a grasp simulator. In their program grippers are modeled as two parallel jaws which slowly move toward the object until they hit it. The contact detection is done by MAYA. A simplified force analysis that only tests for static equilibrium of the squeezing forces and object weight is also performed. The program does not include any quality metrics to evaluate the condition of the grasp. The approach angle of the jaws towards the object is also chosen arbitrarily.

Guan and Zhang [12] presented an algorithm for evaluating the feasibility of an arbitrary grasp and for systematically determining all the canonical grasps for a given dexterous hand and polygonal object. The algorithm formulates the kinematic constraints and force constraints as a set of equalities and inequalities. The whole system is then solved as a constrained global optimization problem. Their method is suitable for generating a variety of grasps for a particular 2D object and hand. The grasps would then need to be analyzed by another program to evaluate their relative quality.

Miller, Knoop, Christensen and Allen [13] designed a grasp planning simulator. In their program the real object is first simplified into a union of shape primitives such as spheres, cylinders, etc. The program also has a database of hand approach positions and orientations related to each of the shape primitives. It uses this database as the search domain for the simplified object. Their simulation considers the robot hand as part of a work cell. There can be obstacles around the hand and object and the program applies obstacle avoidance to the motion trajectory of the hand. Their program can work with a wide range of object shapes. The problem is each of the approach points in the search domain has the same quality value before the program simulates the fingers grasping the object. This blind selection from a big search domain is inefficient and tends to produce a long computation time. The program has no method for grading the approach points and finding the better approach position and orientation before simulating moving towards the object.

Following up on their previous work [12], Guan and Zhang [14] proposed a grasp feasibility analysis for a polyhedral object with triangular facets. However, this time they ignored the force constraints in their algorithm. They modeled the phalanges of the hand as having zero thickness and width, i.e. as single line segments. Similar to the work done by Hwang, Takano and Sasaki [9] they classified the contact points in different categories such as tip-face, link-edge, etc. They then parametrically defined them by a set of equality and inequality constraints. As they did in the 2D case [12], the problem is solved as a constrained nonlinear global optimization problem. They included four numerical examples for a rectangular prism object.

In an attempt on having both the simplicity of a gripper and the capability of grasping a wide range of objects, Balan and Bone [15] introduced a more flexible version of a parallel jaw gripper. In their design the gripper has two cylindrical fingers attached to the first jaw and one cylindrical finger attached to the second jaw. After importing the object model into the system the finger spacing is adapted to find the best gripping position over the outer surface of the object. Their grasp planning algorithm incorporated 3D geometric analysis (including 3D finger-object collision detection) and force closure analysis to generate a grasp planning answer pool. By sorting the answers found with the quality metrics defined the final answer is produced. Their method is relatively simple and can cover wide range of objects but the cylindrical fingers make the grasp quite dependent on the friction force.

As an alternative way to generate the hand approach position Lopez-Damian, Sidobre and Alami [16] used the object's inertial properties. They assumed the object has a uniform density in order to calculate the principal axes of inertia and the position of the centre of mass. They then derived the approach positions for a parallel jaw gripper from these inertial properties. They used a different quality analysis for each grasp, based on the needs and constraints each object and its environment imposed over the grasp. Their method is most useful when the object is located in a known cluttered environment.

2.2. Summary

In recent years a lot of research work has been done in the area of grasp planning. With the fast development of computers, more powerful sensors and better camera technology, researchers are trying to make the grasp planning part of the online operation of robotic systems with methods such as 3D vision based grasping. However, due to complex algorithm and noise sensitivity of most of those systems, offline grasp planning for an already known object, which is true for most industrial cases, still maintains its great value and practicality.

Researchers have been very successful in creating algorithms to solve the problem of grasping an object as a 2D shape. These methods are very useful for thin objects or objects with almost flat and unified thickness. When the object's shape contradicts these assumptions the 2D methods lose their practicality. For generic 3D objects, the major challenge for researchers is to make the 3D object known and understandable to the system as a first step. In order to pass this problem researchers have employed simplified models of the object and then developed algorithms specific to these models. Most of the 3D analysis done was based on using parallel jaw grippers or fingertip grasps. The planning of enveloping grasp for 3D objects using a dexterous hand with several degrees of freedom is an unsolved research problem.

Chapter 3: Grasp Planning Theory and Program Development

3.1. Introduction

This chapter describes the theoretical basis and development of a program for kinematic grasp planning. Forces are not considered by the program. The program inputs are the geometric model of the object; and the geometric and kinematic models of the hand. If a grasp can be found, the outputs of the program are the position and orientation of the palm and the angular joint positions of the fingers for grasping the object with an enveloping grasp. A relatively simple robotic dexterous hand design is adopted in this research. The hand consists of three fingers with two phalanges (and two revolute joints) each that are joined to a rigid palm. This design was previously shown in Fig.1.2. It is capable of performing enveloping grasps with a wide range of objects. It is also relatively inexpensive to manufacture, and is similar in design to a commercially available hand made by Barrett Technology [1] (shown in Fig.1.3).

3.2. Theory

3.2.1. Reduction of Palm Search Dimensions from Six to Three

In chapter 1 the difficulties of planning enveloping grasps for 3D objects were summarized. The greatest difficulty is the size of the search space. Finding the position and orientation of the palm is a six dimensional search. Solving for the six angles of the finger joints increases the dimensions to the search to 12. Conducting a 12 dimensional search is not realistic even with today's powerful computers. Several techniques were adopted to reduce the complexity of this search.

The first technique involves the use of knowledge of the overall shape of the object. If a uniform density is assumed, it is straightforward to calculate the principal axes of the object. In this thesis the principal axis with the smallest principal moment of inertia will be referred to as simply the "principal axis". If the object is a cylinder then the obvious choice for an enveloping grasp is to wrap the fingers of the hand around the principal axis. We extend this idea to any given object as a heuristic. Since the rotation axes of the finger joints are aligned with the palm this heuristic leads to the idea that the palm should be approximately parallel to the principal axis when the object is grasped. This reduces the dimensions of the position search space from three to one, the coordinate of the palm along the principal axis. We allow the palm to pitch and yaw relative to the object but not to roll. This reduces the orientation search space from three dimensions to two.

3.2.2. Grading System

To direct the search and improve its efficiency an object grading system is introduced. The ideas are:

- Instead of simplifying the object and then considering the simplified object as a whole, find a way to cut off the ungraspable sections of the object and input the reduced object to be analyzed. This will have the benefit of reducing the search domain over the object considerably.
- Over the reduced object, grade each elevation or level of approach for the hand according to its suitability for grasping. This grading will help the grasp planning search in choosing more probable positions of the search domain first. It will help the search process to find a feasible grasp, if any exists, in considerably less time.

To implement the grading system the 3D object is reduced to a set of vertical 2D slices[1] hinged on the object's principal axis. The program analyzes each slice and grades their elevation levels according to kinematic constraints, (i.e. finger thickness and palm dimensions), or grasp quality metrics (i.e. object shape and curvature). Then the program combines the grades from each slice together to create a 3D quality grade for the object. The idea is easy to implement and frees the method from analyzing the 3D object at once. It simply cuts the object in

[1] The object is assumed to be free of holes or voids such that the slicing will produce a single, closed, contour. If the object model contains holes or voids they should be filled prior to importing the model into the grasp planning program.

many slices and analyzes each slice, extracts the data out of each slice and accumulates it. This method can easily recognize the kinematically ungraspable areas of the object (e.g. too thin or too thick to grasp or with a curvature pattern that is unreliable for the finger positioning (e.g. too sharp etc.)) and delete them from the search domain.

3.2.3. Finger Modeling and Positioning

The target of the method is to find a feasible enveloping grasp that is within a reasonable error tolerance. There is no need to find the exact 3D contact point between the object and the finger since enveloping grasps are robust to errors.

In this thesis a new finger model is proposed. In this model each 3D finger is modeled with a set of three articulated line segments (ALS), representing the top, centre and bottom of the phalanges. To find the finger positions for the grasp, for each hand approach position each ALS will be placed onto the object contour created by slicing the object with their plane of the movement[1]. By applying the finger kinematic constraints to the results found for each set of three ALS modeling the finger, it is very easy to evaluate the feasibility of finger contact. The proposed method has two other significant benefits:

---

[1] The plane of movement for each ALS is the plane created by rotation of its line segments around their joints. Note that the pitch angle of this plane is determined by the palm positioning, and this plan is always perpendicular to palm contact surface for the given hand design.

1. Each finger is positioned separately. If it exceeds a kinematic constraint the program will reject the grasp and will not proceed to calculate the parameters of the other fingers.
2. It is much easier to compute the contact between line segments and object slices than the contacts between the phalange and palm surfaces and object surface.

The following sections will describe the details of the enveloping grasp planning program developed based on this theory.

3.3. Data and Type

The basic data blocks in the program are 3D coordinates. The 3D coordinates will be used to define higher level data including:

- Line segments, which will be defined by the coordinates of their end points.
- Directional vectors. These vectors can be outward pointing vectors used to define the outside of the object, line vectors (a unit vector indicating the direction from one endpoint to the other), the hand approach unit vector and so on.
- Extracting 2D coordinates by finding the projection of 3D points onto a specific plane.

3.3.1. Object Model

The program requires a discretized 3D model of the object. Converting the object's CAD model to the standard STL format is one way to accomplish this. This conversion creates a shell out of the outer surface of the object by putting together a series of triangles. The STL file includes a set of coordinates for the vertices of these triangles and outward unit normal vectors for each of these triangles. Fig.3.1 shows a CAD model of a sphere on the left and its STL formatted version on the right.

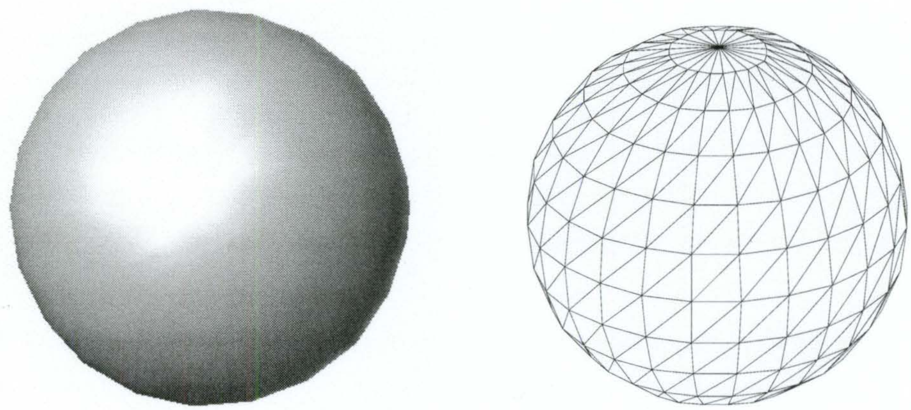

Fig.3.1. CAD Model of a sphere (left) and its STL format (right).

Almost all of the calculations performed by the program will apply to either horizontal or vertical slices of the object in STL Format, done at specific Horizontal or Vertical planes. The horizontal planes are set at different elevations and vertical planes are set at different angles. The resultant slice is the set of lines which are produced by intersecting of triangles of the STL file with the cutting plane, and the set of projected outward normal vectors corresponding to each line. The outward normal vectors are the projection of the STL outward unit vectors onto the cutting plane. Fig.3.2 shows a slice of the sphere shown in Fig.3.1. To more clearly show its discretized nature on the right side a portion of this slice is magnified. Horizontal and vertical slice planes are depicted in Fig.3.3. Fig 3.4. shows a flowchart of the slicing procedure.

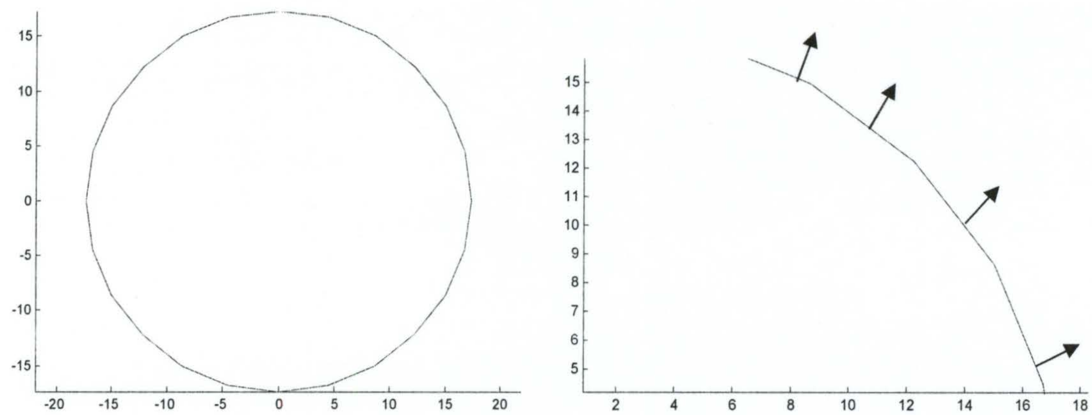

Fig.3.2. Slice of a STL formatted sphere for a cutting plane through the center of the sphere (left) and an enlarged section of it (right).The vectors shown are the projected outward normal vectors.

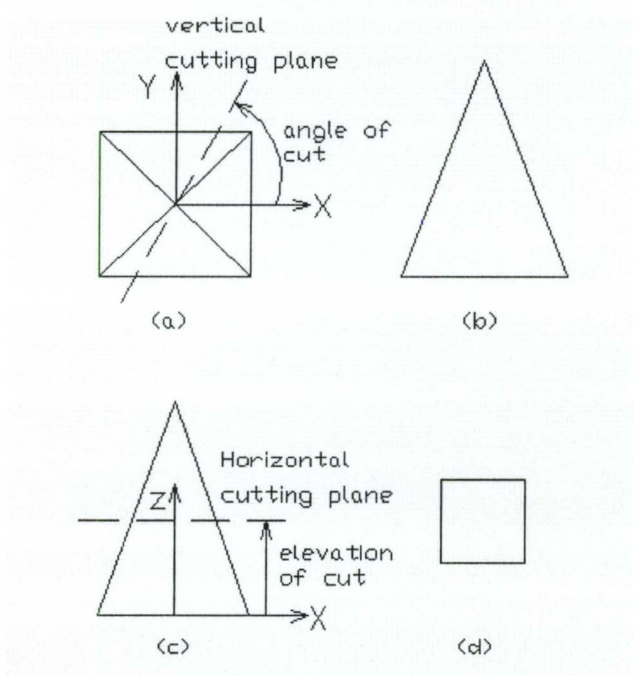

Fig. 3.3. Example of slicing a pyramid object.

(a) Geometry of a vertical cutting plane. (b) Resulting vertical slice. (c) Geometry of a horizontal cutting plane. (d) Resulting horizontal slice.

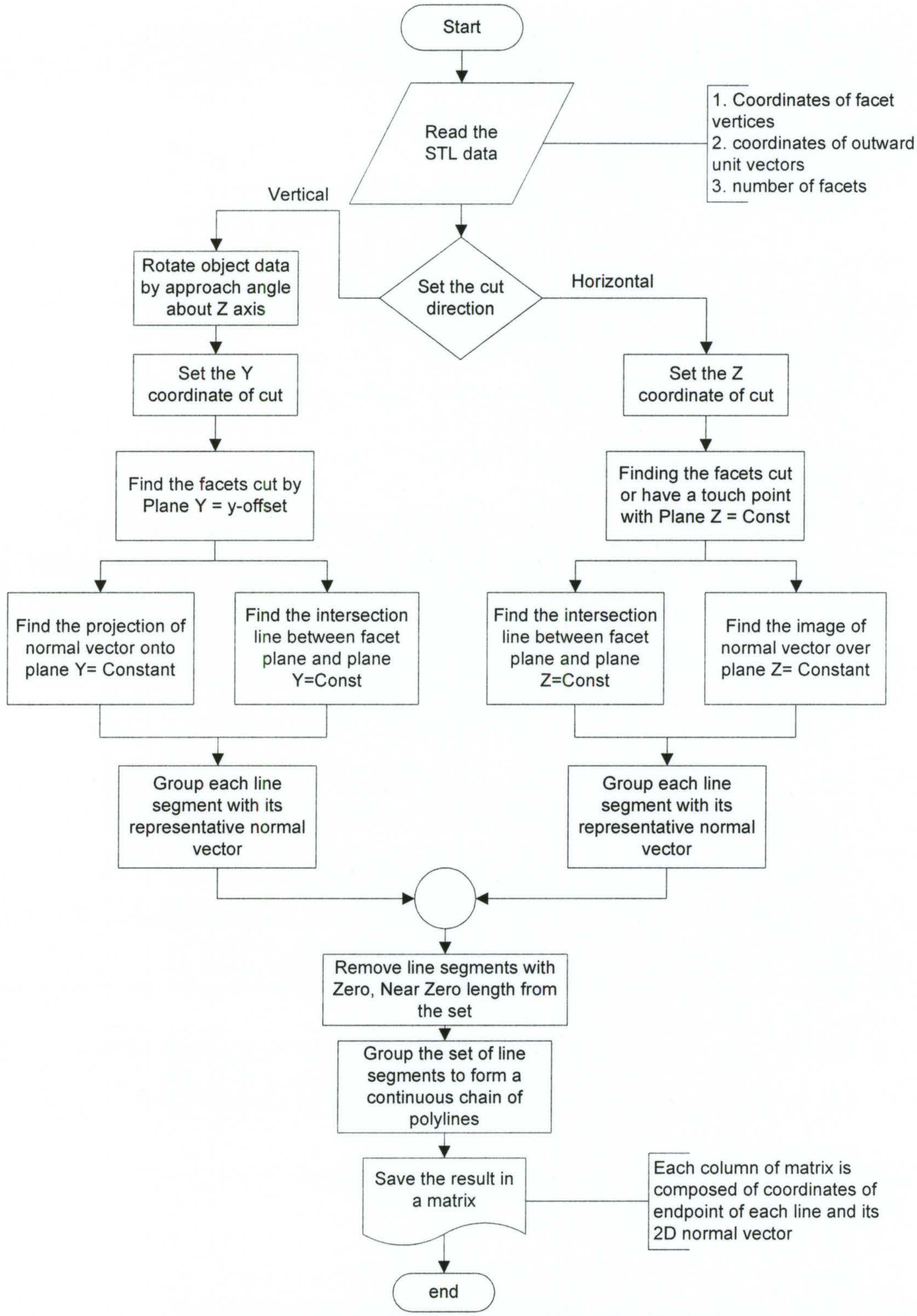

Fig 3.4. Schematic flowchart of the slicing procedure

3.3.2. Hand Model

The other inputs to the program are the geometric and kinematic models of the hand. The model used in the program is a three fingered hand, see Fig.3.5. The fingers are articulated (with two revolute joint each) and have two phalanges. The thumb finger (Finger_2), opposes the other two fingers and is centered between them.

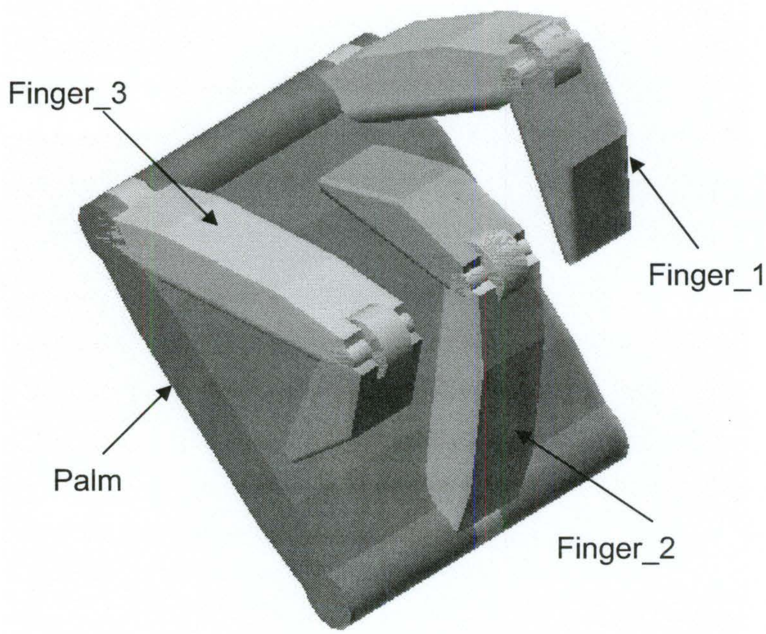

Fig.3.5. 3D Model of the hand used in the grasp planning program.

Parameters that identify the hand are:

- Palm effective *contact length*[1].

---

[1] The effective contact length of the palm or phalanges equals its length minus the unusable regions required for joint motion.

- Palm offset: palm thickness from the centreline of the palm to its *contact surface*.
- Finger offset: maximum thickness from the centerline of the finger to its *effective contact surface* and side walls.
- Tip thickness: thickness of the tip of the second phalange.
- Finger gap: distance between the centrelines of two adjacent fingers.
- First phalange effective *contact length*.
- Second phalange effective *contact length.*
- Angular range of movement of first phalange. This range is expressed relative to the palm centreline.
- Angular range of movement of second phalange. This range is expressed relative to the first phalange centreline.
- Palm margins: distance from centreline of the outside fingers to the outside edge of palm.
- Unusable region lengths: in grasp planning the object should not contact the fingers or palm near to the joints. These regions should be kept clear to allow freedom of joint movement. These parameters separate the *effective contact lengths* from the joint areas.

These parameters are shown in Fig.3.6. Fig.3.7. shows some of the dimensions for the hand used in the simulations presented in the chapter 4 and 5. The other dimensions are: Finger gap=25mm, Finger offset=7mm and Palm margin=7mm.

In setting the angular range of movement for joints it is important to note that the desired type of grasp in this project is enveloping grasping. In this case the fingers should wrap around the object to grasp it, if possible for any specific object.

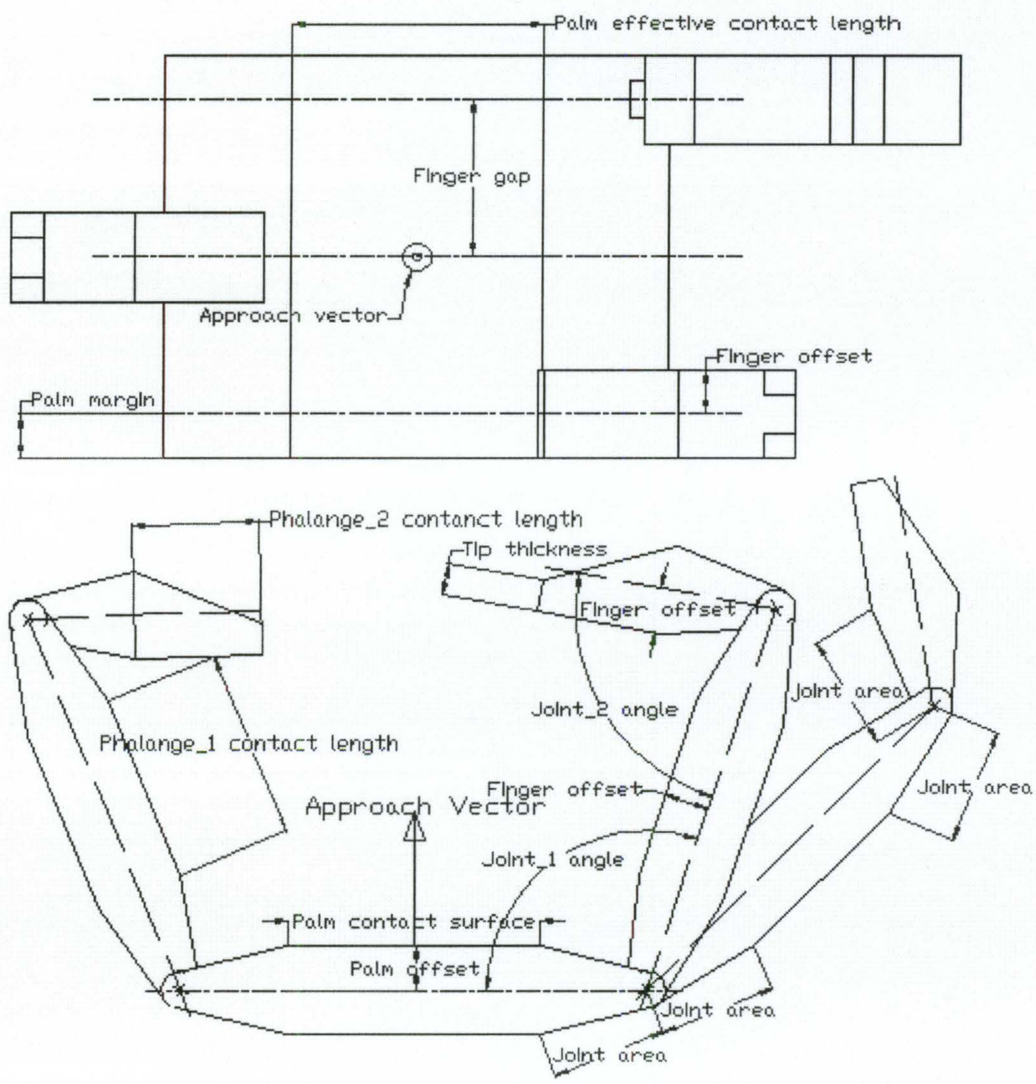

Fig. 3.6. Hand parameters

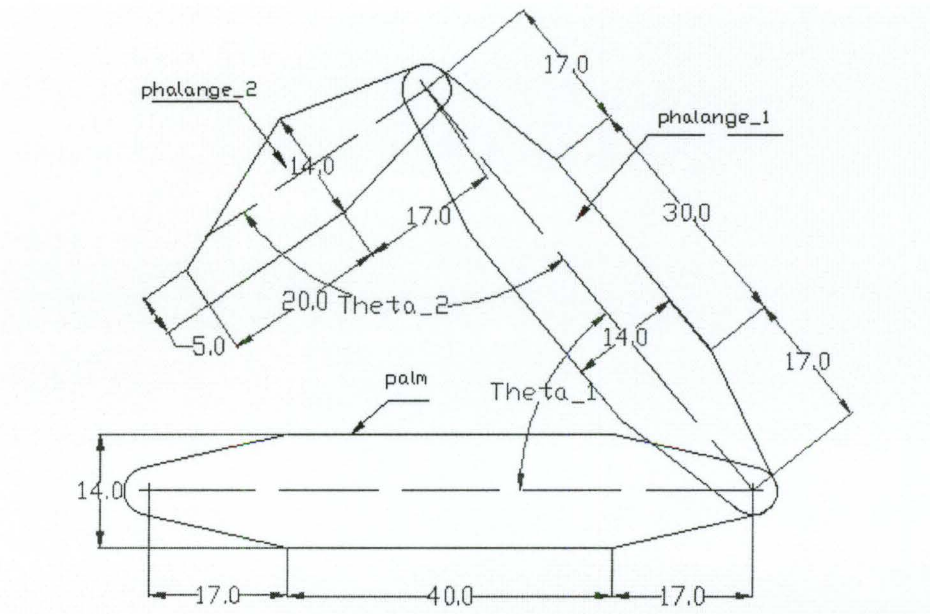

Fig.3.7. Dimensions of the hand model in mm

To imply this characteristic of grasp constraints applied to the relative angles of links with each other, $(theta\_1, theta\_2)$. The angular constraints used in this thesis are given in Table 3.1:

$45^\circ < theta\_1 < 135^\circ$	$45^\circ < theta\_2 < 135^\circ$

Table 3.1 Angular range of joint movements for the hand

To promote enveloping grasp rather than fingertip grasping the fingertips have been designed to be thinner than the rest of the finger. This idea is detailed in Fig.3.8. For enveloping grasp the contact should take place somewhere along P_1P_0. In the limiting case, since $r_1 > r_2$ contact will occur at P_1 and not at P_2 with the thinned finger design.

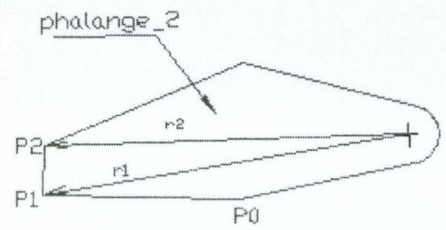

Fig.3.8. Detail of phalange 2 fingertip.

3.4. Reduction of the Search Domain

3.4.1. Why it is needed

The target of the program is to find whether the object is kinematically graspable with the given hand or not, and if yes find a high quality kinematically feasible grasp.

One way of conducting the search is to place the hand at the start of the domain and check all the possibilities for each level of elevation and if rejected continue on the next level and so on until one grasp has been found or the end of the search domain is reached. This kind of exhaustive search is completely inefficient and extremely time consuming. Another approach is to introduce heuristics to eliminate unfavorable regions from the search domain. This approach significantly improves the efficiency and will be pursued in this thesis.

Since the logic of the program is based on scanning the 3D object with directional 2D slices (horizontal and vertical) over search domain, the size of the search domain has crucial impact on amount of the calculations needed to complete the object analysis. As a matter of fact, not all the object positions have good grasp possibilities. Many positions are not capable of any kind of grasp, based on the kinematic limitations of the given hand. So before doing a detailed analysis the program will determine which part of the object is potentially graspable and rank the graspable regions. Although this reduction task adds some extra calculation time to the beginning of the program, it produces some very valuable overall benefits as follows:

1. It can give a good estimation of where the program should start putting the fingers to be reasonably close to the final answer.
2. If the initial estimation is not feasible the grading system provides a set of ranked alternatives.
3. After first evaluating the elevation levels, those which are not graspable can be excluded from the search domain, reducing its size. This can significantly decrease the execution time.

3.4.2. First Stage of Search Domain Reduction

The objective of the first stage is to reduce the size of the search domain in the Z (or elevation) direction. The top and bottom part of the object that are not well suited for grasping are eliminated first. This is accomplished by analyzing the vertical slices of the object in the elevation direction. The minimum of the highest slice elevations and the maximum of the lowest slice elevations are calculated. Regions outside these limits are not well suited for grasping and are eliminated from the search domain. Since the grasp will be more reliable when the full width of the finger makes contact with the object, and the search domain is based on the finger centerlines, the elevation limits should be further reduced by *Finger Offset*. An example is shown in Fig.3.9.

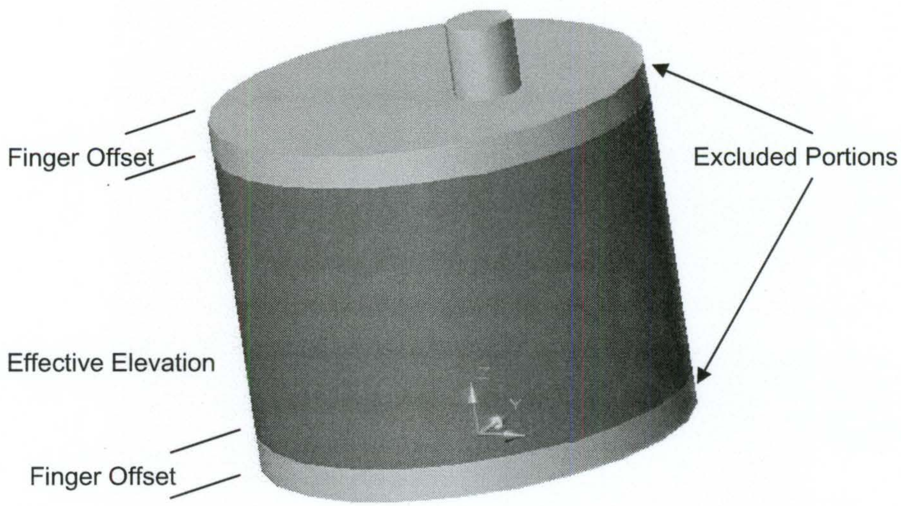

Fig.3.9. Example of finding the *Effective Elevation* of an object

The small cylinder protruding from the top will be eliminated from the search domain by the algorithm, as will the areas where the finger widths overhang the object. This leaves the range in the elevation that should be effective for grasping that we term the *Effective Elevation*. The size of the elevation direction search domain is further reduced by discretizing it. The *Effective Elevation* is discretized into intervals *Height Step* long. The parameter *Height Step* should be selected such that it divides *Finger Gap* evenly (i.e. *Finger Gap* will be an integer multiple of *Height Step*). The discretized elevations are then assigned indices, starting at 1 with the lowest elevation to produce the set of *Indexed Elevations*.

The 2D slices are created by passing the vertical plane through object and rotating it around Z axis by different angles relative to the global X axis. Fig.3.10 shows a sample object and a vertical slice of it.

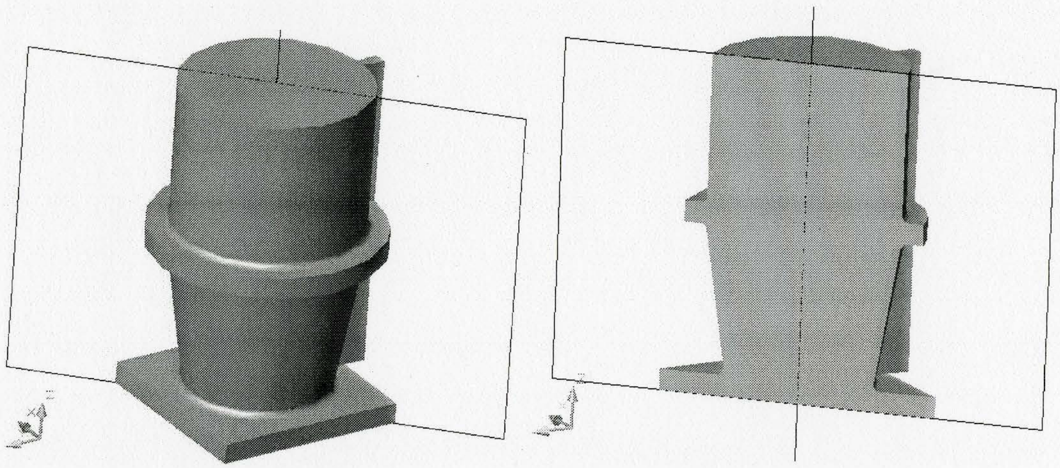

Fig.3.10. Vertical slice of a sample object

A flowchart of the complete procedure for this first reduction stage is given in Fig.3.11.

3.4.3. Second Stage of Search Domain Reduction

It is known that certain object surface shapes such as concavities are desirable for grasping. Also some objects will be too large or too small to be enveloping grasped by a particular hand design. These ideas are extended and used to grade the grasping quality of the *Indexed Elevation* in this section. At the same time some of the *Indexed Elevations* are rejected as ungraspable, further reducing the size of the search domain.

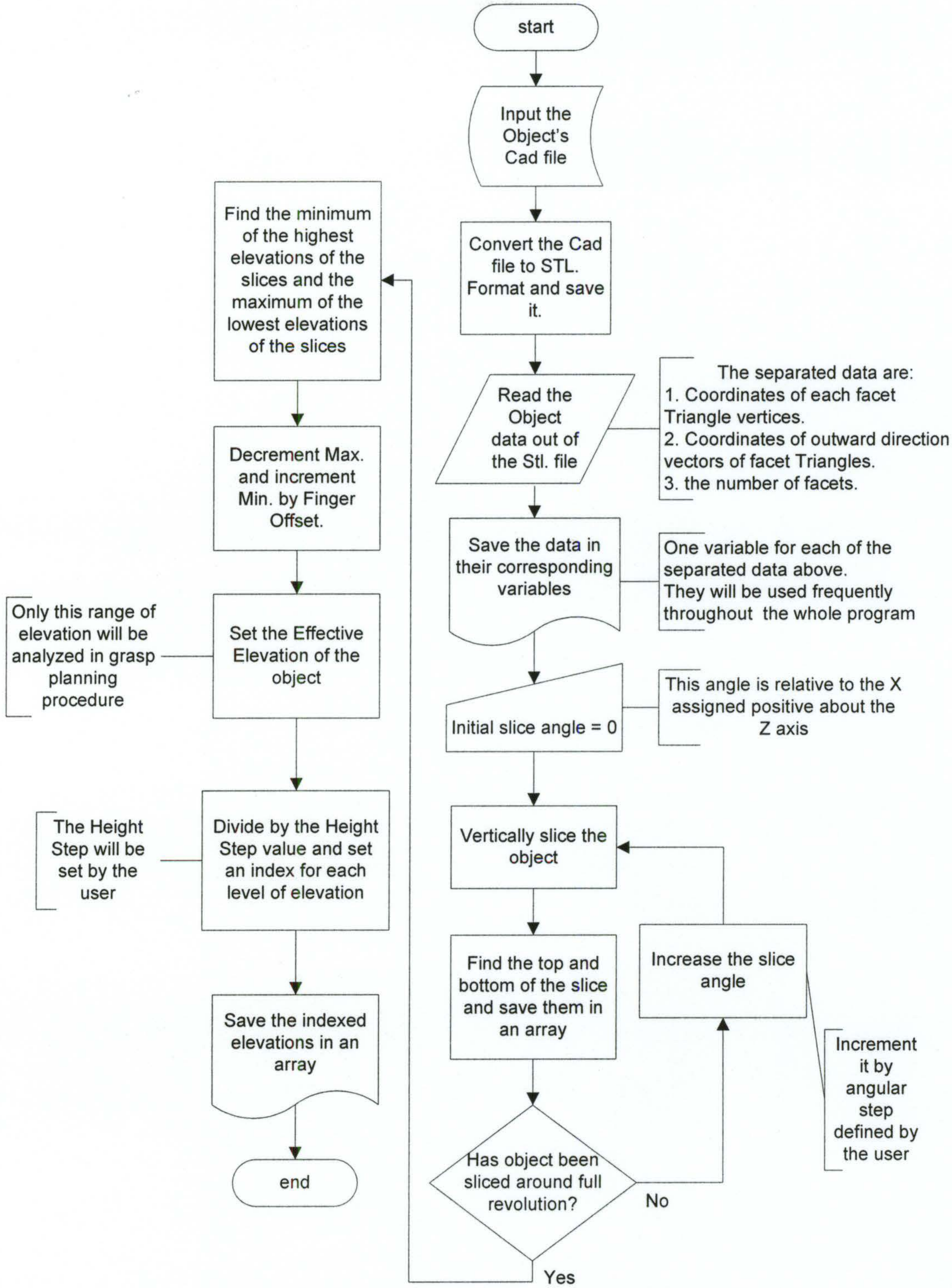

Fig.3.11. Procedure for creating the Indexed Elevation.

3.4.3.1. Analyzing the Curvature

The quality of the grasp is very dependent on the curvature of the object surface locations contacting the finger surfaces. If the fingers are placed over locations with better resting points, suitable concavities, or where the side slope directs the grasping force in a way to support the weight of the object better, the grasp should be more stable and reliable.

To grade the curvature for each elevation of the vertically sliced contour the *curvature pattern* of the left side and right side of the contour will be graded separately, and then the average of these grades will be assigned as the grade for that *indexed elevation* of that specific vertical slice. For each *vertical slice* the grade for all the indexed elevations are computed and saved. At the end, the grade for each indexed elevation is calculated the average of all the grades for that elevation.

The *curvature pattern* for each finger touch point will be defined by combining the *angular trend* before and after the finger contact point. The convention for before and after positioning for the right side and the left side of the slice is shown in Fig.3.12. It comes from the direction which the scanner pointer travels the slice contour as shown in the figure. It is very important to note that no matter if the program wants to analyze before or after the finger contact point the base point for the analysis is the finger contact point itself.

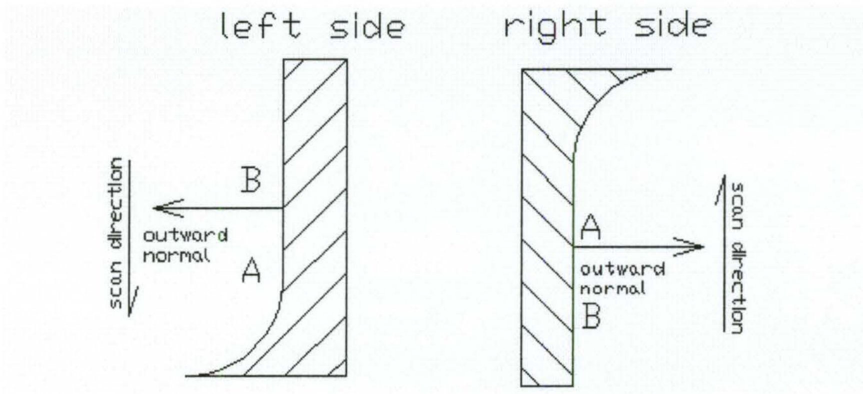

Fig.3.12: Before (B) and after (A) of the contact point

The positivity or negativity of the *angular trend* is decided by the angular change of the tangential line as the scan point moves away from the finger contact point (before or after) on the slice contour. For instance in Fig.3.13, the trend before the contact point P1 is analyzed and the scan point should move away,(in this case before), from the contact point to reach the second point P2, since $alpha2$ is less than $alpha1$ the trend is negative.

Note:

- Points P1 and P2 are produced by slicing the object STL model file.
- If the finger contact is placed onto a straight line then the *angular trend* is zero.

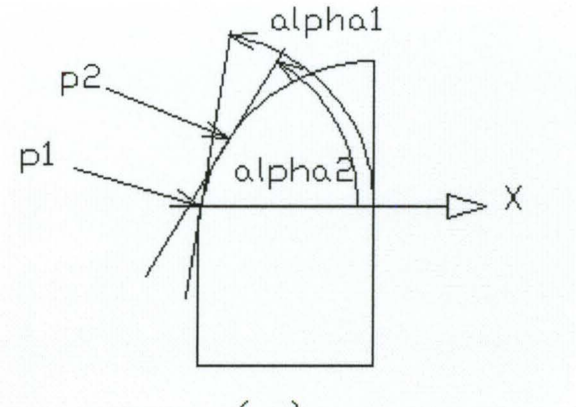

Fig.3.13, Finding the angular trend for a left side touch point

To implement the *angular trend* the program considers a certain number of consecutive line segments over the sliced contour before and after the finger touch. The angular relation between this set of selected line segments will define the *angular trend* that exists before and after the finger contact. For the simulations in chapters 4 and 5 the program will check three consecutive lines after and three consecutive lines before the contact point. This counting mechanism brings up an important issue. Since the contour is created out of the STL file, lines are created by cutting through triangular facets, so the length and density of the lines in different parts of the contour is very non-uniform. Fig.3.14 shows the point distribution over one arbitrary slice of an STL object.

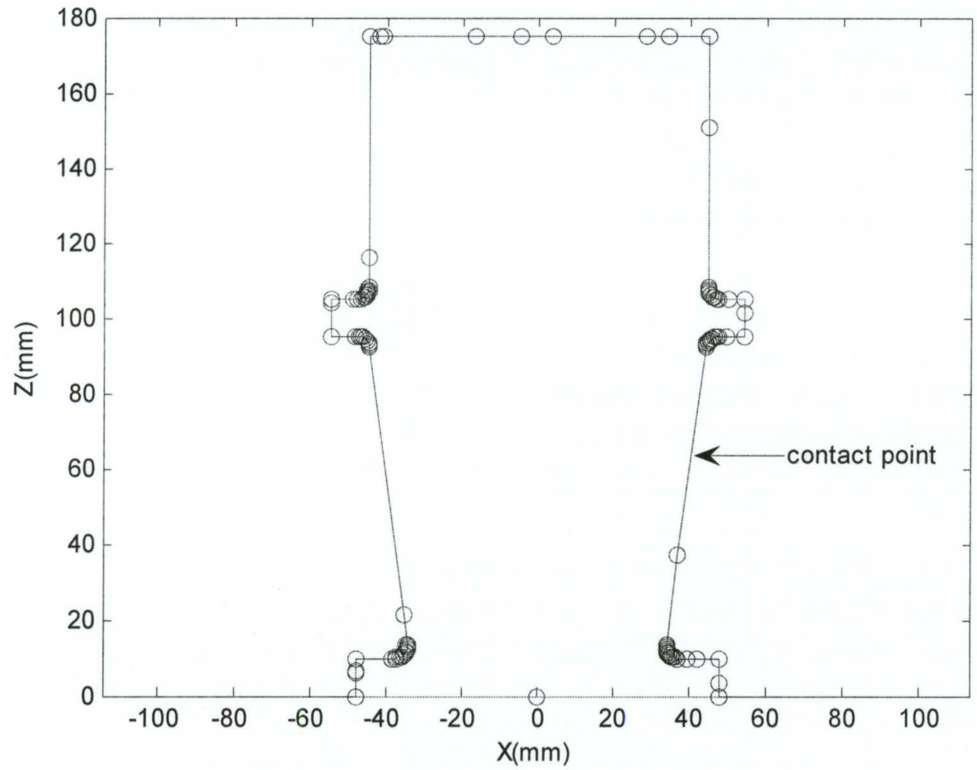

Fig. 3.14. Point distribution over the slice contour.

So if the counted number before or after the contact point is the only determining factor the result for the curvature pattern can be inconsistent. For instance, assuming the contact point is located on the right side (indicated by the arrow). The correct answer for the curvature pattern is that the contact point is on a straight line, but if the program starts to count the lines before and after the contact point and takes their *angular trend* then the answers can be inconsistent and incorrect.

To overcome this problem the program checks the length of the lines in the range of the counter, and makes its decision based on rules below:

- If the line after the contact point is *too long* then the counter will not go further.
- If the line after the contact line is *long* then the counter will decrease the count range.
- If the line in the count range is *too short* then the counter will skip it and selects the next line. The number of skips is limited. It can be changed by the user.
- Otherwise the program will continue on its regular routine.

Note that the adjectives like *long* and *too long* or *too short* are qualitative and their values must be defined by the user. They could be quantified depending on the thickness of the finger of the hand, for example.

By using the *angular trend*s a set of nine categories of curvature pattern is established. Depending on *angular trend* combination within any of these categories the finger contact will be assigned a specific grade, see Fig.3.15. The categories shown in Fig.3.15. are for finger touch on the right side of the object. The mirror images are used for the left side of the object.

When choosing the grade for the categories the direction of the object weight force vector was considered. The assumption is the weight vector acts downwards (i.e. in the negative Z direction). The finger placements that should better resist this force are assigned larger grades.

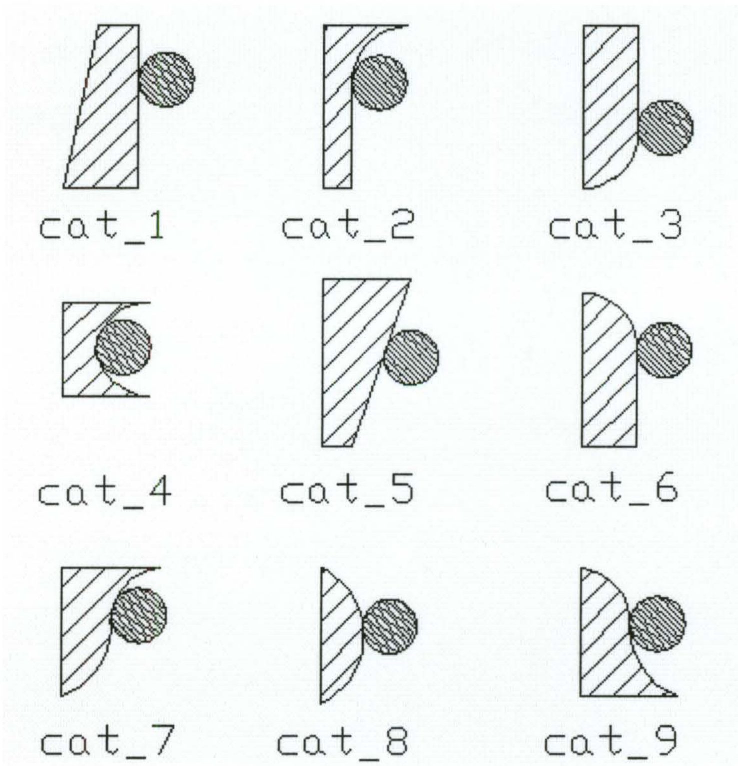

Fig.3.15, Curvature categories.

The categories ordered from the best to the least desirable are:

1. Cat_4, in this position the finger is located in a concavity. This is the best and most stable position a finger can grip the object. The finger is completely constrained and has no degree of freedom in the vertical direction.
2. Cat_2, in this case the finger is located under a ridge. Although the finger is located on a flat surface the ridge over the finger can restrain the object from slipping downwards due to gravity.
3. Cat_7, it has the benefits of Cat_2 but below the finger the object gets thinner. This demotes the grade of the grip since if for any reason the

finger applies excessive force it is possible that the finger slips under and grasp will be unstable.

4. Cat_1, the finger is just located on flat vertical surface. There is nothing about this that is special of detrimental for grasping. Since small angular deviations from vertical line practically does not alternate the grasping conditions, in this thesis the term *straight vertical* is given to a range of angular deviations from vertical. This deviation is set now in the program to be $\pm 12^{\circ}$ [1] and is called as *Vertical Deviation Range*[2].
5. Cat_5, the finger is still located on flat surface but this time it is significantly deviated from vertical. At this point the position will get demerit points, since the steeper the contour gets the more possible it is for the finger to slip. The demerit points will be deducted from the grade given to a vertical surface,(Cat_1). An important consideration is the slope should not exceed the limit when the finger force gets out of the friction cone. In this thesis the friction coefficient is assumed to be $\mu = 0.3$ so the friction cone angle would be 17°. If the slope of the line exceeds this limit then the grade would be marked as ungraspable. So the demerit grading system just works in the domain beyond the *Vertical Deviation Range*

---

[1] It can be changed by user.

[2] The range of angular deviations of lines from vertical that the program will consider as vertical lines.

(where the contour gets out of the range of being vertical) and the *Limit angle* where the line becomes ungraspable. The demerit formula is:

$$G = Cat\_5 - (abs(ang) - Threshold) \tag{3.1}$$

in which ang stands for the angle of the line measured from vertical.

6. Cat_3, is like Cat_1 but with a ridge below which can increase the risk of slippage.
7. Cat_6, the condition of Cat_6 is worse relative to Cat_3, since the object gets thinner above the finger contact. Considering the object weight force direction and the tendency of the object to slip downwards this category has a larger risk of slippage.
8. Cat_9, is like Cat_6 but the *angular trend* below the finger contact may somehow push the finger up and worsens the slippage risk.
9. Cat_8, is the worst case. The finger is located over a protrusion and any kind of disturbance may cause slippage.

In summary, Cat_4 has the highest grade and Cat_8 has the lowest grade; and the others fill the spectrum between these two categories.

There are situations when the curvature pattern found has intermediate characteristics (something between two categories) and cannot completely be allocated to one of the standard categories above. To handle these situations a bonus and demotion algorithm was developed for the program. With these algorithms the program determines the dominant category of the curvature pattern found and, depending on the minor category that the curvature pattern tends to; the curvature grade is promoted slightly or demoted slightly.

3.4.3.2. Analyzing the Effective Diameter

As mentioned earlier the type of grasp targeted by this program is an enveloping grasp. With this class of grasp the fingers should wrap around the object and the tighter the fingers wrap the better the grasp is. By having this point in mind and the fact that the thinner the object is the better the fingers can wrap around it leads to another grading parameter, the *Effective Diameter* of each *indexed elevation* of the object.

The Effective Diameter is obtained by averaging the thicknesses of all the vertical slices at a particular elevation. To find the thickness the program gets the vertical slice and then elevates the pointer to the indexed elevation under concern. At this elevation the left outermost and right outermost points crossing this elevation should be found. The distance between these points measures the thickness. Fig.3.16 shows the thickness found for two *indexed elevations*, *a* and *b* over a test slice.

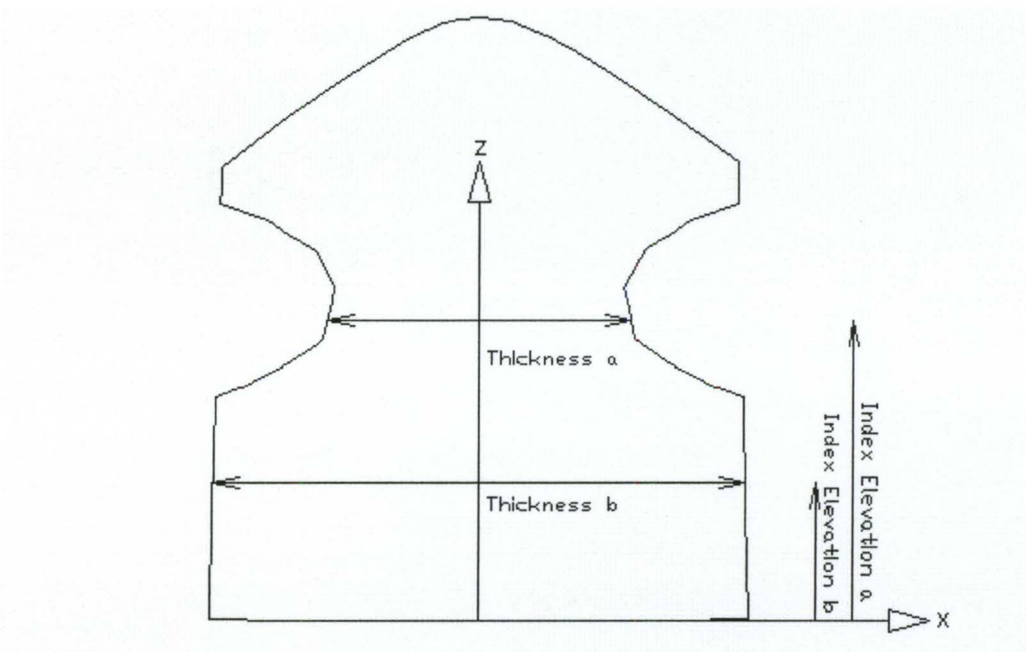

Fig.3.16. Thickness for two indexed elevations *a* and *b*.

The principal guidelines in grading by *Effective Diameter* are:

- The smaller the *Effective Diameter* of the object the greater the grade will be.
- The *Effective Diameter* should not be less than *Smallest Graspable Diameter* threshold. If it is less, then that *indexed elevation* is flagged as ungraspable.
- The *Effective Diameter* should not be greater than the *Largest Graspable Diameter* threshold. If it is greater, then that *indexed elevation* is ungraspable.

Note:

- The *Smallest Graspable Diameter* in this thesis is defined as the diameter of the smallest cylinder that can fit on the hand when the finger is at the lower limit of its joint angular range. Fig.3.17 show this parameter for the given hand model.

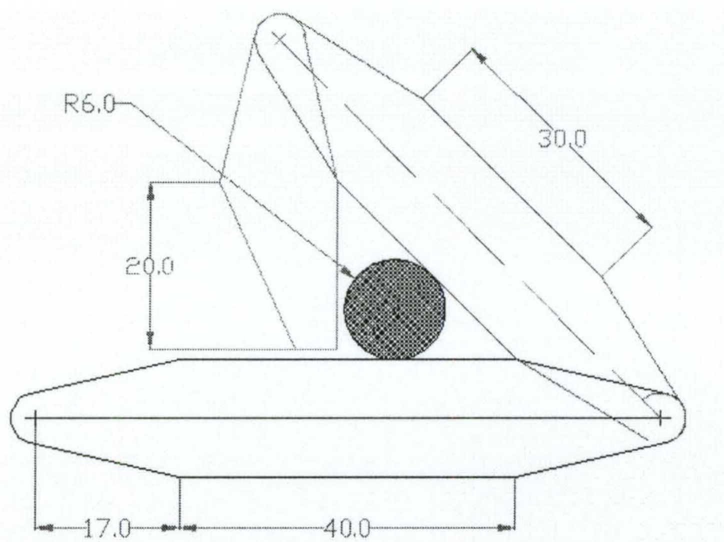

Fig.3.17. Smallest Graspable Diameter.

For this hand model, the joint limits leads to a triangular space between the phalanges. To have a mathematical estimation of it the radius formula for an inscribed circle in the triangle can be used [18]:

$$r = \frac{K}{S} \tag{3.2}$$

Where the r is the radius of the inscribed circle, K is the area of the triangle and S is the semi perimeter of the triangle.

- Since no mathematical technique was available for calculating the *Largest Graspable Diameter* in this thesis it will be the *Smallest Graspable*

Diameter multiplied by a *Magnifying Factor.* The resultant cylinder should be tested with the hand to confirm that it complies with the joint angular ranges. For simulation in chapters 4 and 5 the magnifying factor is set to be:

$$Magnify\_Factor = 9 \tag{3.3}$$

Fig.3.18 shows the test of a cylinder with the *Largest Graspable Diameter.*

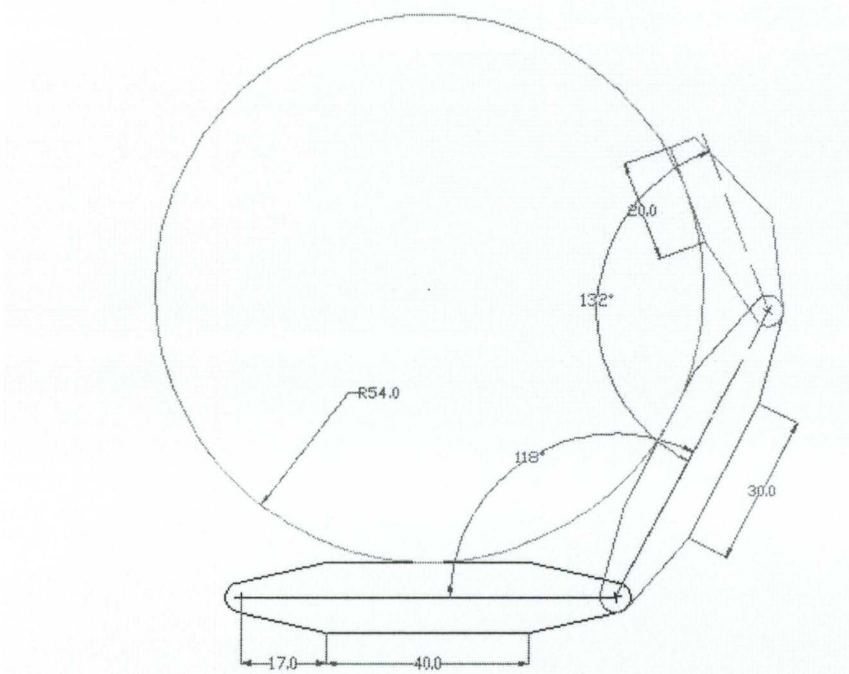

Fig.3.18 Largest Graspable Diameter.

Since for different objects the *Effective Diameter* may be different and we wish to have a relative measure rather than absolute measure, the members of the thickness array found for *indexed elevation* are normalized by their least graspable value. The combination of the grading for *Curvature Pattern* and *Effective Diameter* is summarized in Fig 3.19.

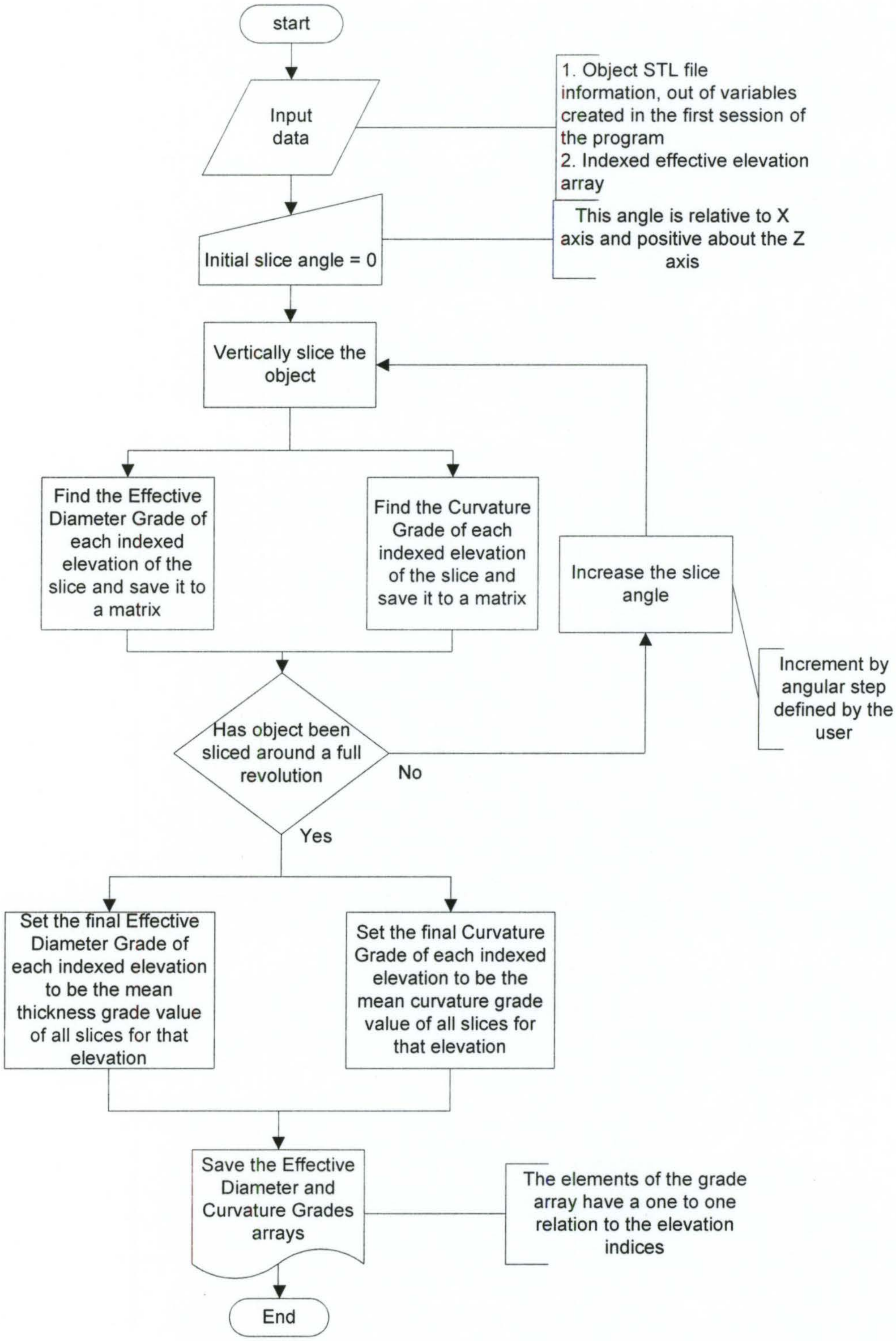

Fig 3.19. Grading Flowchart.

3.4.3.3. Significance Factors

The last step in the second stage is to combine the raw *Curvature Pattern* and *Effective Diameter* grades. A weighted average will be used where in this thesis the weighting factors are termed *Significance Factors*. These factors are set by the user. By changing them it is possible to vary the influence of the *Curvature Pattern* and *Effective Diameter* on the grasp planning.

$$Indexed\_Elevation\_Grade = Sf_c \times CurvatureGrade + Sf_t \times ThicknessGrade \tag{3.4}$$

$$Sf_t = 1 - Sf_c \tag{3.5}$$

Where:

Sf_c is the *Curvature Pattern* grading Significance Factor.

Sf_t is the *Effective Diameter* Significance Factor.

3.4.4. Third Stage of Search Domain Reduction

So far the program has only been searching in the elevation direction which could be interpreted as finding the quality of the grasp for one finger of the hand. Of course we are interested in properly placing the complete hand consisting of three fingers and a palm. Now the palm placement will be analyzed. This will be based on combining finger based elevation grades to construct a palm based quality elevation grade. Three phases will be followed at this stage:

1. First palm approach check.

2. Grade the palm position in graspable elevations.
3. Final grade sorting.

3.4.4.1. First Palm Approach Check

To be able to place the palm onto the object the *Largest Span Height*[1] should be greater than the distance between the first and the third finger centrelines. This condition is based on the conservative assumption that the palm approaches the object with its *Approach Vector*[2] horizontal. With most objects the palm will have to be rotated to make a multipoint contact with the object. Choosing the vertical palm alignment makes the program conservative, since a rotated palm requires less vertical distance. If the *Largest Span Height* does not comply with this, the object is considered ungraspable by the program and no further calculations are done.

3.4.4.2. Grading the Palm Position

If the object passes the previous stage it means that there is at least one height span that enables the palm to approach the object. The palm grading rules are:

---

[1] The largest vertical distance between two consecutive ungraspable indexed elevations produced by the second stage of domain reduction.

[2] Normal unit vector directed outwards from the centre of the palm contact surface.

- Position scanning will be done from top of the object to the bottom.
- Each palm position grade will be assigned to its *Finger_1* indexed elevation.
- The palm grade will be the average grade of its three fingers *indexed elevation grades*.
- When positioning the palm all of the *indexed elevations* between the elevations of *Finger_1* and *Finger_3* should be graspable.
- If an ungraspable elevation appears between *Finger_1* and *Finger_3* then *Finger_1* will move to the next graspable elevation below this ungraspable elevation. Additionally, all the indexed grades between the elevations of the old position of *Finger_1* and the new position of *Finger_1* will be assigned as ungraspable. This step is added to eliminate any future searching over the places where the palm can not be placed.

3.4.4.3. Final Grade Sorting

The vertical positioning of the palm is guided by the *indexed elevation* grades, where the best grade is the first choice. In some situations there are number of *indexed elevations* having the same grade, the question is how to sort them out? Since physically it is more convenient and practical for a robot hand to reach the object from the top than the bottom, in the case of similar grades the indices will be sorted from higher elevation to lower elevation. This guides the program to search for feasible grasps at the higher elevations before trying the lower elevations.

3.5. Palm Positioning

The result of the previous section is a set of proposed elevations that may lead to a feasible grasp. The *indexed elevation* grades can guide the palm elevation but can not be used to find the *approach angle*. At this point the program first elevates the palm to the suggested elevation. The program starts the approach angle from zero, and attempts to place the palm and finger onto the object to check the grasp. If the grasp is feasible the program will output the results, if not it will increase the approach angle by *Angular Step*[1] and try again. In the program the *Palm Approach vector* is always kept parallel to the global X axis, and the *palm* is moved in the negative X direction. In order to simulate the change of the approach angle the program rotates the object instead around the global Z axis. The procedure can be summarized as:

- Finding the palm domain[2].
- Placing the palm onto the object.
- Check for any kind of penetration of the palm into the object

---

[1] Integer dividend of 360° specified by user.

[2] Palm domain is the search domain that contains the contact points between the palm and the object at the current elevation.

3.5.1. Finding the Palm Domain

As previously mentioned the only data available for analysis are lines and points[1]. The approach angle of the palm is set and the palm elevation is known already, so in order to find the palm placement the set of line segments in the right side of the slice which have at least one endpoint in the *Palm Range*[2] should be found. This set is termed the *Palm Domain*.

3.5.2. Placing the Palm onto the Object

It is very important to note that in palm positioning, the slice plane and the plane vertically passing through the middle of the palm are made to be equal. It is initially assumed that the contact points of the palm and object will be on XZ plane. If this is not true the palm will penetrate the object, this will be detected by the program and the grasp attempt will be rejected. It is also assumed that object motion will not occur due to the contact of the palm and object when the grasp is executed.

The analysis is performed in the XZ plane. To put the palm into contact with the slice, the palm surface line segment is moved towards the object until it touches one of the *palm domain* line segment endpoints. After the first contact point is found it is saved and then used as rotation point like a pin point. The

---

[1] Refer to 3.3.

[2] Vertical distance between one endpoint of palm to the other endpoint of palm.

palm is rotated around it in the pitch direction until it hits the second contact point.

The conditions applied to the rotation direction are as follows:

- If the first contact point is on the top half of the palm then the palm will be rotated to find the second contact point on the bottom half (i.e. the direction of rotation is chosen to move the lower half of the palm closer to the object).
- If the first contact point is on the bottom half of the palm then the palm will be rotated to find the second contact point on the top half (i.e. the direction of rotation is chosen to move the top half of the palm closer to the object).
- If the first contact point is located at the midpoint of the palm then the program will find the second outermost point (i.e. with the second largest X coordinate) in the range of palm rotation, either on the top half or bottom half, and set it as the second contact point.

The reason for the above decisions is, since the hand is three fingered and the second finger is located between and opposite to the first and third fingers, by spreading the palm contact points on both sides (i.e. top and bottom) of the hand the resultant palm placement will be more stable.

3.5.3. Object Penetration Check in the Vertical Plane

In the previous section, when the program was trying to place the palm onto the object, the palm was just modeled by the line segment defining its contact surface in the XZ plane. After virtually placing the palm onto the object the program should check if the physical thickness of the palm allows this palm positioning or not. In the other words, it must check if the palm and object have any kind of intersection with each other. The program creates the outline of the palm, in the XZ plane using the *palm offset*, and tests for any intersection of these lines with the vertical slice contour to implement this penetration check.

3.6. Finger Positioning

The last part of the grasp planning for a specific palm elevation and approach angle involves placing the fingers around the object and checking to see if the hand can kinematically grasp the object or not. The fingers will be positioned sequentially from *Finger_1* to *Finger_3*. Note that, in order to prevent any unnecessary calculation in any of the positioning procedures if the program finds that the grasp is not feasible then it will reject the current grasp candidate and will proceed with the next available grasp candidate in the search domain.

3.6.1. Finger Modeling

For finger modeling the program should be compatible and robust enough to work with different types of fingers, and with a minimal set of hand parameters. The parameters used for the fingers are just their thickness, width, phalange lengths and angular range of motion.

This minimal list approach has the advantage of computational efficiency but also introduces a programming challenge. Different types of robot fingers can have different cross sections which are not necessarily rectangular or square or circular. The problem that arises is by having the same object, the same palm dimensions, and same finger thickness but different finger cross section. For the same hand elevation and approach angle the position of the real contact point between the finger and the object may be different. This is because the contact point position is dependent on how and where the finger surface collides with the object surface which will be different for different finger cross sections. Another difficulty is the program has already approximated the smooth curved surface of the object with flat triangular facets by employing the STL format and this approximation will create an error between the modeled and real contact points.

This leads to the question: is it really necessary to find the 100% accurate contact points to find the kinematically feasible grasp? The answer is no. If the program can find a *Grasp Parameter Set*[1] within reasonable deviation tolerances it is acceptable for two reasons. First, tightening the deviation tolerance and trying to find the real contact point of the finger would overload the program with a huge amount of extra calculations. Second, there are other software packages that can make fine corrections to the found contact points in considerably less time, for instance the contact detection library package Swift ++ [17] .

---

[1] The set of geometric parameters that can fix and define the hand position, orientation and finger angular parameters of a kinematically feasible grasp.

3.6.1.1. Proposed Finger Model

As previously mentioned the only data the program works with are lines and points. Each finger is modeled by three separate Articulated Line Segments (ALS). Three ALS are used to model the top, bottom and centreline of the contact surface of the phalanges respectively. Each ALS will be treated as a finger and will be separately placed onto the object and checked for kinematic feasibility. Fig.3.20 shows the ALS model proposed for a finger. Fig.3.21 shows the top ALS in the top view of Finger_1.

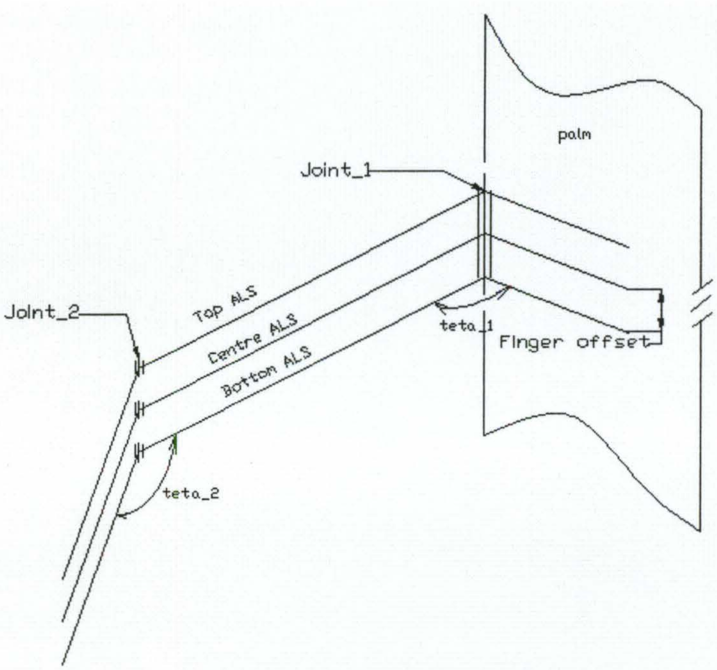

Fig.3.20, Three ALS model of a finger

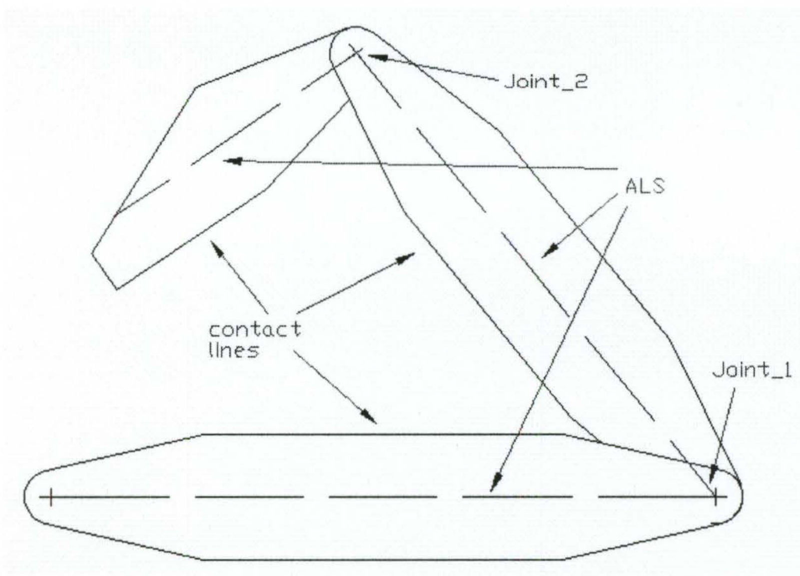

Fig.3.21, Finger_1 top ALS.

The benefits of this model are:

- The real contact must occur vertically somewhere between the top ALS and the bottom ALS of the finger. By analyzing the joint angles related to these ALS their range can be found. If any part of this range violates one of the *Angular Ranges* of the joints then the grasp candidate will be rejected. This frees the program of the difficulty of finding the real contact.
- It may happen that the indexed elevation is very close to a sharp ridge or protrusion on the object so the centre ALS of the finger placed at that elevation may indicate a feasible grasp but in reality part of the finger has penetrated into the nearby protrusion. This will force the ALS at the protrusion (either the top or bottom ALS) to have a drastic angular change

to comply with the object geometry[1]. By introducing a *Joint Angular Deviation* Parameter[2] and using it to compare the joint angles of three ALS it is very easy to detect this condition, and if it happens reject the grasp candidate. The values used in the program and for the simulations presented in the next chapters 4 and 5 are listed in Table 3.2.

$\Delta\theta_1 = 5^\circ$	$\Delta\theta_2 = 5^\circ$

Table 3.2. Joint Angular Deviation parameter for the first and second joint of each finger.

For example $\Delta\theta_1 = 5^\circ$ means that the differences between the first joint angle created by placing the top and bottom finger ALS onto the object and the first joint angle created by placing the centre finger ALS onto the object should not exceed 5°.

3.6.2. Finger Positioning Sequence

The combination of the palm and finger phalanges acts as an articulated linkage. The position and orientation of each link is dependent on the position and orientation of the preceding link. Considering this rule the sequence of finger positioning will be:

1. Locating the palm
2. Locating the phalange 1

[1] This drastic angular change may even force the joint angle outside of its Angular Range causing the grasp candidate to be immediately rejected.

[2] The allowable change in joint angles among the three ALS representing one finger.

3. Locating the phalange 2

Note that after each body is located its position and orientation are fixed by the program.

3.6.2.1. Fixing the Palm

The palm positioning was described in the proceeding sections. In this section the use of the data produced by fixing the palm is described.

By having the palm in a known fixed location the position of the first joint which connects the palm to *Phalange_1* can easily be found, since this joint is aligned with the palm. Before continuing with the next stage, the finger slice planes should be defined. The top finger slice plane is the plane that is created by rotating the top line of the finger about joint_1 (or joint_2 since their rotation axes are parallel). The *Center Finger Slice Plane* is created by rotating the finger centreline about joint_1, and finally the *Bottom Finger Slice Plane* is created by rotating the bottom line of the finger. These three planes are created for each of the three fingers of the hand (i.e. there are nine planes in total). The importance of these planes can be explained by:

- All the finger positioning should be analyzed over the object contour created by cutting the STL object model with these planes.

- The Gap Space[1] between the palm and the object should be measured on each of these planes. This distance will be used to find the palm position when the program is using the horizontal slices.

Before passing this phase, the program tests for any penetration between the palm outline and the object contour in the Slice Plane. If this happens the grasp candidate is rejected.

3.6.2.2. Fixing the First Phalange

In order to find the resting locations of the first phalange the program should simulate the rotation of this link and find its contact point with the object. This is accomplished by executing the following sequence:

1. Determining the *phalange search domain*: in order to remove unnecessary calculations, the phalange contact will only be checked over the portion of the object that is physically reachable by contact surface of the phalange. It is important to note that this search domain is a subset of the object contour created by the finger slice plane.
2. Finding the contact point: the contact point of the phalange is found by first measuring the joint_1 angles that occur when the phalange contacts the points in its search domain. The phalange contact point is the one corresponding to the widest joint_1 angle. Also note that the

[1] Space between palm contact surface and object.

contact point can only be on the phalange_1 contact line, and the *joint zone* must be free of contact.

3. Angular range check: the angle of contact found should be in the *Angular range* of the first joint, otherwise the contact is rejected and as a consequence the grasp candidate is rejected.
4. Phalange and Object Separation: the program checks for any penetration of the object and the physical body of phalange_1. If there is any then the grasp candidate is rejected.

The fixed and known location for the first phalange provides the known position for the second joint which is the pivot of rotation for the second phalange.

3.6.2.3. Fixing the Second Phalange

The second phalange positioning is similar to the positioning of the first phalange with one important difference. In finding the phalange domain for the first phalange, since it was not supposed to have any kind of tip contact the line segments of the sliced contour which had at least one endpoint in the range of the *phalange contact surface radius* were included in the *Phalange Search Domain*. Since the second phalange may have contact at the fingertip, in establishing the phalange search domain if any of the contour line is partially in the domain range it will be cut at the border. Then the cut point will be the new endpoint in the domain. Since in finding the angle of contact the program just can deal with endpoints the proper determination of the *Phalange Search Domain* is essential. Fig.3.22 shows the difference between the two routines. The search pattern wants to find the line segments and their representing endpoints in the range of R centered at C.

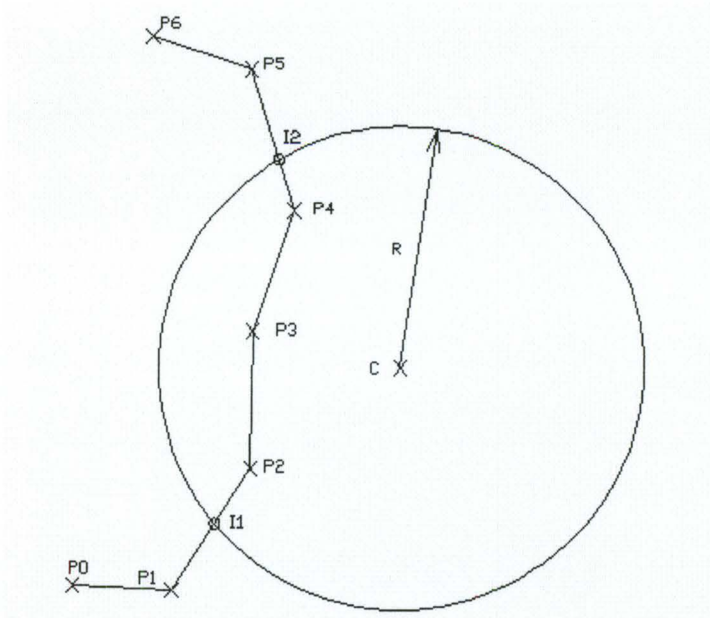

Fig.3.22. Search domain selection

The initial search domain consists of the endpoint set:

$$initial\_domain = [p_0, p_1, p_2, p_3, p_4, p_5, p_6]$$

By implementing the search pattern used for the first phalange the result of the search would be:

$$first\_pattern\_domain = [p_1, p_2, p_3, p_4, p_5]$$

Implementing the search pattern used for the second phalange the would give the result

$$second\_pattern\_domain = [I_1, p_2, p_3, p_4, I_2]$$

3.7. Summary

The whole procedure of hand positioning is done for each point in the search domain. Each point in this domain is defined by an *elevation index* and an *approach angle*. After finding the first kinematically feasible grasp the program will quit and will output the *Grasp Parameters Set* calculated. Otherwise the program will continue until it reaches the end of the search domain and will mark the object as ungraspable. The flowchart for the complete grasp planning program is given in the Fig.3.23.

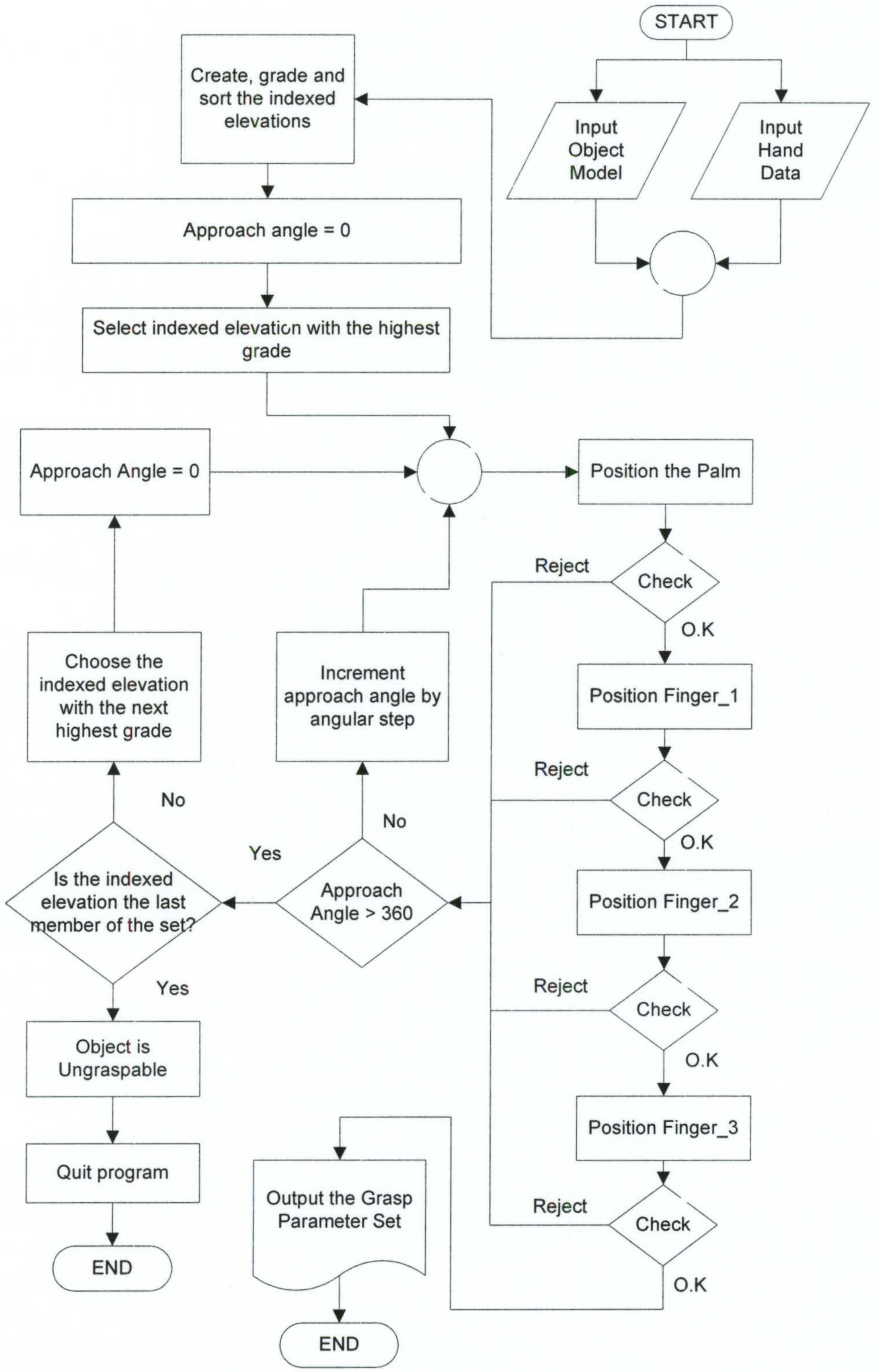

Fig. 3.23. Flowchart of the complete grasp planning program.

Chapter 4: First Numerical Example

4.1. Introduction

In this chapter the grasp planning the program is tested on a specially designed object. The program was implemented in MATLAB and consists of 57 functions and 4300 lines of code. The results the program produces as it executes are described in detail.

4.2. Design of the Test Object

The test object has been designed as shown in Fig.4.1.

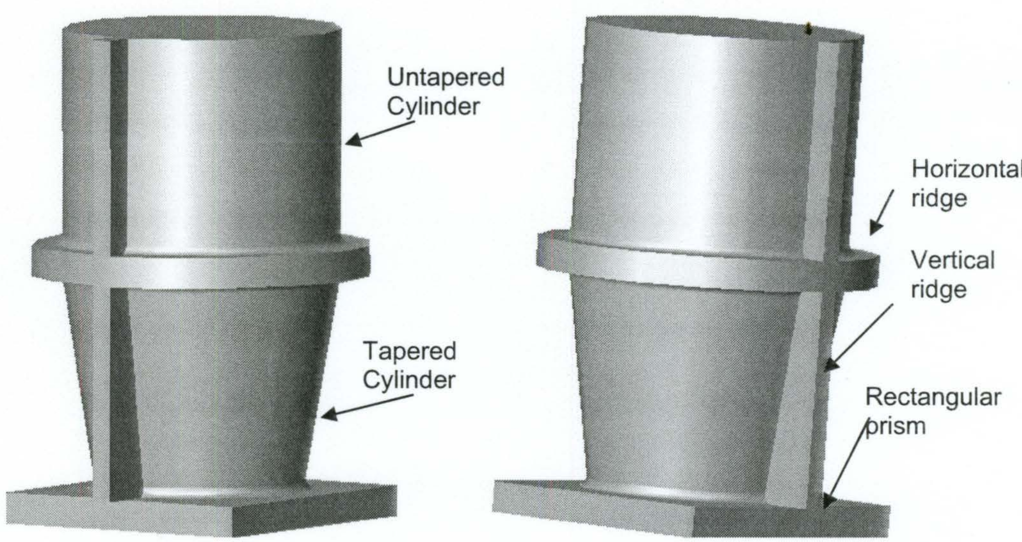

Fig.4.1. Two different views of the test object.

This object has five distinct separate elements which were designed to test different aspects of the grasp planning program. The elements are:

1. Rectangular prism base. This prism may pass through the elevation grading system, but its cross section has been designed to make the hand positioning infeasible.
2. Tapered cylinder. This element was added to show the effect of the palm rotation, palm thickness, and the effects of the ridges joined to it.
3. Horizontal ridge. This element was added over the tapered cylinder to create a potential resting place for the hand.
4. Untapered cylinder. This element was added to allow the effects of a taper to be evaluated.
5. Vertical ridge. This was added to test the angular search and the finger and palm penetration avoidance of the program.

Fig.4.2 shows the STL format of the test object CAD model.

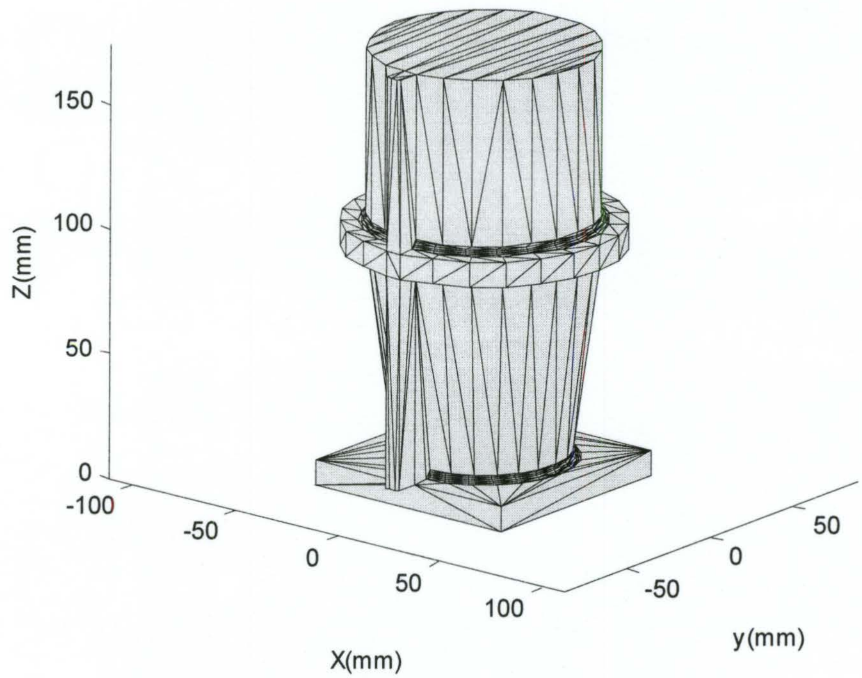

Fig.4.2 Test object STL format.

4.3. Program Outputs

4.3.1. First Stage

The objective of the first stage is to find the *Effective Elevation*. The result is overlaid on the vertical slice for the approach angle of zero in Fig.4.3. The portion of the object which is marked by the bar chart in the figure is the calculated *Effective Elevation*. Since the object has no extension which fails the object *principal axis* check there is no elimination of the elevation from the check. However as it can be seen in Fig.4.3 the *Finger Offset* caused the top and bottom ends of the object height to be trimmed off. The total object height is $175mm$, the *Effective Elevation* covers the span from $7mm$ to $168mm$. By having the $HeightStep = 2.5mm$ there will be 65 *Indexed Elevations* within the *Effective Elevation*.

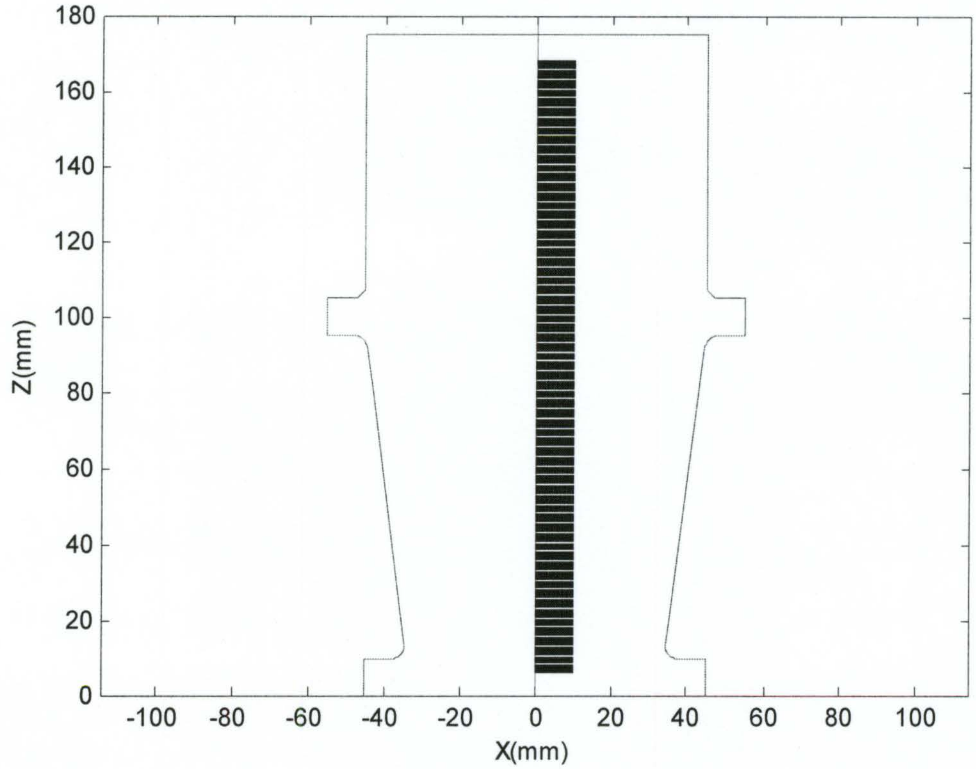

Fig.4.3. Effective Elevation.

4.3.2. Second Stage

The second stage itself is composed of three steps. The following sections will show the program results from each step.

4.3.2.1. Curvature

Fig.4.4. shows the curvature grading over the *Effective Elevation.* In the program the grading is normalized. For the sake of presenting the graph the grade magnitudes are magnified by 20. A brief discussion of the grading result (starting at the top):

- The first couple of grades near the top elevation are less than the grades following them on the untapered cylinder. This is because of the vicinity of the finger positioning to the upper limit of the object captured by *Cat_6*.
- On the remainder of the cylindrical section the grade is constant since the finger contact is on a vertical straight line in the vertical sense (i.e. *Cat_1)*.
- After the cylindrical section there is a sharp curve due to the top of the horizontal ridge. A sharp or sudden curve change is not reliably graspable so it should be excluded from the future search domain. The negative grade at this level (i.e. $Z = 107mm$) indicates that.

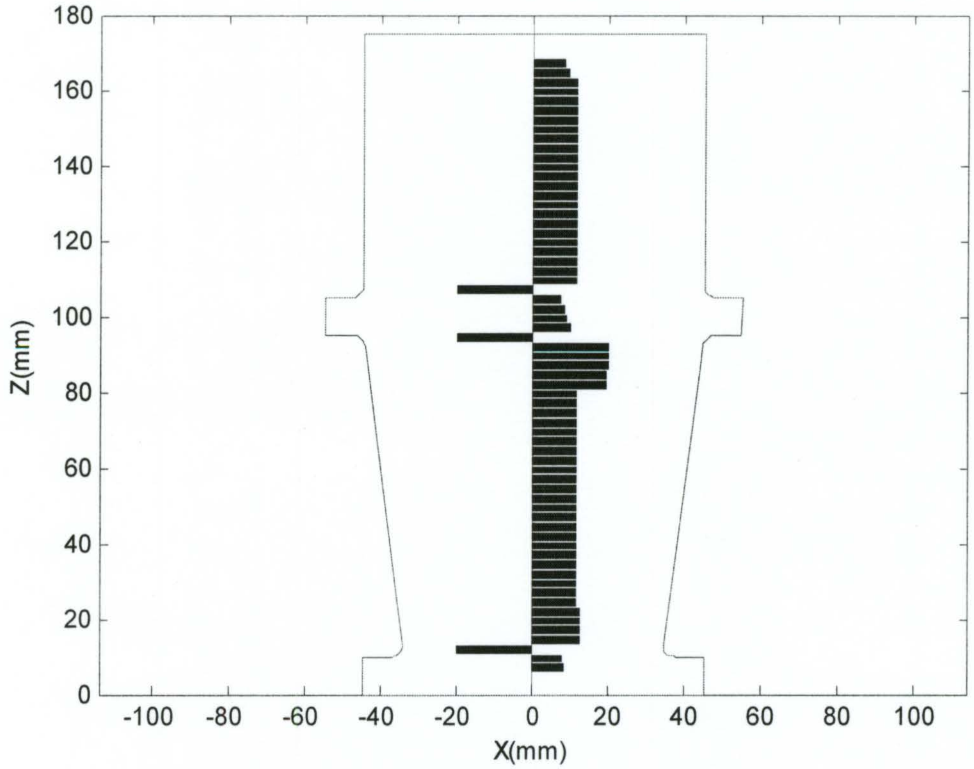

Fig.4.4. Curvature Grades (horizontally magnified by 20)

- When the search reaches the *indexed elevation* just below the top of the horizontal ridge $(Z = 104.5mm)$ the pattern of the curve sees a sharp change above the contact point, so the grade falls (using *Cat_6*). As the search moves downwards and away from the sharp edge the grade improves again.
- At the bottom of the horizontal ridge another sharp edge occurs and is graded as ungraspable as before mentioned.
- The best curvature grade occurs at the beginning of the tapered cylinder section, just below the horizontal ridge *indexed elevation* #35, $Z = 92mm$.

This is because horizontal ridge creates a resting point for a finger (using *Cat_2)*.

- As the search pointer moves lower, all the grade parameters are the same as *indexed elevation* #35 except that the pointer gets farther and farther from the resting point provided by the horizontal ridge, so its beneficial effect is reduced. The bar chart clearly shows this effect.

  The grades from level #35 down to level #31 are: $[19.9904 \quad 19.7960 \quad 19.5736 \quad 19.3417 \quad 19.1168]$.

- Sharp drop in the grade when $Z = 77mm$. At this point, the search pointer just observes a straight line and since the side lines of the cylinder are tapered less than *Vertical Deviation Range*, they are graded as a vertical straight line, the same as the untapered cylinder section.

- From the level #7, $Z = 22mm$, of the *indexed elevation* the program starts to react to the lower protrusion created by the prismatic base. Since the side lines of the cylinder are tapered inwards, the curvature pattern starts to tend to *Cat_4*.

- Another sharp curve at $Z = 12mm$ causes an ungraspable grade to be assigned.

- A similar analogy to the top part of the first cylinder applies to the last two bars of the grade chart for the cubic base.

4.3.2.2. Effective Diameter

The Fig.4.5 shows the *Effective Diameter* grades over the *Effective Elevation*.

As the figure shows, in general the smaller the average cross section the greater the grade will be.

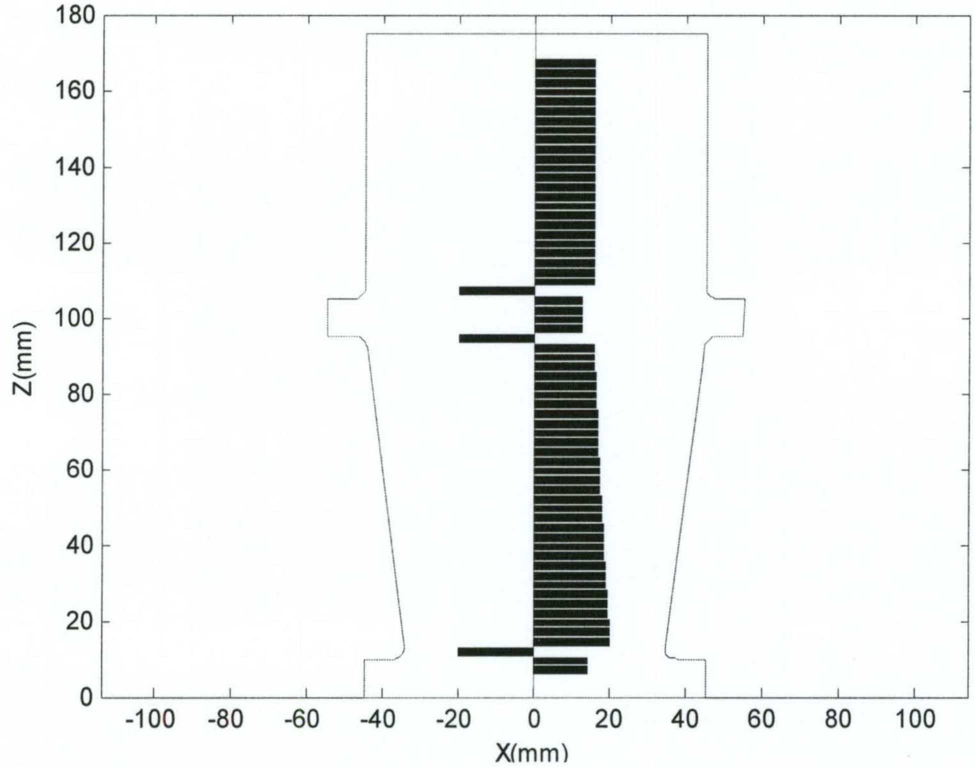

Fig.4.5. the Effective Diameter grades (horizontally magnified by 20)

As before, the negative grades indicate the ungraspable regions. An ungraspable *Effective Diameter* grade is caused when the *Effective Diameter* is outside of the *graspable diameter range*[1]. This did not happen in any part of the *Effective*

[1] Diameter range between smallest Graspable Diameter and Largest Graspable Diameter.

Elevation, but there are three elevations indicated as ungraspable in Fig.4.5. What made them ungraspable is the interaction between the *curvature grade vector* and *Effective Diameter grade vector.* In order to avoid any unnecessary calculations over the ungraspable elevations, if an ungraspable elevation is found it will be marked in both grade vectors and will be excluded from future analysis. The ungraspable elevations seen here in the *Effective Diameter* grading actually originated with the curvature grading.

4.3.2.3. Adding Significance Factors

For this simulation the significance factors related to curvature and thickness grades were chosen to be: $Sf_c = 0.6$ and as a result $\rightarrow Sf_t = 0.4$. Fig. 4.6 shows the combination of the Curvature grading, Fig.4.4, and *Effective Diameter* grading, Fig.4.5, obtained using these significance factors.

4.3.3. Third Stage

In this stage the vertical placement of the palm is considered. As explained in section 3.4.4 this involve three phases.

The Largest Span Height[1] is large enough for palm first approach so the object will pass the first stage. The objective of the second and third stages is to

[1] From Fig.4.7 this equals: $97mm - 12mm = 85mm$.

grade and sort the positioning condition of palm for each indexed elevation, respectively.

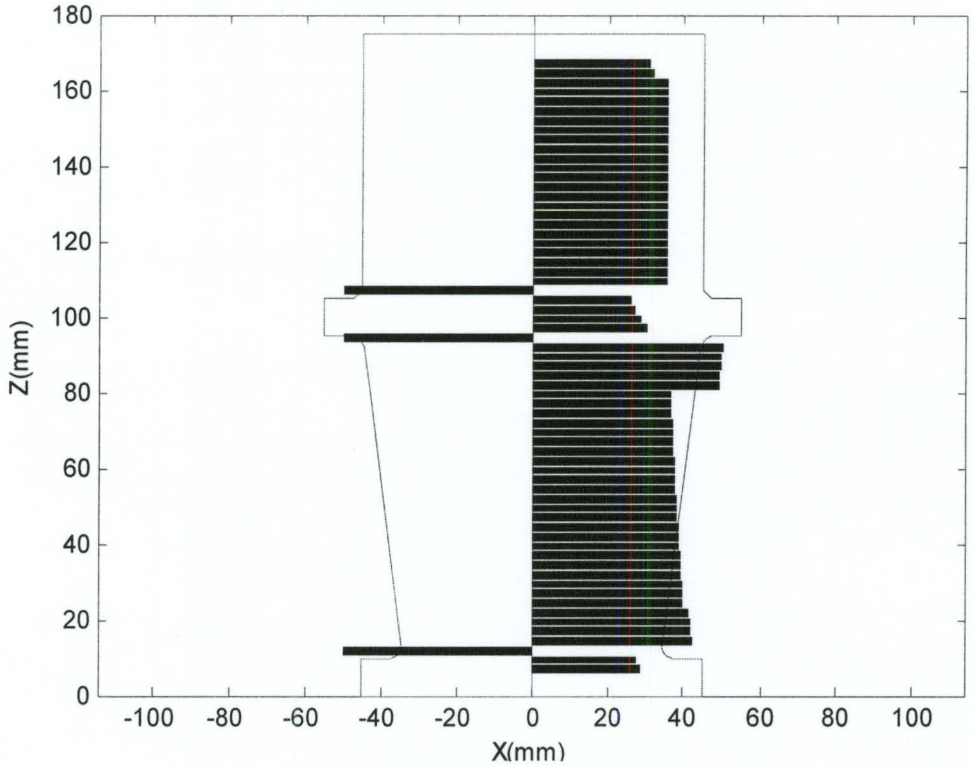

Fig. 4.6. Grades after implementing Significance factors, (Horizontal scale of 50)

As previously mentioned[1] the palm grade will be assigned to the index of its Finger_1 elevation and its magnitude will be the average of the three finger grades. If the one of the fingers reaches an ungraspable level, all the palm positions that have any overlap with this level will be marked as ungraspable.

---

[1] Refer to section 3.4.4.

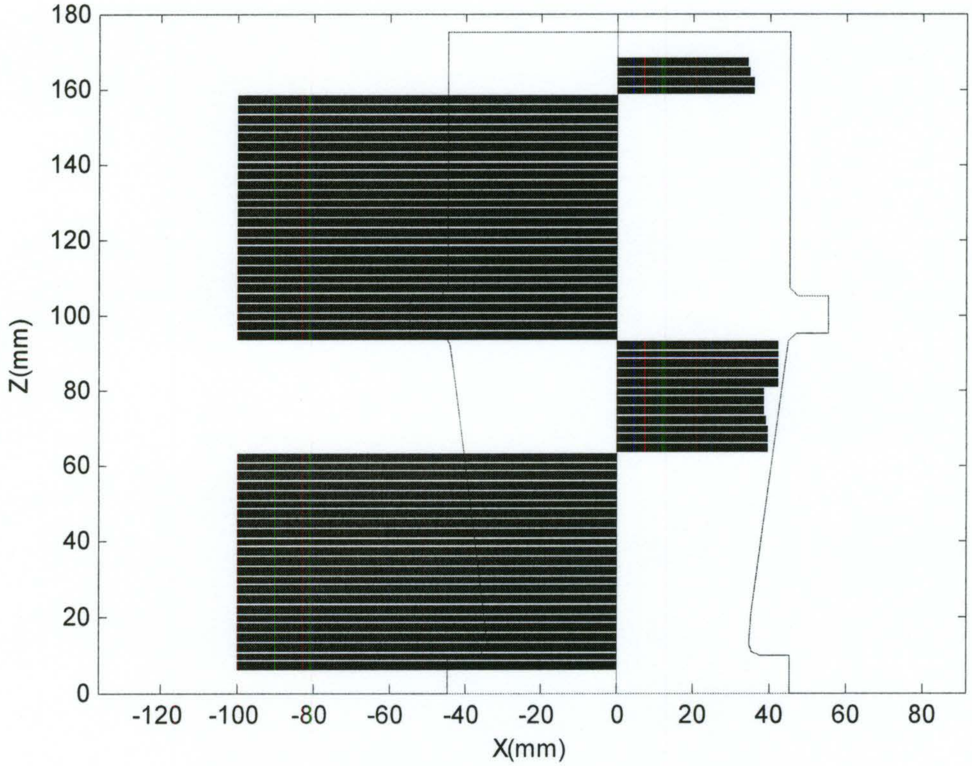

Fig.4.7. Grades after the third stage.

Fig.4.7 shows the bar chart of the final grades. There are ungraspable regions:

- The first region on the top part of object, $Z = [94.5mm, 157mm]$ is the result of overlapping the palm vertical dimension with the ungraspable indexed elevations created by the curvature grading.
- The second region $Z = [7mm, 62mm]$ starts at the top because of the overlapping of the palm vertical dimension with the ungraspable indexed elevations created by the curvature grading, and continues because the *Finger_3* elevation exits the range of *Effective Elevation*.

The final output of the third stage is the vector of sorted indices, for this test object the result is:

$$output = [31 \quad 32 \quad 33 \quad 34 \quad 35 \quad 24 \quad 25 \quad 26 \quad 27 \quad 28 \quad 29 \quad 30 \quad 63 \quad 62 \quad 64 \quad 65]$$

4.4. Palm Positioning

After executing the palm positioning procedure of section 3.5 the test object was found to be graspable. According to search domain priorities the first feasible grasp is at $ApproachAngle = 70^0$ and *indexed elevation* #26 $Z = 69.5mm$. The figures shown in this section and the following section are at this specific grasp position and orientation. Now the results corresponding to sections 3.5.1, 3.5.2 and 3.5.3 will be described.

4.4.1. Palm Domain

Fig.4.8 shows the vertical slice of the object for the approach angle mentioned above. The circles on the right side of the slice show the endpoints of the line segments that whose vertical coordinates are in the *Palm Domain*. As it was expected from line segments created by cutting an STL format file, their lengths can be quite random, with a long line segment on the top and couple of very tiny line segments on the bottom of the *Palm Domain* found.

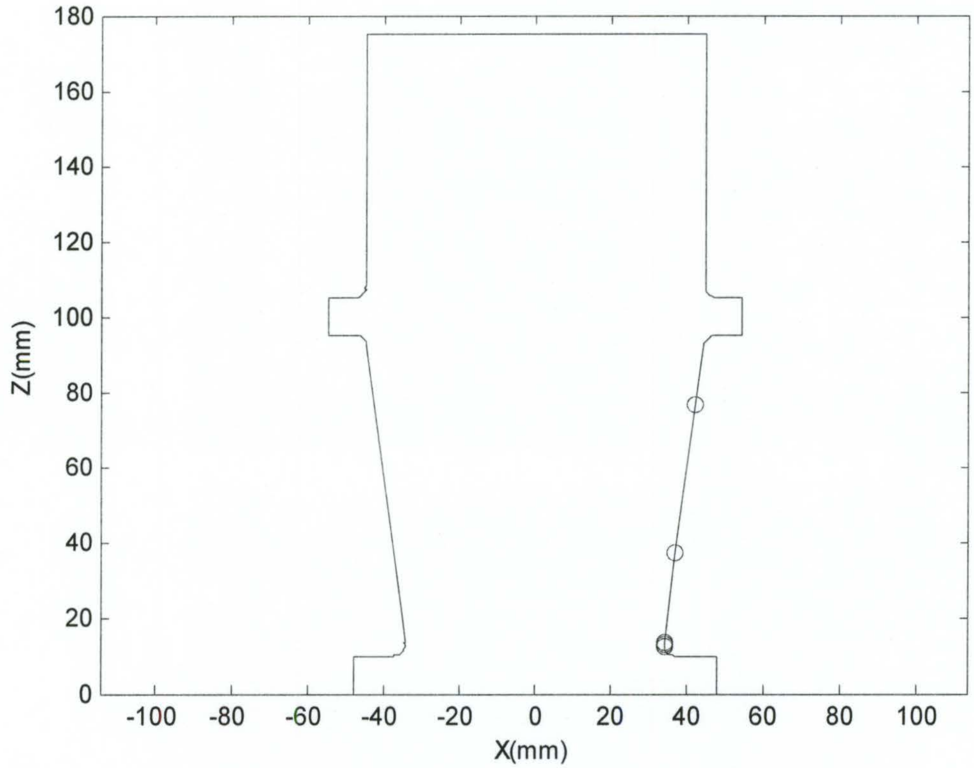

Fig.4.8. Palm Domain.

4.4.2. Placing the Palm onto the Object Side

After finding the Palm Domain the palm *contact surface* should be placed onto the side of the slice. Fig.4.9 shows the contact line segment of the palm on the object. Since the object has no concavity at these elevations the palm contact surface is actually tangent to the surface of the tapered cylinder and has an unbroken line of contact. The angle of rotation that the palm undergoes to find its vertical position on the object side is assigned to the *Grasp Parameter Set* as *Palm Pitch Angle.*

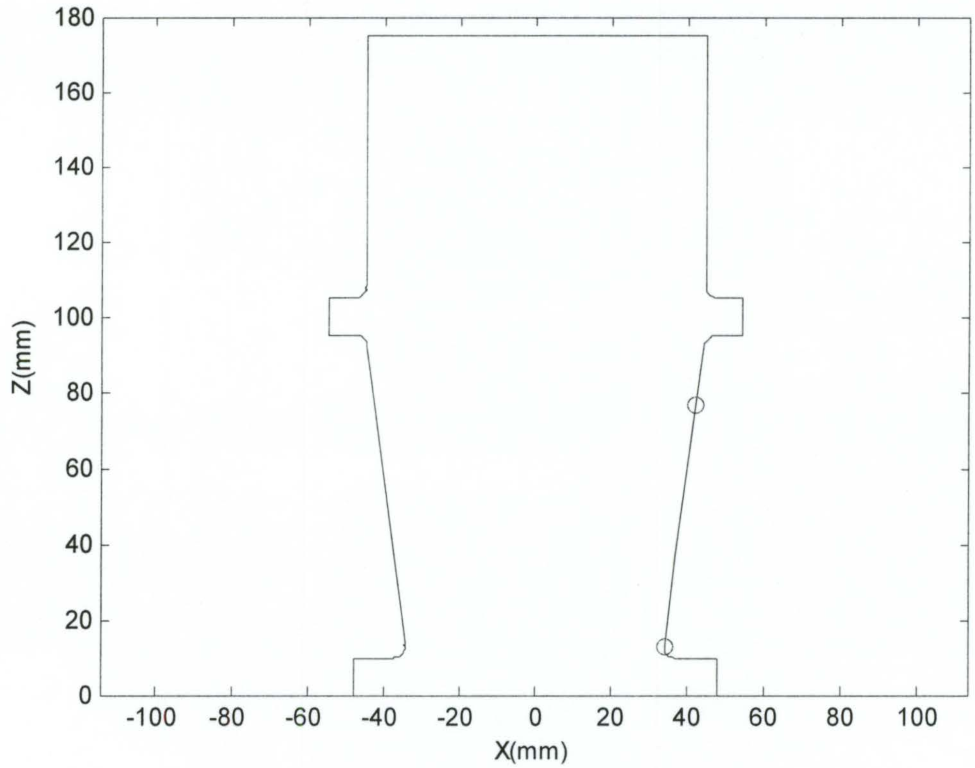

Fig.4.9. Palm contact line

4.4.3. Vertical Penetration Check

When the palm contact line is placed onto the object it is possible that the palm and object will penetrate into each other. Fig.4.10 shows the palm outline for this vertical slice. As the program found and the figure shows the palm and object do not penetrate. The vertical palm positioning is finished and program proceeds to the finger positioning.

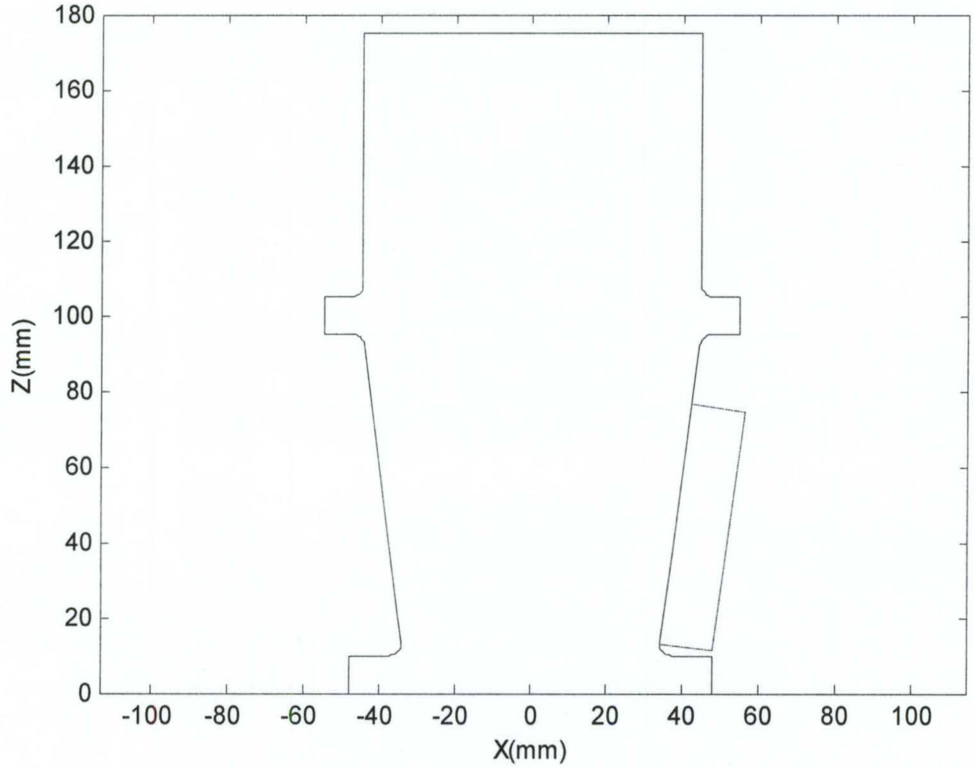

Fig.4.10. Palm outline and lack of penetration with the object

4.5. Finger Positioning

4.5.1. Fingers Slice Planes

As described in section 3.6 each finger is modeled with three separate ALS, and each of them are tested with the object separately. Each ALS acts in its *finger slice plane*. Since all the positioning parameters and joint placements will be found on these planes the first step is to compute them.

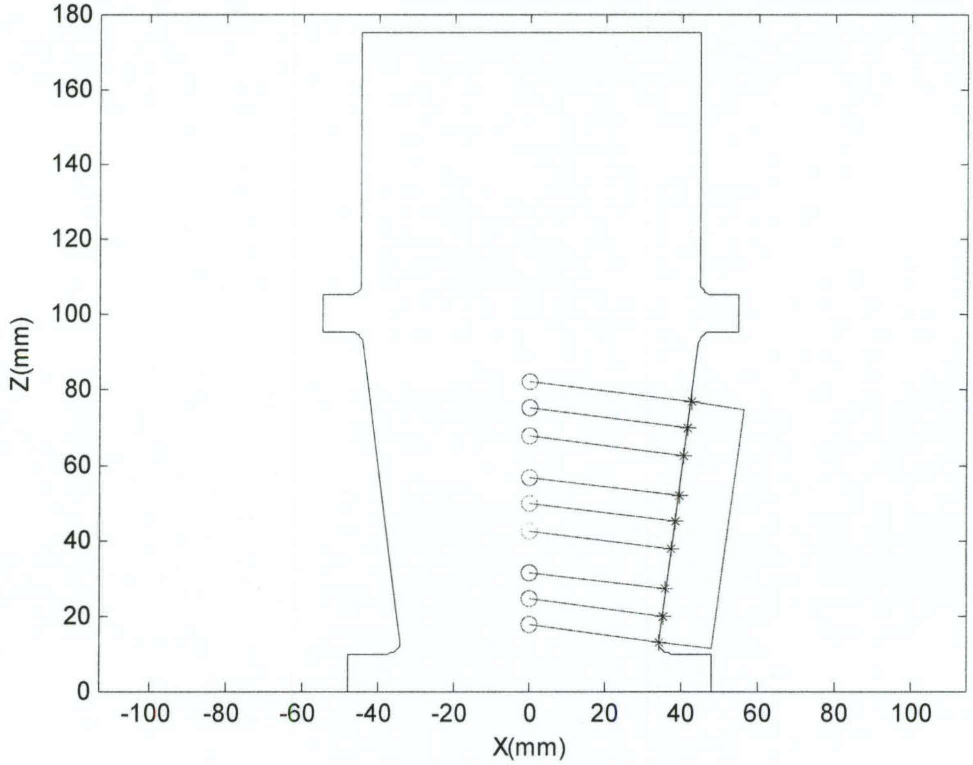

Fig.4.11. Fingers slice planes.

Fig. 4.11 shows the direction and elevation of the slice planes computed for each finger. The circles show the intersection of these planes with the object *principal axis.* With the *finger slice planes* known, the *Gap Space* parameters for each finger level were calculated. As can be seen from the Fig.4.11 all of them equal zero for this particular palm location.

4.5.2. Fixing the Palm

Out of the nine ALS for the hand, the centre ALS of *Finger_1* is chosen to depict the steps of the finger positioning procedure. All the coordinates and slice contours are shown on the corresponding *slice plane*. The centreline of the palm is always kept in the XZ plane. In the top view the palm *contact surface* should always be perpendicular to the X axis, since if it rotates then the vertical palm *contact line* established in the previous section would become invalid. The palm is moved towards the object until the distance between its *contact surface* and the object reaches the *Gap Space* found previously. Fig.4.12 shows the position of the first joint, shown by a circle. The asterisk marks show the centre and opposite end of the palm centreline. After the palm location has been fixed the program checks for interference of the palm and object in the slice plane. For the case shown the palm and object have no penetration into each other and passed the test.

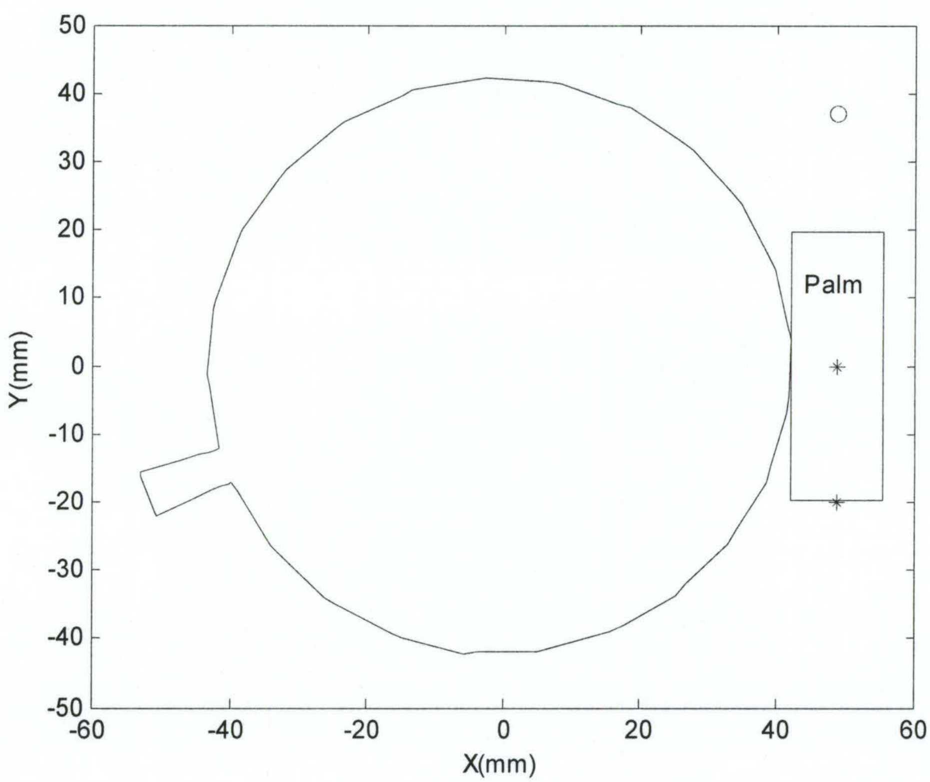

Fig.4.12. Fixed location for palm and first joint.

4.5.3. Fixing the First Phalange

The first joint connects the palm to the first phalange. Now the program should rotate the first phalange to find the contact of its contact surface with the object.

4.5.3.1. First Phalange Search Domain

To find the contact point between the first phalange *contact surface* and the object contour, the program must first find the search domain (all the line endpoints in the disk that the phalange *contact surface* sweeps over the slice plane). The Diamond shaped points in Fig.4.13 are the endpoints that belong to the *phalange domain.*

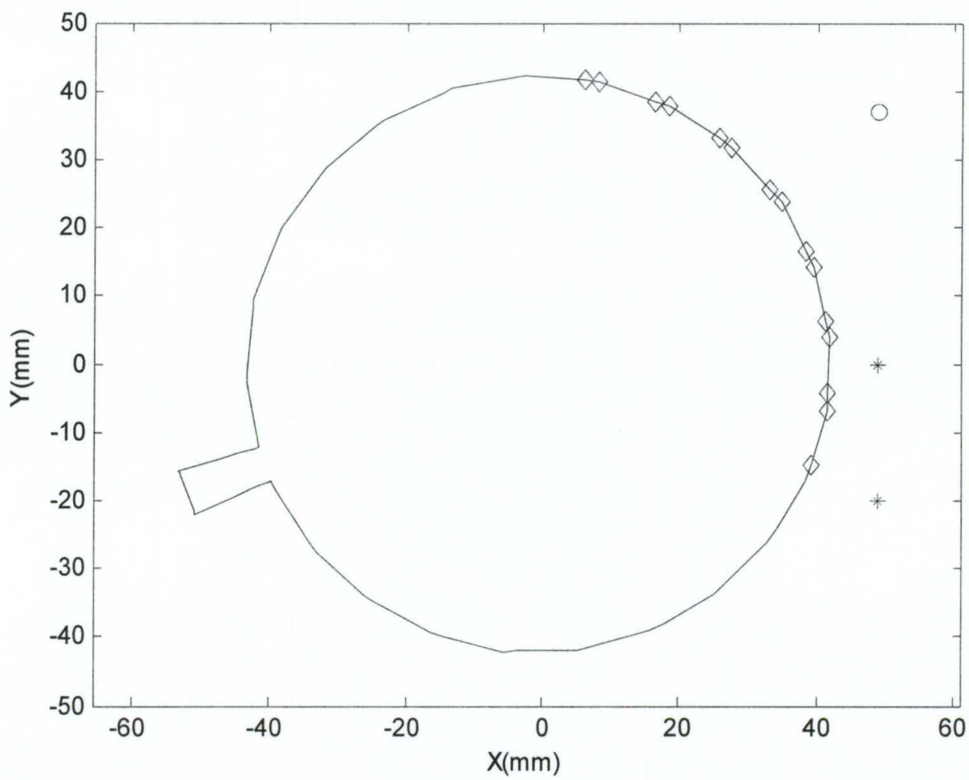

Fig.4.13. Contact search domain for the first phalange

4.5.3.2. Finding the Contact Point

The *angle of contact* for each of the points found in *phalange domain* was tested and the point corresponding to the largest angle in the group was taken as the contact point for the first phalange[1].By finding the contact point the program established the position and orientation of the phalange *contact line*. Fig.4.14 shows the first phalange positioning on the object.

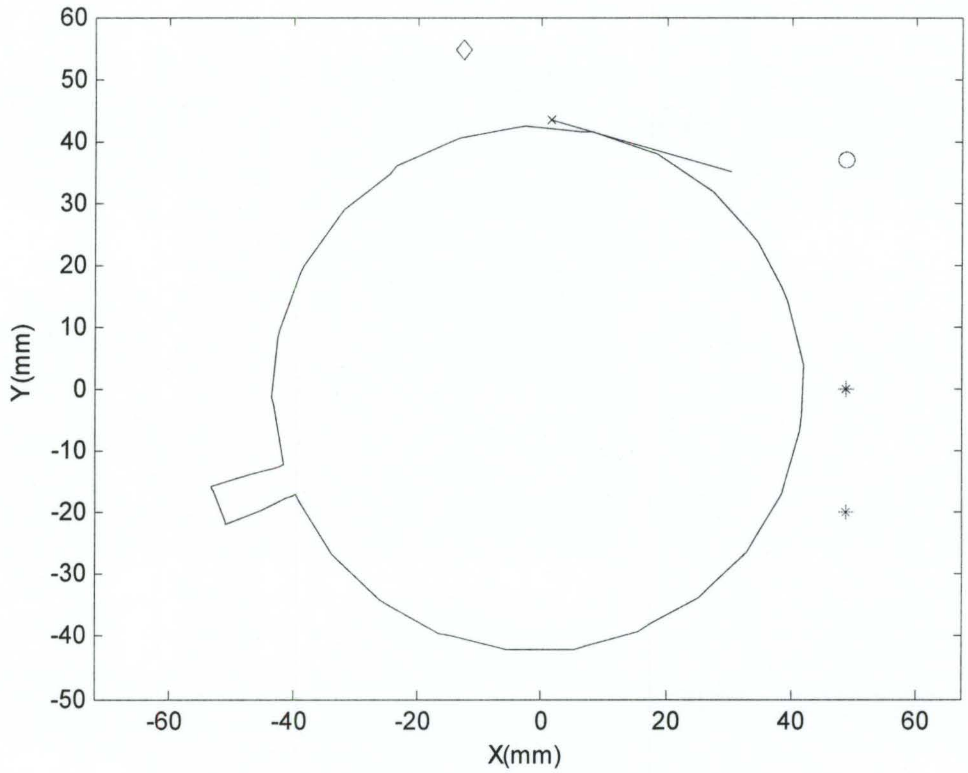

Fig.4.14. Fixed location for the first phalange fixed.

As demonstrated by the figure, the *contact line* is correctly placed over the object and no penetration happened, the × mark shows the end of the *contact line*. The

[1] As long as the angle is within the angular range for the first joint, and there is no penetration of the contact line into the object.

angle of contact, $\theta_1 = 106.2^\circ$ is in the *angular range* of the first joint and the phalange placement was completed successfully. By fixing the first phalange the position of the second joint became fixed and could be calculated. It is shown by the diamond-shaped mark in Fig.4.14.

4.5.4. Fixing the Second Phalange

Following the procedure of section 3.6.2.3, the search domain for the second phalange is given in the Fig.4.15 and the fixed location is shown in Fig.4.16.

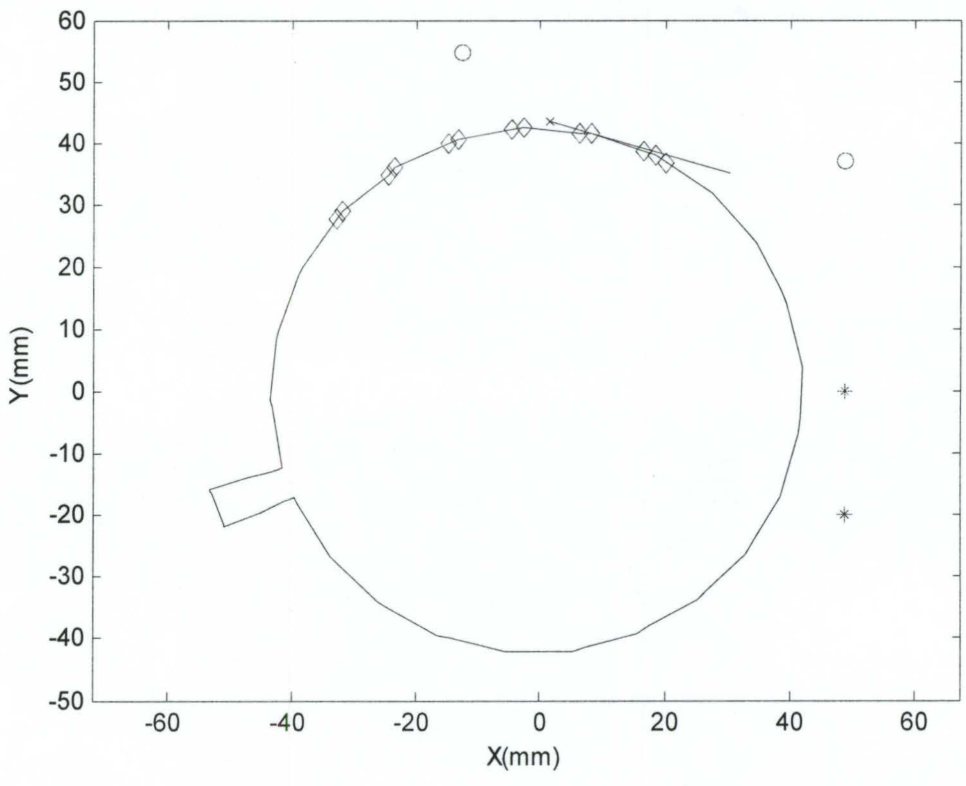

Fig. 4.15. Contact search domain for the second phalange.

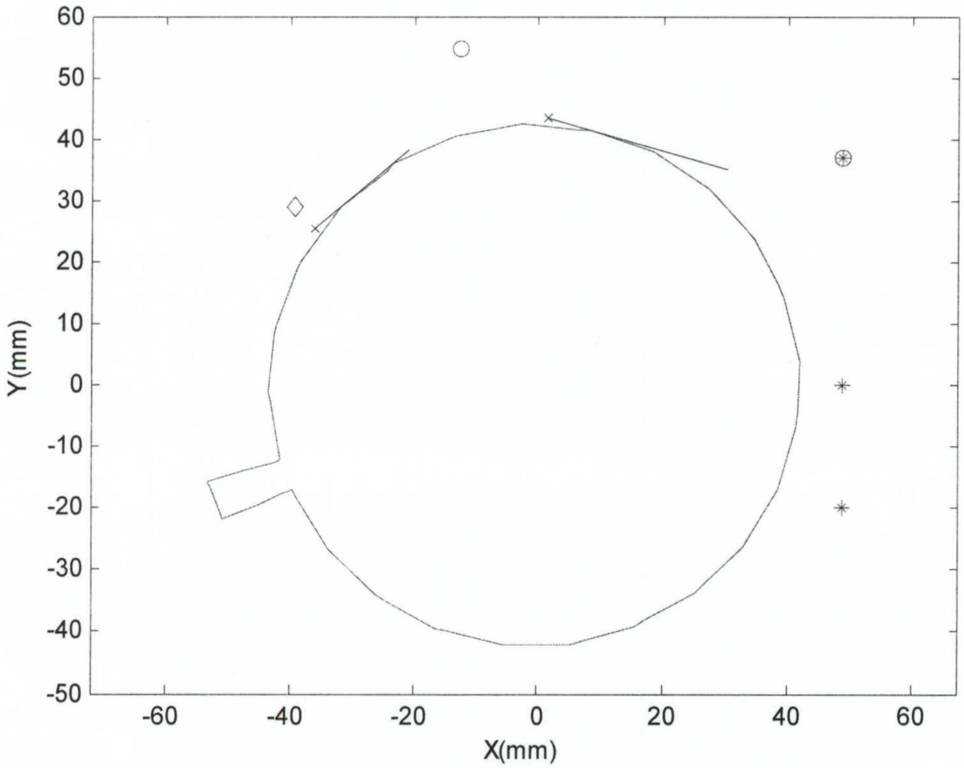

Fig. 4.16. Second phalange fixed.

The *angle of contact,* $\theta_2 = 122.9^\circ$ is within the *angular range* of second joint and the contact line has no interference with the object.

Besides checking the object interference with the contact line the program also checks the interference condition with the tip of the second phalange. The tip is modeled by the line between the end of the *contact line*, the $\times$ mark, and the top of the finger point, the diamond-shaped mark in Fig.4.16. There is no penetration and second phalange placement was completed successfully.

4.5.5. ALS Checking

The results given so far suggest that the *Finger_1* centre ALS has passed the finger positioning check. However as previously mentioned[1] each finger is modeled with three ALS, and each ALS must be positioned over the object and checked. If any fails then the grasp will fail, and if not there is still one more test to perform before passing the finger positioning. The angular deviations between the joint angles found for the three ALS for each finger should be less than the *Joint Angular Deviation* detailed in section 3.6.1.1. The *top* ALS and *bottom* ALS were successfully positioned. Their corresponding joint angles and *angular deviations* are as follows:

$$Center\_ALS:\{\theta_1 = 106.2^\circ \quad \theta_2 = 122.9^\circ\} \tag{4.2}$$

$$Top\_ALS:\begin{cases}\theta_1 = 107.1^\circ \rightarrow \Delta\theta_1 = 0.9^\circ < DeviationRange\\ \theta_2 = 123.7^\circ \rightarrow \Delta\theta_2 = 0.8^\circ < DeviationRange\end{cases} \tag{4.3}$$

$$Bottom\_ALS:\begin{cases}\theta_1 = 105.2^\circ \rightarrow \Delta\theta_1 = 0.9^\circ < DeviationRange\\ \theta_2 = 122.1^\circ \rightarrow \Delta\theta_2 = 0.8^\circ < DeviationRange\end{cases} \tag{4.4}$$

Both Top ALS and Bottom ALS passed the *angular deviation test* and as a result the positioning of Finger_1 was accepted.

[1] Refer to Finger Modeling section 3.6.1.1.

4.6. Final Output

The same routine described for *Finger_1*, was repeated for the remaining two fingers. If at any stage the positioning failed then the grasp for that position was rejected and the process was restarted for the next point in the search domain. The *Grasp Parameter Set* found for this object is as follows:

$GraspCondition \rightarrow Passed$

$$Finger\_1 \rightarrow \theta_1 = 106.2^\circ \quad \theta_2 = 122.9^\circ \tag{4.5}$$

$$Finger\_2 \rightarrow \theta_1 = 102.4^\circ \quad \theta_2 = 120.0^\circ \tag{4.6}$$

$$Finger\_3 \rightarrow \theta_1 = 98.3^\circ \quad \theta_2 = 116.0^\circ \tag{4.7}$$

$$ApproachAngle = 70^\circ \qquad Palm\_Pitch\_Angle = 7.3^\circ \tag{4.8}$$

Finger_1 Approach Elevation index=26 $Z = 69.5mm$. (4.9)

Note the output only lists the joint angles found for the centre ALS for each finger. Figures 4.17 - 4.19 show three different views of the grasp found with the 3D CAD model of the hand and the object.

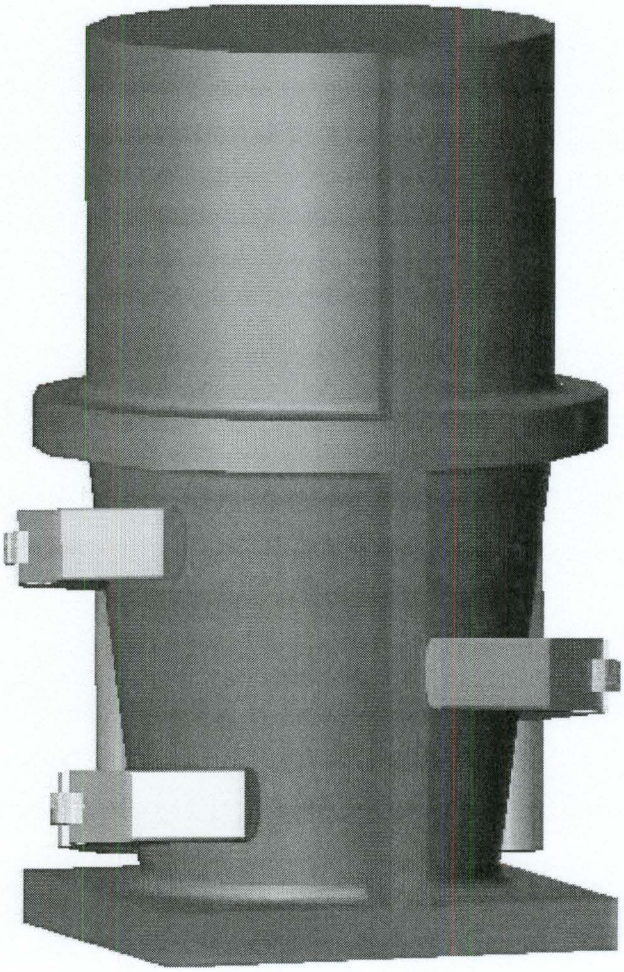

Fig.4.17. First view of the 3D object grasp.

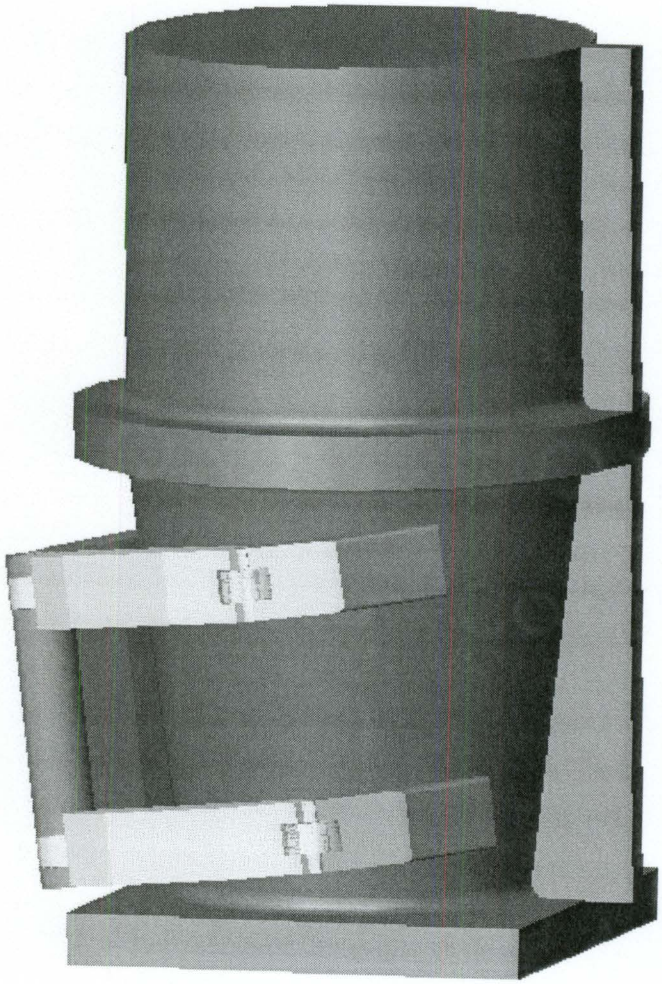

Fig.4.18. Second view of the 3D object grasp.

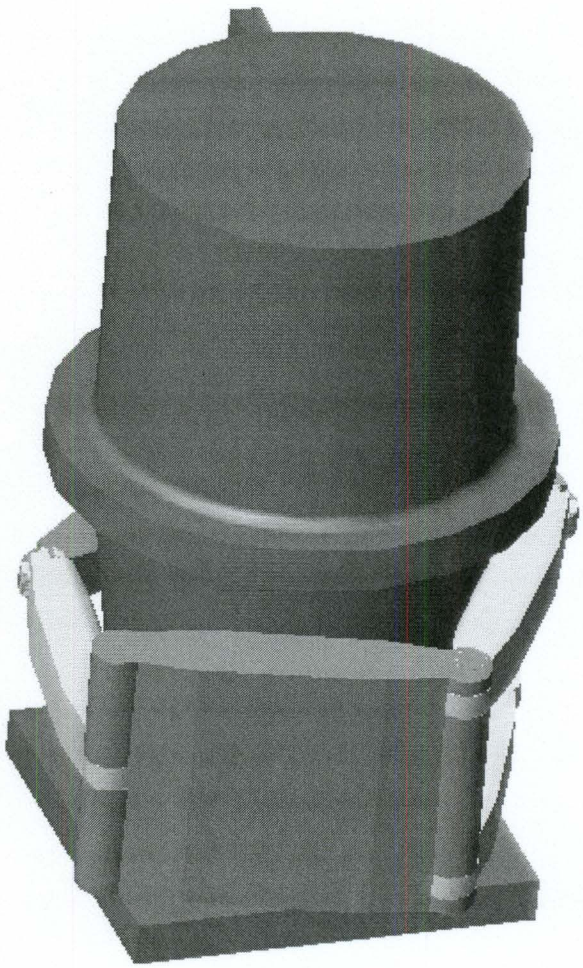

Fig.4.19. Third view of the 3D object grasp.

Chapter 5: Generality

In the previous chapter the grasp planning method was tested using a single object. Ideally a grasp planning method should be applicable to a wide range of object shapes. In this chapter the generality of the developed method will be explored by testing it with several object shapes and sizes. The program parameters were kept constant for all the tests.

5.1. Rectangular Prism

Rectangular prism shaped objects occur in industrial environments (e.g. square tubing) and in service applications (e.g. boxes and cartons). The prism shown in Fig.5.1 has the dimensions $15mm \times 30mm \times 124mm$. The grades after implementing significance factors are plotted in Fig.5.2. Since the *Effective Diameter* is constant (and equals $23mm$) the grades are constant except near the top and bottom of the *Effective Elevation.* At these levels the proximity to the extents triggers *Cat_6* (near the top) and *Cat_3* (near the bottom) grades. After analyzing the palm vertical dimension, negative grades are assigned to the portion where Finger_2 goes past the bottom of the *Effective Elevation*. The result of positioning Finger_1 is shown in Fig.5.4. The formatting of this figure is the same as that used in chapter 4. A view of the 3D grasp is presented in Fig.5.5.

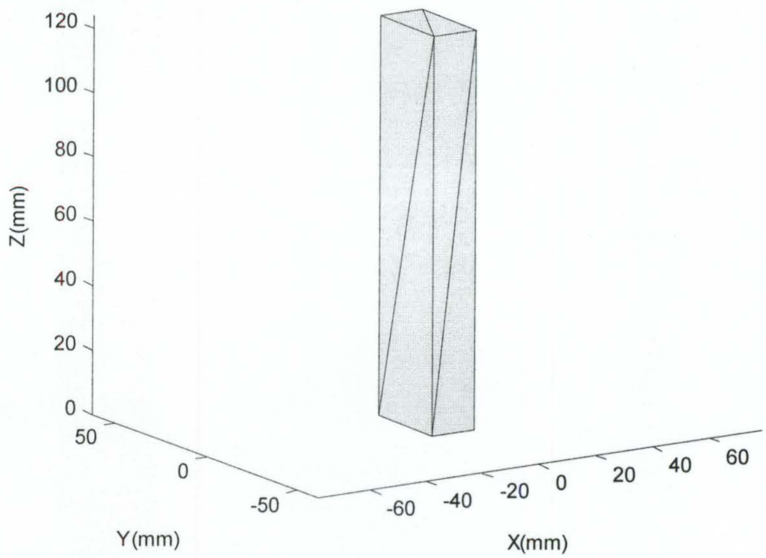

Fig.5.1. Rectangular prism in STL format

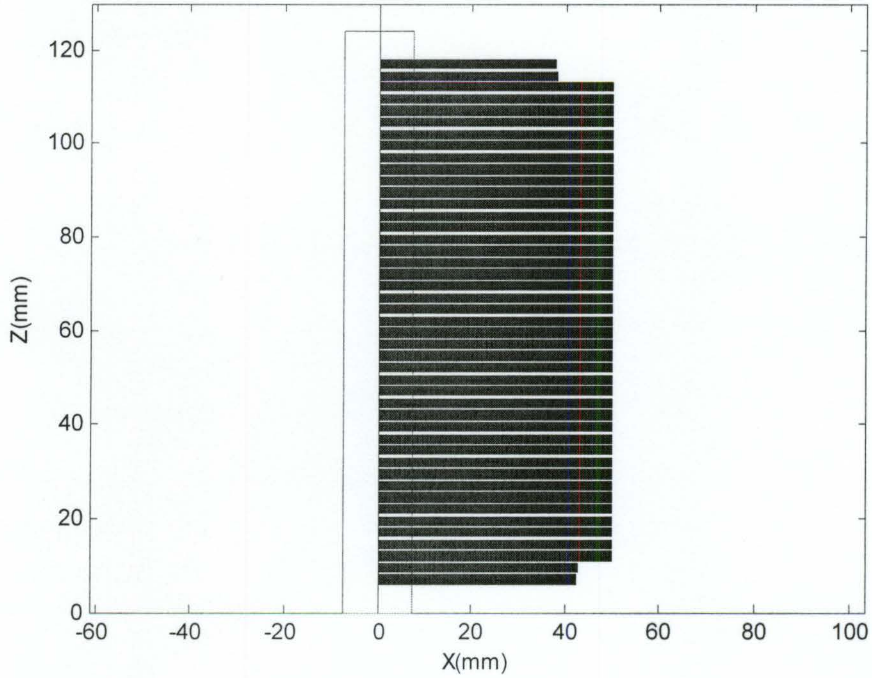

Fig.5.2. Grades after implementing the Significance factors, (Horizontal scale of 50)

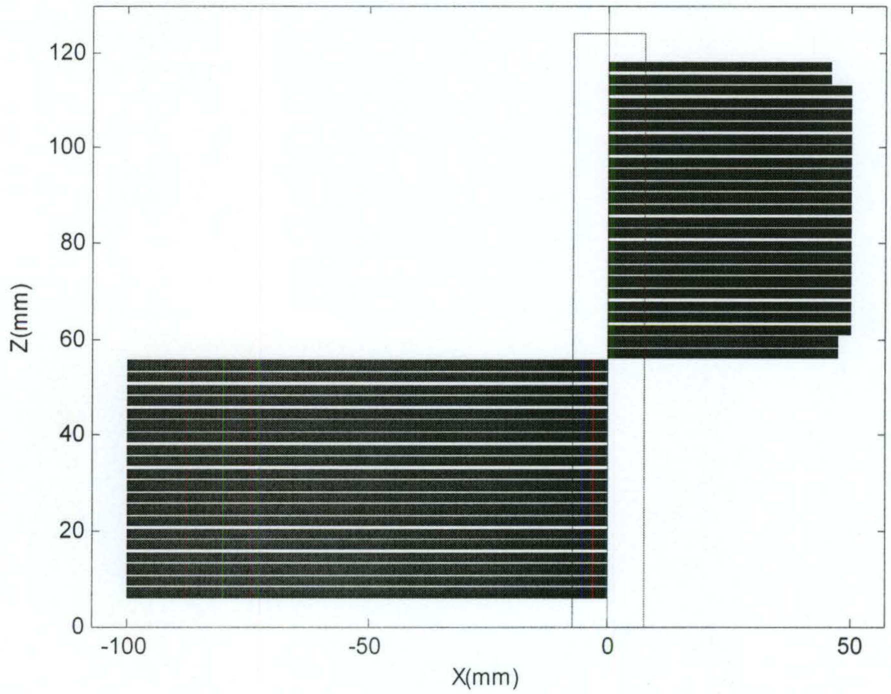

Fig.5.3. Grades after adding the palm vertical dimension

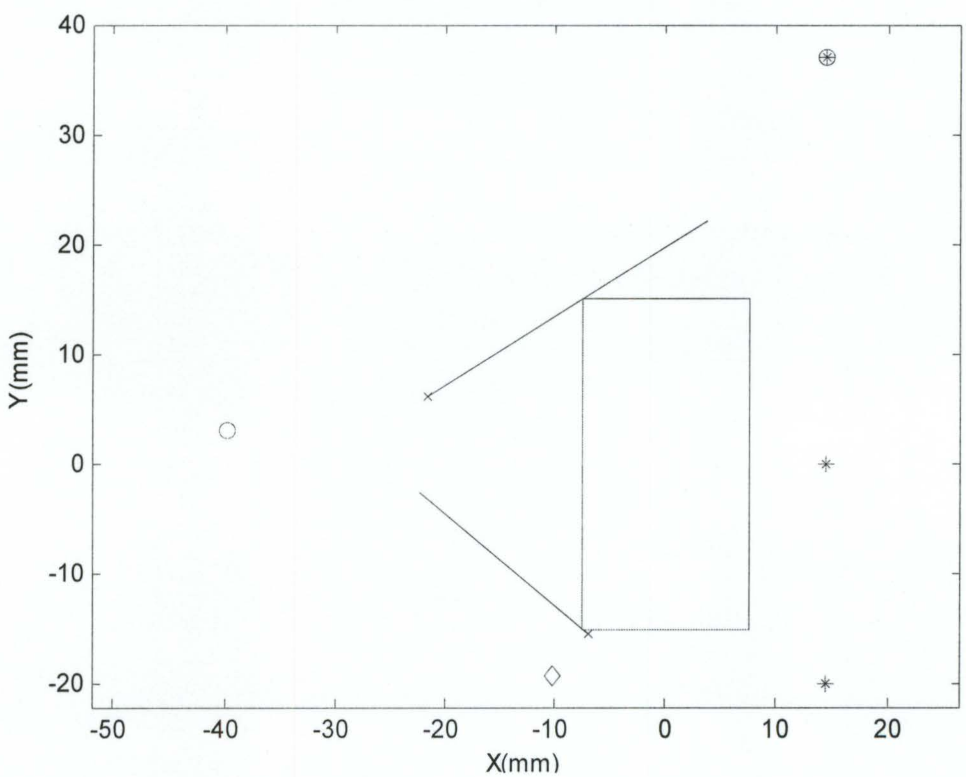

Fig.5.4. Finger_1 centre ALS fixed.

The *grasp parameter set* is:

$GraspCondition \rightarrow Passed$

$Finger\_1 \rightarrow \theta_1 = 58.0^\circ \quad \theta_2 = 72.2^\circ$ (5.1)

$Finger\_2 \rightarrow \theta_1 = 58.0^\circ \quad \theta_2 = 72.2^\circ$ (5.2)

$Finger\_3 \rightarrow \theta_1 = 58.0^\circ \quad \theta_2 = 72.2^\circ$ (5.3)

$ApproachAngle \rightarrow 0^\circ \qquad Palm\_Pitch\_Angle \rightarrow 0^\circ$ (5.4)

Finger_1 Approach Elevation index=43 $Z = 112mm$. (5.5)

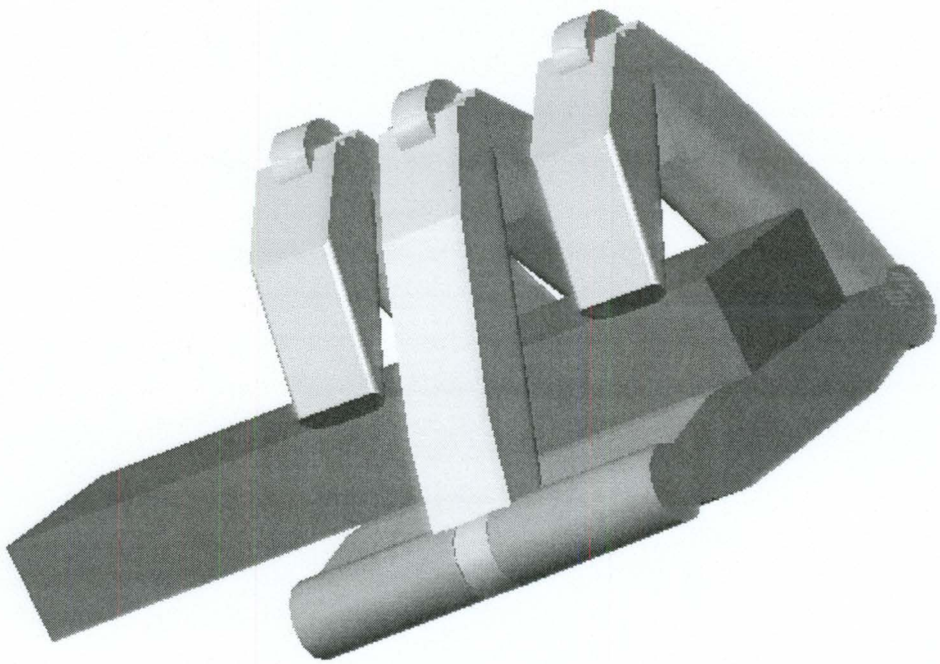

Fig.5.5. View of the 3D grasp.

As described in section 3.6.2.3, the program can also detect contact between the tip of the second phalange and the object. To demonstrate this, the length of the second phalange was reduced from $20mm$ to $18mm$ and the grasp was re-planned for the rectangular prism object. As shown by Fig.5.6 the program calculated the resulting tip contact properly.

The *Grasp Parameter Set:*

$$Finger\_1 \rightarrow \theta_1 = 58.0^\circ \quad \theta_2 = 68.6^\circ \tag{5.6}$$

$$Finger\_2 \rightarrow \theta_1 = 58.0^\circ \quad \theta_2 = 68.6^\circ \tag{5.7}$$

$$Finger\_3 \rightarrow \theta_1 = 58.0^\circ \quad \theta_2 = 68.6^\circ \tag{5.8}$$

$$ApproachAngle \rightarrow 0^\circ \qquad Palm\_Pitch\_Angle \rightarrow 0^\circ \tag{5.9}$$

Finger_1 Approach Elevation index=43, $Z = 112mm$. (5.10)

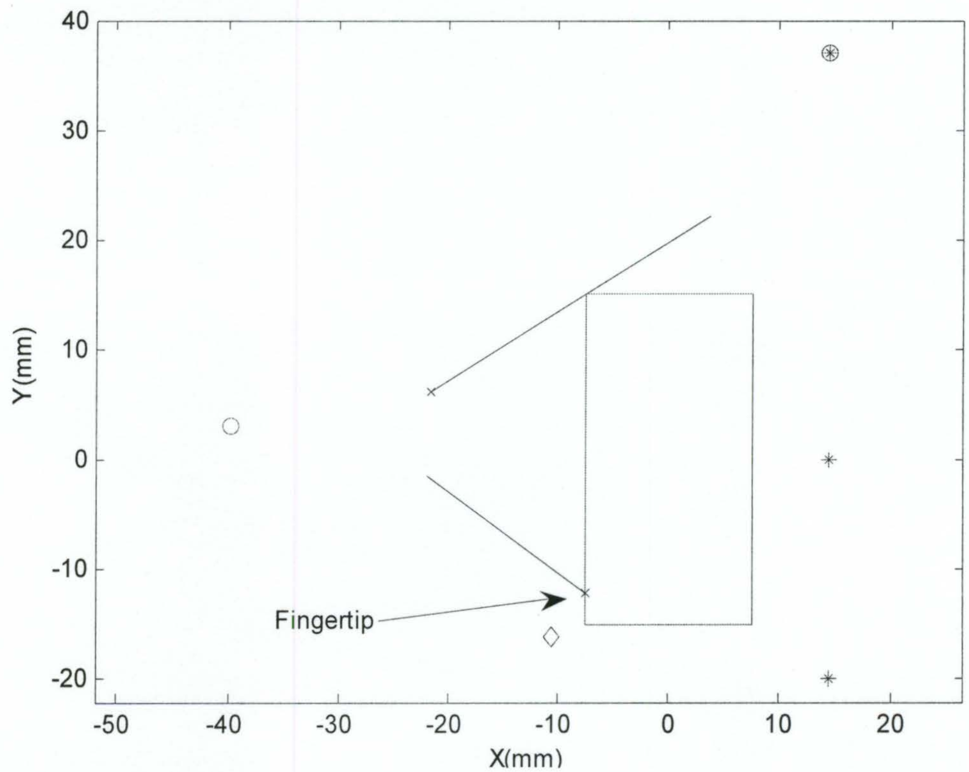

Fig.5.6. Finger_1 centre ALS fixed.

5.2. Cylinders

The program was also tested with three cylinders $124mm$ long, with diameters of $D1 = 10mm$, $D2 = 40mm$ and $D3 = 60mm$, respectively. The first and third cylinders were classified as ungraspable by the program as a result of their *Effective Diameters* being less than the *Smallest Graspable Diameter* and greater than the *Largest Graspable Diameter*, respectively. The STL model of the second cylinder is shown in Fig.5.7, the grading results, given in Fig.5.8 and 5.9, follow the same pattern as those for the rectangular prism. The Finger_1 positioning and the 3D grasp are shown in Fig.5.10 and 5.11 respectively.

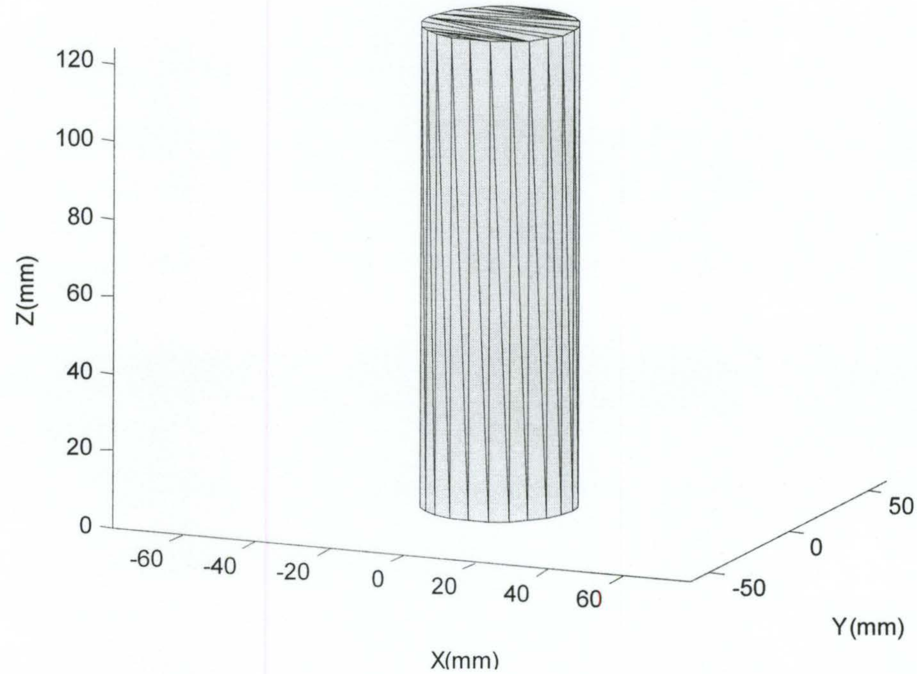

Fig.5.7. Cylinder with $40mm$ diameter in STL format

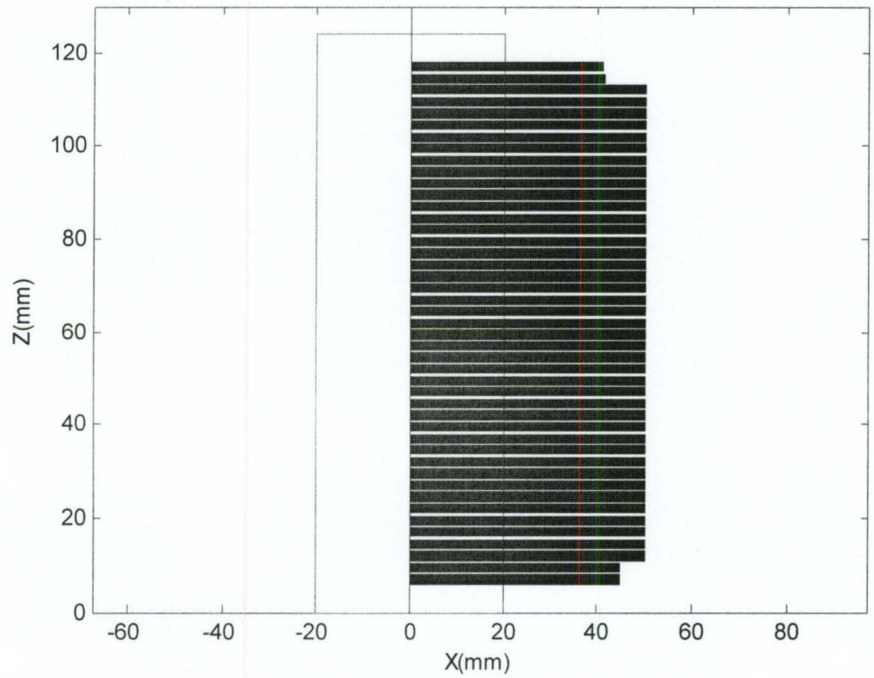

Fig.5 8. Grades after implementing the Significance factors, (Horizontal scale of 50)

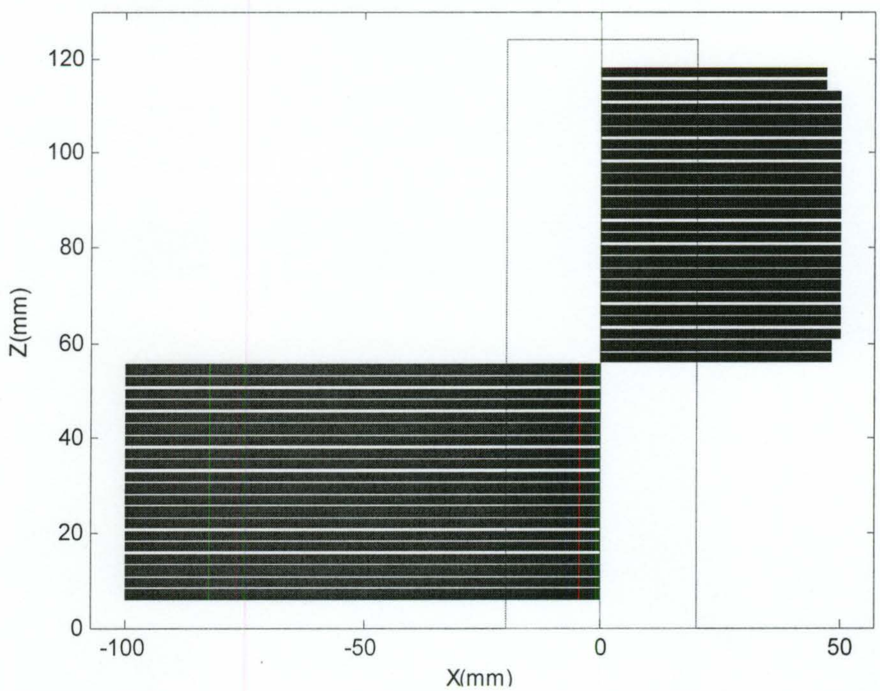

Fig.5.9. Grades after adding the palm vertical dimension

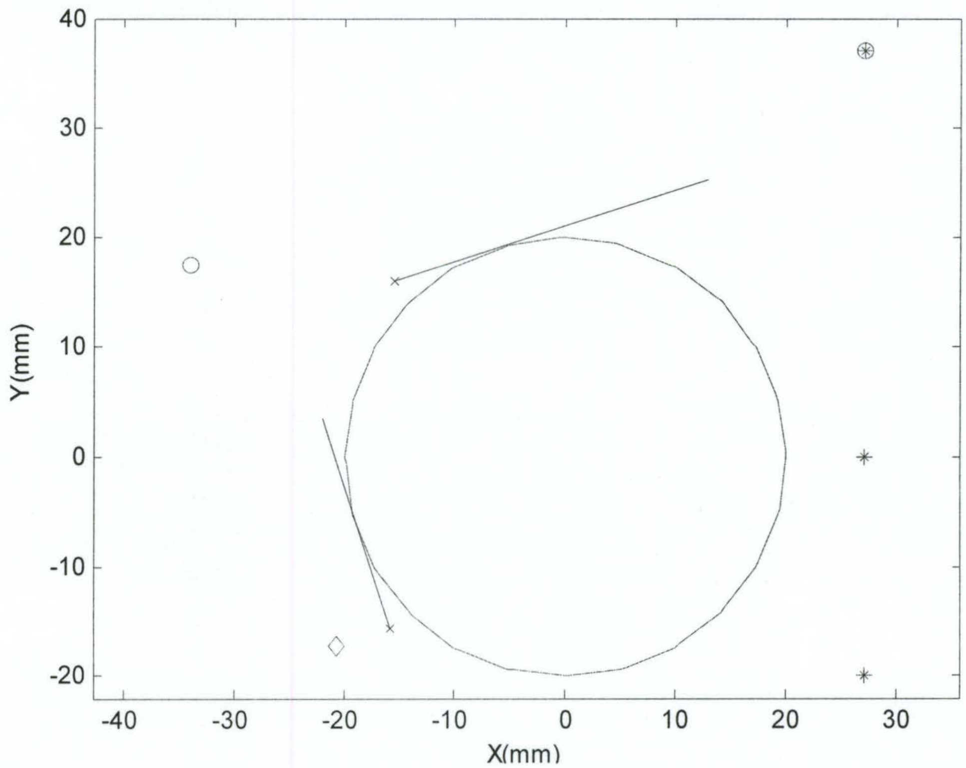

Fig.5.10. Finger_1 centre ALS fixed.

The *Grasp Parameter Set* is:

$$Finger\_1 \rightarrow \theta_1 = 72.2^\circ \quad \theta_2 = 90.0^\circ \tag{5.11}$$

$$Finger\_2 \rightarrow \theta_1 = 72.2^\circ \quad \theta_2 = 90.0^\circ \tag{5.12}$$

$$Finger\_3 \rightarrow \theta_1 = 72.2^\circ \quad \theta_2 = 90.0^\circ \tag{5.13}$$

$$ApproachAngle \rightarrow 0^\circ \qquad Palm\_Pitch\_Angle \rightarrow 0^\circ \tag{5.14}$$

Finger_1 Approach Elevation index=43, $Z = 112mm$. (5.15)

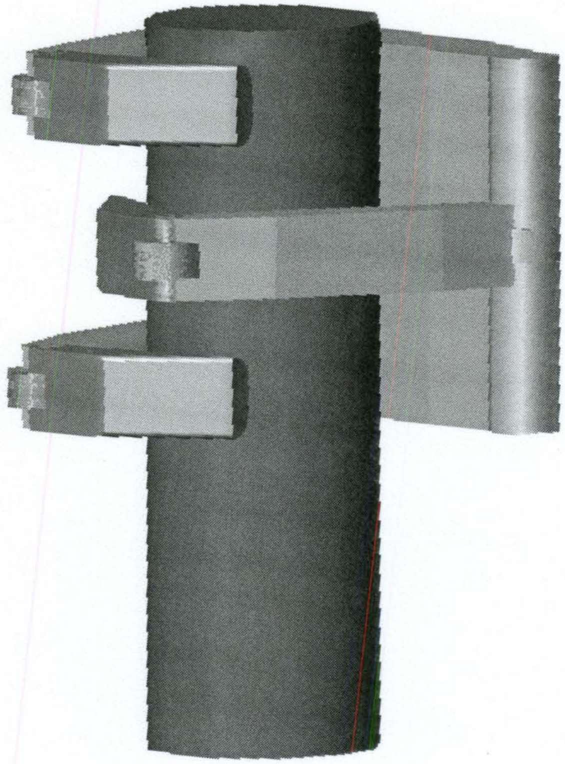

Fig.5.11. View of the 3D grasp.

5.3. Tapered Cylinder

Although the custom object employed in chapter 4 included a tapered cylinder element the interaction with the horizontal and vertical ridges made interpreting the grade result complicated. In this section a simple tapered cylinder, whose dimensions were taken from a plastic drinking cup, will be tested. The object (shown in Fig.5.12 in STL format) has a height of $110mm$, a bottom diameter of $56mm$ and a top diameter of $80mm$. The taper angle is 7°. The grades plotted in Fig.5.13 and Fig.5.14 clearly show the influence of the diameter changing with the elevation. The Finger_1 positioning result is shown in Fig.5.15. The vertical palm positioning result is shown in Fig.5.16.

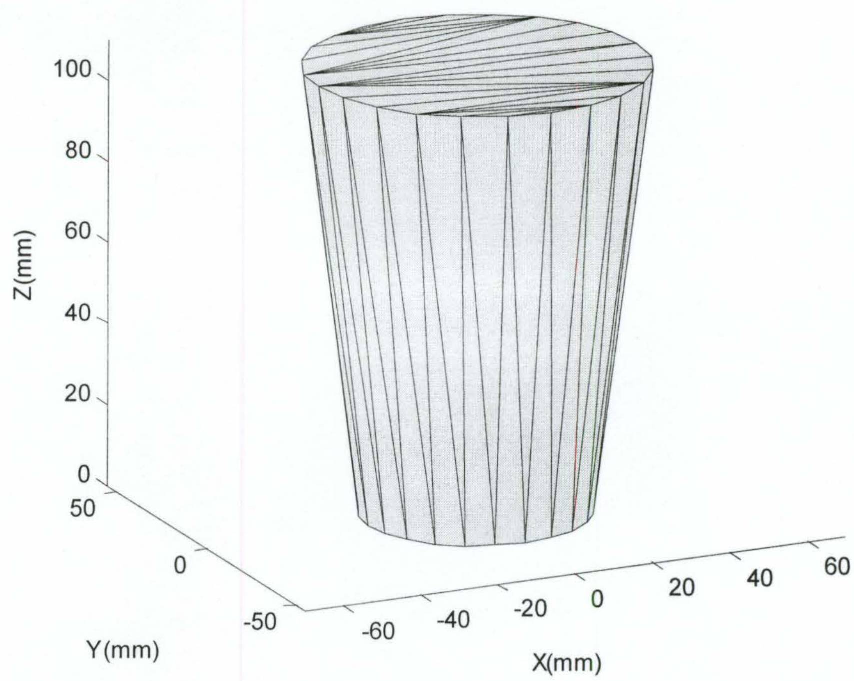

Fig.5.12. Tapered cylinder in STL format

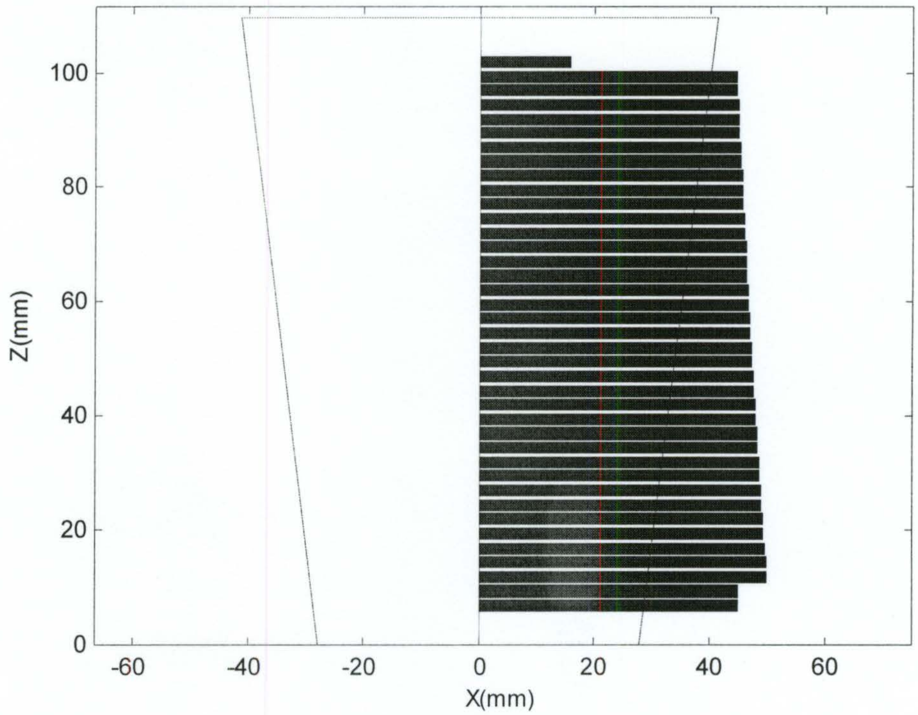

Fig.5.13. Grades after implementing the Significance factors, (Horizontal scale of 50)

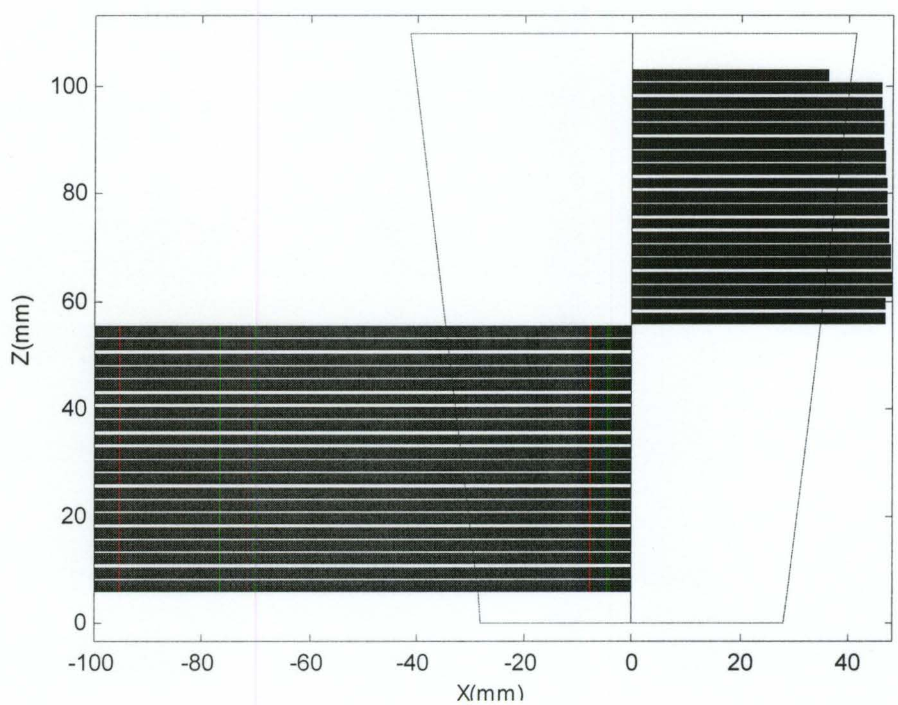

Fig.5.14. Grades after adding the palm vertical dimension

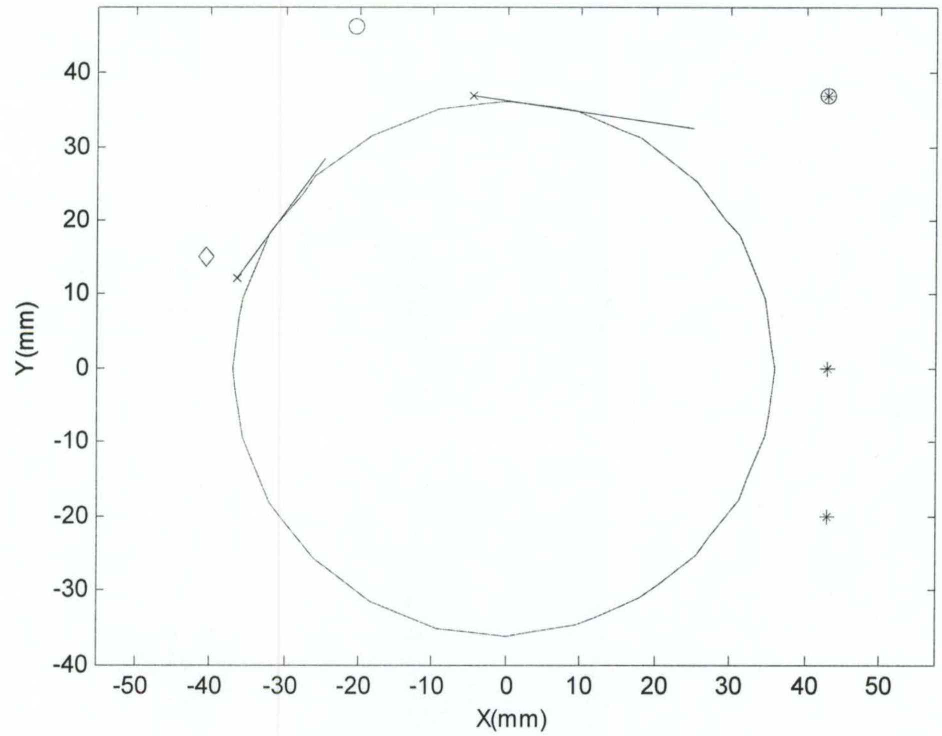

Fig.5.15. Finger_1 centre ALS fixed.

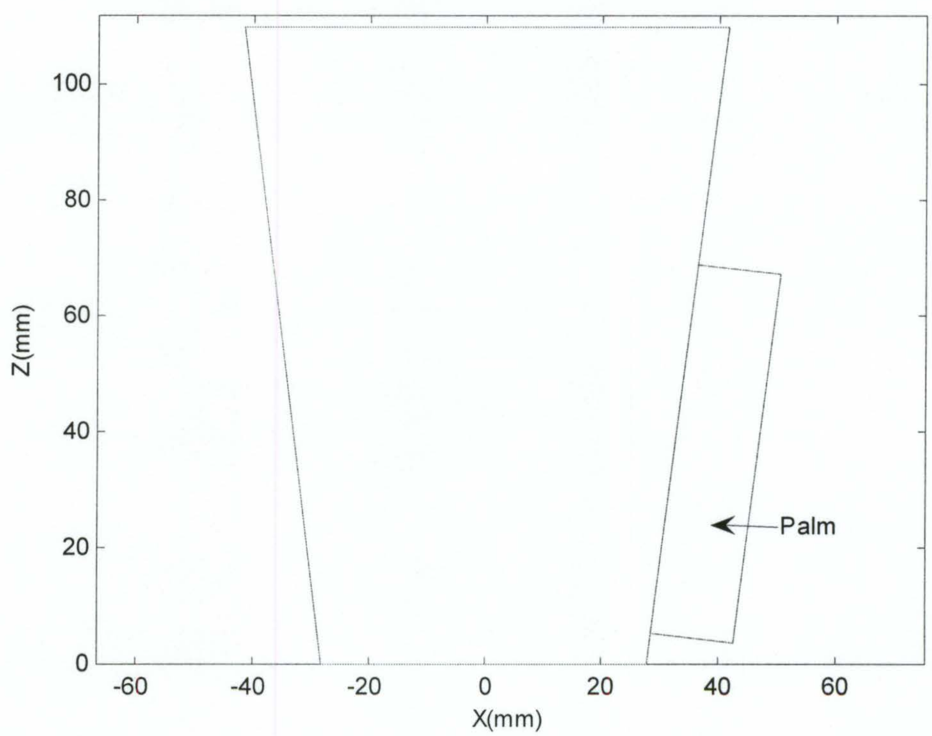

Fig.5.16. Vertical palm positioning

The *Grasp Parameter Set* is:

$$Finger\_1 \rightarrow \theta_1 = 98.3^\circ \quad \theta_2 = 117.6^\circ \tag{5.16}$$

$$Finger\_2 \rightarrow \theta_1 = 94.4^\circ \quad \theta_2 = 113.1^\circ \tag{5.17}$$

$$Finger\_3 \rightarrow \theta_1 = 89.9^\circ \quad \theta_2 = 108.5^\circ \tag{5.18}$$

$$ApproachAngle \rightarrow 0^\circ \qquad Palm\_Pitch\_Angle \rightarrow -7^\circ \tag{5.19}$$

Finger_1 Approach Elevation index=23, $Z = 62mm$. (5.20)

5.4. Inverted Tapered Cylinder

For comparison purpose an inverted version of the tapered cylinder studied in section 5.3 was tested. The STL model is shown in Fig.5.17. As expected, the grades are essentially an inverse of those calculated for the tapered cylinder, see Fig.5.18 and Fig.5.19. The Finger_1 positioning result and the vertical palm positioning result are given in Figs.5.20 and 5.2, respectively. Comparing Fig.5.21 with Fig.5.16 it can be observed that the method positioned the palm in a vertically consistent fashion with regards to the shape of the objects.

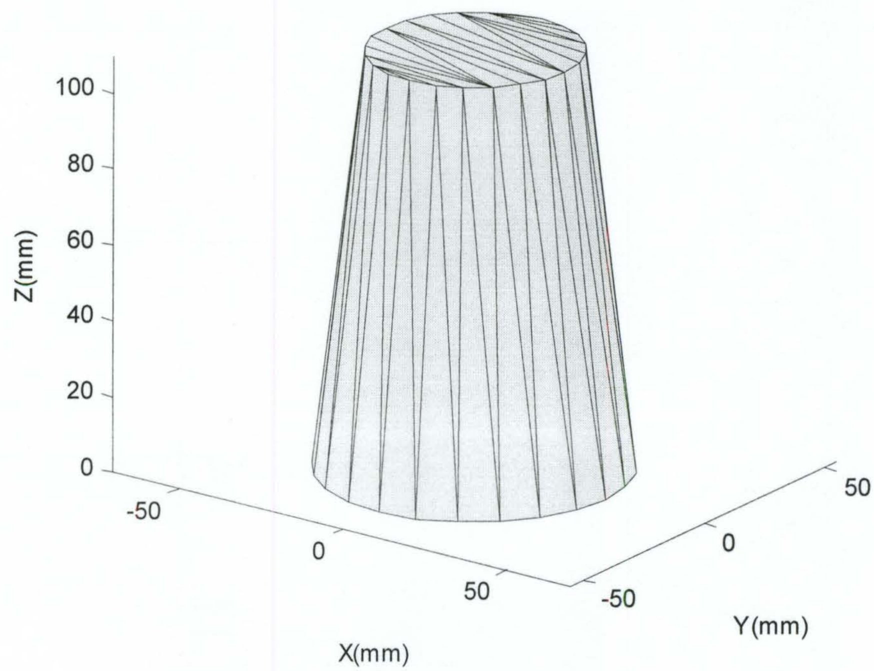

Fig.5.17. Inverted tapered cylinder STL format

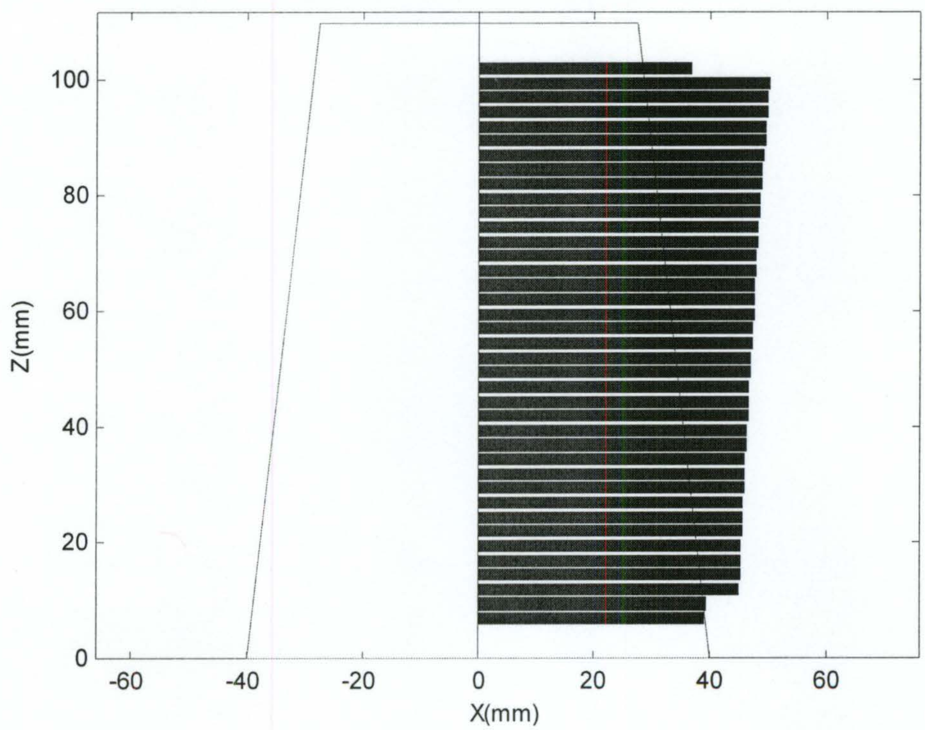

Fig.5 18. Grades after implementing Significance factors, (Horizontal scale of 50)

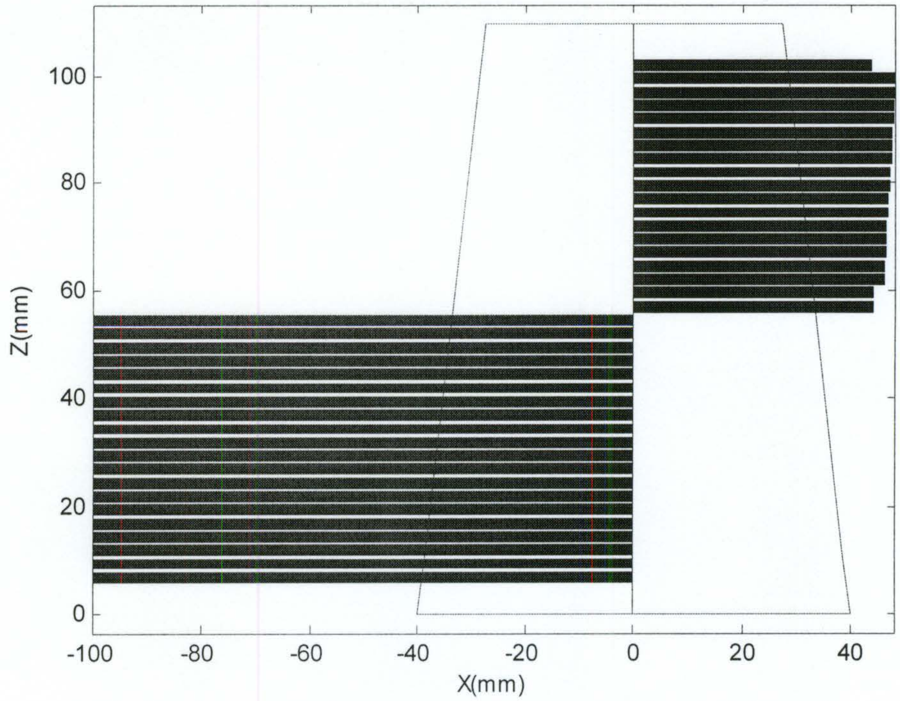

Fig.5.19. Grades after adding the palm dimension

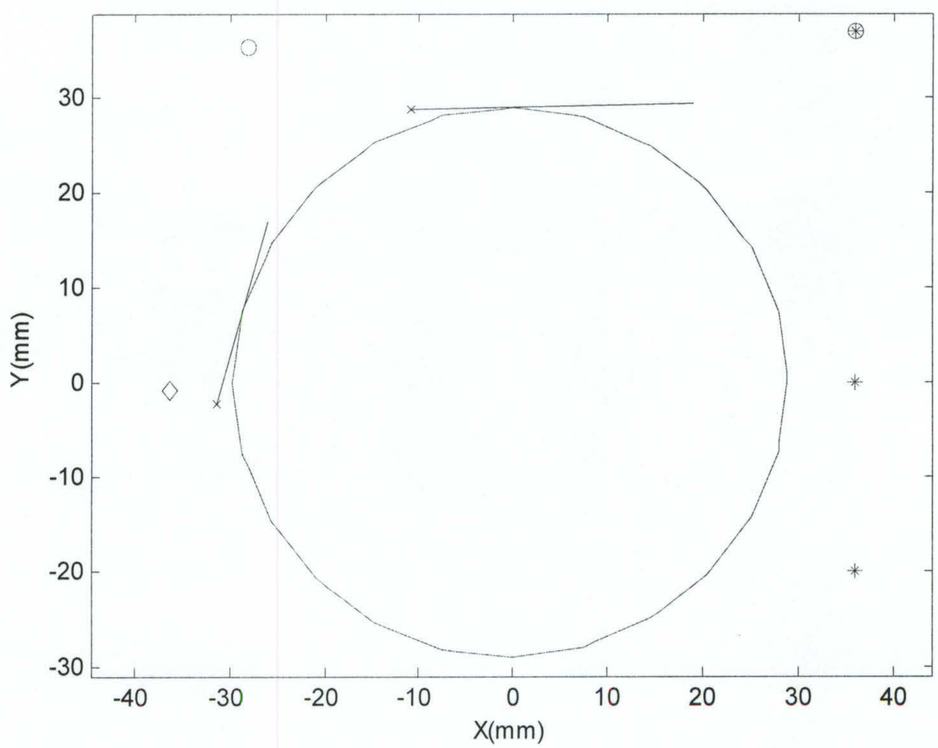

Fig.5.20. Finger_1 centre ALS fixed.

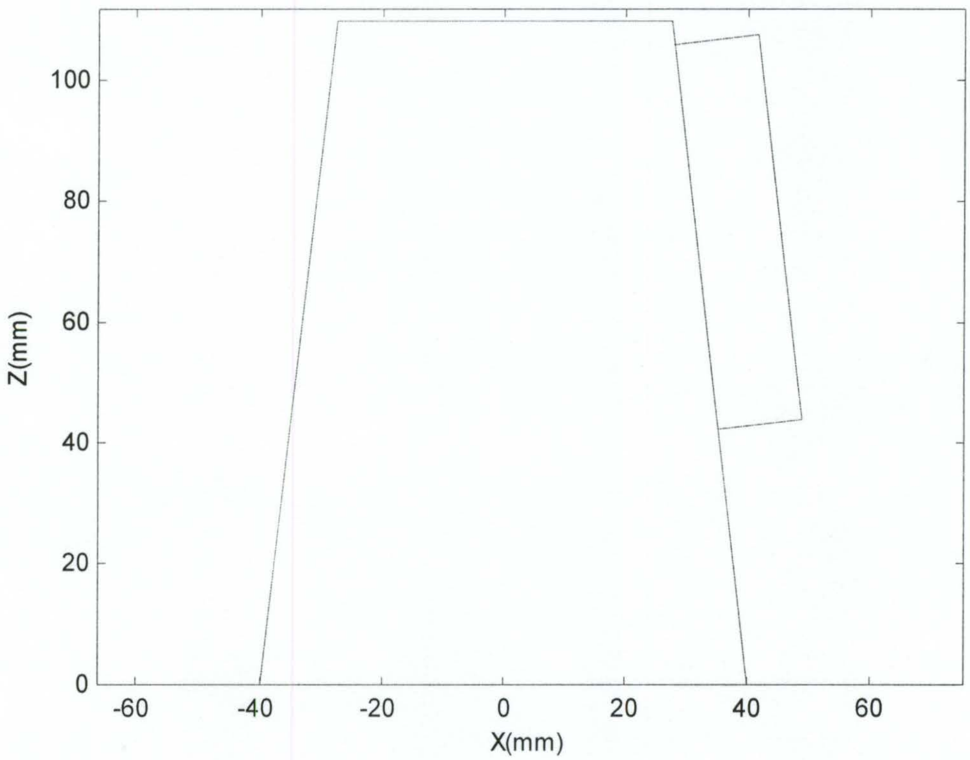

Fig.5.21. Vertical palm positioning

The *Grasp Parameter Set* is:

$$Finger\_1 \rightarrow \theta_1 = 88.5^\circ \quad \theta_2 = 107.2^\circ \tag{5.21}$$

$$Finger\_2 \rightarrow \theta_1 = 92.9^\circ \quad \theta_2 = 110.9^\circ \tag{5.22}$$

$$Finger\_3 \rightarrow \theta_1 = 96.7^\circ \quad \theta_2 = 115.7^\circ \tag{5.23}$$

$$ApproachAngle \rightarrow 0^\circ \qquad Palm\_Pitch\_Angle \rightarrow 6.5^\circ \tag{5.24}$$

Finger_1 Approach Elevation index=38, $Z = 99.5mm$. (5.25)

Chapter 6: Conclusions

In this thesis a new method is presented for 3D enveloping grasp planning with a dexterous hand. The objective of this method is to find the kinematically feasible 3D enveloping grasp for an arbitrary object whose 3D model is available, by applying the kinematic constraints of a three fingered dexterous hand. The method was successfully tested with several object shapes and sizes.

6.1. Accomplishments

- A new method for defining the search domain for grasp planning is proposed. This method has three significant benefits:
  - By using the grasp kinematical constraints (i.e. hand dimensions, curvature constraint and thickness constraints), the method eliminates the ungraspable areas from the search domain. As a result the remaining search domain has a better probability of a successful grasp.
  - Before entering the finger planning level the method will consider the kinematical constraints of the palm geometry and eliminates all the areas of the search domain in conflict with these constraints.

- In the process of allocating the search domain, if any of the object areas is potentially graspable, the method assigns a grade to it based on the kinematical preferences of palm and finger positioning in enveloping grasp (specifically, the Curvature Patterns and Effective Diameter). The method uses these grades to sort the graspable members of the search domain. This sorting system directs the planning system to test the areas of the object which should result in a better quality grasp first.

The result of the method is a reduced and graded search domain. It is important to have a smaller search domain since it requires less calculation time to analyze. It is important to have a graded search domain since it exposes the places with potentially better quality of the grasp early in the planning process.

- A simple and practical method is proposed to model the 3D finger for enveloping grasp planning. The great benefit of the new model is that it frees the grasp planning process of the complex calculations of finding the real finger contact points with the object. In the proposed method each finger is modeled with three articulated line segments (ALS) which are bound to each other by a kinematic constraint (the *Joint Angular Deviation*). Each of these ALS are tested with the object separately. In order to have a successful contact they have to comply with the kinematic constraint of the hand (e.g. joint movement range) and the kinematic constraint that bounds them together.

- In the proposed method each of the grasp parameters (i.e. palm position, palm orientation and joint angles) is calculated in separate stages. If in each stage the program finds the grasp is unfeasible it will stop the current analysis and will go to the next available member of the search domain. This considerably simplifies the numerical analysis and keeps the grasp planning procedure free of any redundant calculations.
- The proposed method has successfully been implemented using a MATLAB program and various 3D object models. The program is composed of 57 functions and 4300 lines of code.

6.2. Limitations

Although the enveloping grasp planning method proposed has shown promising results it has some limitations.

- A 3D model of the object in STL format is required. This model can be obtained from an existing CAD model of autonomously using a vision system or laser scanner.
- In the STL format model of the object, the principal axis with the smallest principal moment of inertia and the global Z axis should be collinear.
- The method assumes that the object is free of holes or voids.

- Since the method does not find the exact contact points with the object a tuning method might be needed to adjust the grasp parameters found before performing the grasp with a robotic hand.
- A grasp that is kinematically feasible may not be feasible when forces and moments are considered.

6.2. Recommendations for Future Work

- Extending the method to analyze the force and moment properties of the grasp.
- Analyzing the probable grasps around the other principal axes of the object and try to find the optimum grasp among the answers found.
- Implementing an optimization method to find the best enveloping grasp based on the force properties and kinematical properties of the enveloping grasps found.
- Extending the method to consider holes in the object and use their geometric benefits (e.g. a better resting point for fingers) to find a more stable grasp.
- Modify the method to cover dexterous hands with more DOF.

References

[1] W. T. Townsend, "The BarrettHand grasper–programmably flexible part handling and assembly". Industrial Robot: An International Journal, Vol.27, No.3, pp.181—188, 2000.

[2] Mark R. Cutkosky, "On grasp choice, grasp models, and the design of hands for manufacturing tasks". IEEE Transactions on Robotics and Automation, Vol. 5, No. 3, pp. 269-279, June 1989.

[3] D. Wren and R. B. Fisher," Dexterous hand grasping strategies using preshapes and digit trajectories". Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, pp. 910-915, 1995.

[4] H. Asada and M. Kitagawa," Kinematic analysis and planning for form closure grasps by robotic hands". Robotics and Computer Integrated Manufacturing, Vol. 5, No. 4, pp. 293-299, 1989.

[5] B. Mirtich and J. Canny, " Easily computed optimum grasps in 2-D and 3D". Proceedings of the IEEE International Conference on Robotics and Automation, pp. 739-747, 1994.

[6] T. Yoshikawa, "Passive and active closures by constraining mechanisms". Proceedings of the IEEE International Conference on Robotics and Automation, Minneapolis, Minnesota, pp. 1477-1484, April 1996.

[7] M. Kaneko, Y. Hino and T. Tsuji, "On three phase of achieving envelope grasps". Proceedings of the IEEE International Conference on Robotics and Automation, Albuquerque, New Mexico, pp. 835-390, April 1997.

[8] G. Smith, E. Lee, K. Goldberg, K. Bohringer, J. Craig, " Computing parallel-jaw grippers". IEEE International Conference on Robotics and Automation, Detroit, Michigan, pp. 1897-1903, May 1999.

[9] C. Hwang, M. Takano and K. Sasaki,"Kinematics of grasping and manipulation of a B-Spline surface object by a multifingered robot hand". Journal of Robotic Systems, Vol. 16, No. 8, pp. 445-460, 1999.

[10] C. Borst, M. Fischer and G. Hirzinger, "A fast and robust grasp planner for arbitrary 3D object". Proceedings of the IEEE International Conference of Robotics and Automation. Pp. 1890-1898, May 1999.

[11] M. A. Moussa and A. Serban, " AMSEK: A simulator for robotic grasping of arbitrarily shaped objects". Proceedings of the International Conference on Computer and Industrial Engineering, pp. 641-646, 2002.

[12] Y. Guan and H. Zhang," Feasibility analysis of 2D grasps". Proceedings of the IEEE/RSJ International Conference on Intelligent Robotics and Systems, Las Vegas, Nevada, pp. 3435-3440, October 2003.

[13] A. T. Miller, S. Knoop, H. I. Christensen and P. K. Allen, "Automatic Grasp Planning Using Shape Primitives". Proceedings of the IEEE International Conference of Robotics and Automation, pp. 1824-1829, 2003.

[14] Y. Guan and H. Zhang, "Kinematic feasibility analysis of 3-D multifunctional grasps". IEEE Transactions on Robotics and Automation, Vol. 19, No. 3, pp. 507-513, June 2003.

[15] L. Balan and G. M. Bone, " Automated gripper jaw design and grasp planning for sets of 3D objects". Journal of Robotic Systems, pp. 147-162, 2003.

[16] E. Lopez-Damian, D. Sidobre and R. Alami, " A grasp planner based on inertial properties". Proceeding of the IEEE International Conference on Robotics and Automation, Barcelona, Spain, April 2005.

[17] Collide Research Group, Computer Science Department, University of North Carolina," SWIFT++: A new collision detection package", http://valis.cs.-uiuc.edu/~sariel/research/CG/compgeom/msg00417.html. May, 2001.

[18] "Inscribed Circle", http://www.mathwords.com/i/inscribed_circle.htm.