INTELLIGENT GRIPPER DESIGN AND APPLICATION

FOR

AUTOMATED PART RECOGNITION AND GRIPPING

by

Jianqiang Wang

A thesis submitted in compliance with the full requirements for the Doctor Technologiae: Engineering

In the

Faculty of Electrical and Mechanical Engineering

Port Elizabeth Technikon

October 2002

Promoters

Dr Theo van Niekerk

Dr Danie Hattingh
This thesis is dedicated to
my wife Jie Yang, my daughter QingNing Wang,
and my parents
who have been a constant source of support and understanding
Jianqiang Wang hereby declares that:

- At no time during the registration for the degree of Doctor Technologiae has the author been registered for any other university degree:

- the work done in the thesis is his own; and

- all sources used or referred to have been documented and recognized.
Acknowledgements

I would express my gratitude to the following people and institution for their assistance, time, support, and encouragement:

- My promoter, Dr Theo van Niekerk, for his support, guidance and encouragement
- My promotor, Dr Daniel Hattingh, for his support, guidance and encouragement
- Prof Eugene du Preez for his support and encouragement
- Prof Hugh Jeffery for his support and encouragement
- Dr J Huang and all the team members in the Manufacturing Technology Research Centre and staff in the workshop for their assistance
- Hefei University of Technology for giving me the opportunity of pursuing my Ph.D at the Port Elizabeth Technikon
- My wife Jie Yang, my daughter QingNing Wang, and my parents for their love, support, sacrificing, and understanding throughout the duration of my research
Abstract

Intelligent gripping may be achieved through gripper design, automated part recognition, intelligent algorithm for control of the gripper, and on-line decision-making based on sensory data. A generic framework which integrates sensory data, part recognition, decision-making and gripper control to achieve intelligent gripping based on ABB industrial robot is constructed.

The three-fingered gripper actuated by a linear servo actuator designed and developed in this project for precise speed and position control is capable of handling a large variety of objects. Generic algorithms for intelligent part recognition are developed. Edge vector representation is discussed. Object geometric features are extracted. Fuzzy logic is successfully utilized to enhance the intelligence of the system. The generic fuzzy logic algorithm, which may also find application in other fields, is presented. Model-based gripping planning algorithm which is capable of extracting object grasp features from its geometric features and reasoning out grasp model for objects with different geometry is proposed. Manipulator trajectory planning solves the problem of generating robot programs automatically. Object-oriented programming technique based on Visual C++ MFC is used to constitute the system software so as to ensure the compatibility, expandability and modular programming design. Hierarchical architecture for intelligent gripping is discussed, which partitions the robot’s functionalities into high-level (modeling, recognizing, planning and perception) layers, and low-level (sensing, interfacing and execute) layers. Individual system modules are integrated seamlessly to constitute the intelligent gripping system.
Table of Contents

Abbreviations ................................................................................................................... vi

List of Figures ..................................................................................................................... vii

List of Tables ..................................................................................................................... x

Glossary ............................................................................................................................. xi

Chapter 1  Introduction ................................................................................................. 1

1.1  Aim ............................................................................................................................ 3

1.2  Objectives ................................................................................................................. 3

1.3  Hypothesis ............................................................................................................... 5

1.4  Methodological Justification .................................................................................... 5

1.5  Delimitations ............................................................................................................. 7

1.6  Significance of Research ........................................................................................ 8

   1.6.1  Within the Technikon ....................................................................................... 8

   1.6.2  General .............................................................................................................. 8

1.7  Organisation of Thesis ............................................................................................. 9

Chapter 2  Automated Part Recognition and Intelligent Gripping:
Relevant Concepts, Framework and Technology ......................................................... 11

2.1  Intelligent Robot Overview ..................................................................................... 13

2.2  Robot Vision System (Image Processing, Model-based Geometry
Recognition, Geometric Feature Coding) ..................................................................... 16

   2.2.1  Digital Image Processing .................................................................................. 18
2.2.1.1 Image Acquisition ................................................................. 18
2.2.1.2 Illumination Techniques ....................................................... 19
2.2.1.3 Image Pre-processing .......................................................... 21
2.2.2 Model-based Geometry Recognition ........................................ 28
2.2.2.1 Geometric Feature Extraction and Recognition ..................... 29
2.2.3 Foundations of Fuzzy Feature Extraction and Recognition ............ 32
2.2.3.1 Fuzzy Inference System Design and Its Elements .................. 32
2.2.4 Fuzzy Set Classification .......................................................... 39
2.3 Intelligent Gripping ..................................................................... 40
2.4 Intelligent Robot Gripper (Multi-sensor Based Gripping Applications,
Gripper Structure & Design) .............................................................. 43
2.5 Robot Trajectory Planning ............................................................ 47
2.5.1 Rotation Matrices ................................................................. 48
2.5.2 Composite Rotation Matrix ...................................................... 49
2.5.3 Vision-Based Path Planning ..................................................... 50
2.6 PC-based Intelligent Robot Gripping Architecture ............................ 50
2.7 Conclusion .................................................................................. 52

Chapter 3 System Setup: Gripper Design, Data Acquisition, Gripper
Control, and Software Components .................................................. 54

3.1 System Framework (Robot, Gripper, Sensory System) ....................... 54
3.2 Linear Servo Actuator Design ....................................................... 56
3.2.1 Actuator Structure Design ....................................................... 57
3.2.2 Analysis of Actuator Operation ................................................. 59
3.3 Gripper Design ......................................................................... 61
3.3.1 Gripper Structure Design ......................................................... 62
3.3.2 Analysis of Gripping Process .................................................. 63
3.3.2.1 Analysis of Gripper Kinematics ........................................... 64
3.3.2.2 Analysis of Gripping Force ................................................ 65
3.4 Implementation of System Control ................................................. 67
3.4.1 Robot Vision System Control .................................................. 69
Chapter 4  Robot Vision System ................................................................. 89

4.1  Architecture of Robot Vision System ..................................................... 91

4.2  Image Processing ................................................................................ 92
   4.2.1  Image Sensing ............................................................................. 92
   4.2.2  Image Preprocessing ................................................................. 93
       4.2.2.1  Image Filtering ................................................................. 93
       4.2.2.2  Noise Cleaning ................................................................. 95
       4.2.2.3  Averaging ........................................................................ 95
       4.2.2.4  Thresholding ................................................................... 95
   4.2.3  Boundary Detection .................................................................... 96
       4.2.3.1  Edge Detection ................................................................. 97
       4.2.3.2  Edge Linking ................................................................... 100
       4.2.3.3  Edge Following and Thinning ........................................... 101

4.3  Object Recognition (Feature Extraction, Geometry Recognition, and
     Description) .................................................................................... 103
   4.3.1  Object Features ........................................................................ 103
   4.3.2  Extraction and Representation of the Profile ............................. 105
       4.3.2.1  Vector Tracing Method ................................................... 105
       4.3.2.2  Continuous Fuzzy Controller ......................................... 108

4.4  Object Classification .......................................................................... 112
4.4.1 Determination of Object Geometry ID .................................................. 113
4.4.2 Calculation of Object Orientation ....................................................... 115
4.5 Conclusion .............................................................................................. 116

Chapter 5 Model-Based Gripping Planning .............................................. 117

5.1 Object Grasp Modeling ........................................................................ 118
5.2 Extraction of Grasp Features ................................................................. 121
  5.2.1 Grasp Feature of Round Objects ....................................................... 122
  5.2.2 Grasp Feature of Wedge .................................................................. 122
  5.2.3 Grasp Feature of Block ................................................................... 123
  5.2.4 Grasp Feature of Parallelepiped ....................................................... 124
  5.2.5 Grasp Feature of Polyhedron ........................................................... 124
5.3 Model-Based Gripping Planning ............................................................ 125
  5.3.1 Gripping Planning for Round Object ................................................. 125
  5.3.2 Gripping Planning for Wedge Object ............................................... 126
  5.3.3 Gripping Planning for Block Object ................................................. 127
  5.3.4 Gripping Planning for Parallelepiped Object ................................... 128
  5.3.5 Gripping Planning for Polyhedron Object ....................................... 129
5.4 Conclusion .............................................................................................. 133

Chapter 6 Trajectory Planning and Intelligent Gripping ................. 134

6.1 Eye-hand Coordinate System Transformation .................................... 135
  6.1.1 Robot Coordinate Systems ............................................................... 135
  6.1.2 Transformation Matrices and Quaternion ....................................... 137
  6.1.3 Quaternion ..................................................................................... 139
  6.1.4 Computation of Object Frame ....................................................... 140
6.2 Robot Trajectory Planning .................................................................... 141
  6.2.1 RAPID Program Structure .............................................................. 141
  6.2.2 Robot Manipulator Path Planning ................................................... 142
6.3 Sensory Data Based Gripping Control and Analysis ....................... 143
# Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADC</td>
<td>Analog-to-Digital Converter</td>
</tr>
<tr>
<td>API</td>
<td>Application Interface</td>
</tr>
<tr>
<td>B-Rep</td>
<td>Boundary-Representation</td>
</tr>
<tr>
<td>CAD</td>
<td>Computer Aided Design</td>
</tr>
<tr>
<td>CCD</td>
<td>Charge-Coupled Device</td>
</tr>
<tr>
<td>COG</td>
<td>Centre of Gravity</td>
</tr>
<tr>
<td>CSG</td>
<td>Constructive Solid Geometry</td>
</tr>
<tr>
<td>DAC</td>
<td>Digital Analog Controller</td>
</tr>
<tr>
<td>DAQ</td>
<td>Data Acquisition Card</td>
</tr>
<tr>
<td>DLL</td>
<td>Dynamic Link Library</td>
</tr>
<tr>
<td>DOF</td>
<td>Degrees of Freedom</td>
</tr>
<tr>
<td>EVR</td>
<td>Edge Vector Representation</td>
</tr>
<tr>
<td>F/T</td>
<td>Force/Torque</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphics User Interface</td>
</tr>
<tr>
<td>MBF</td>
<td>Membership Function</td>
</tr>
<tr>
<td>MFC</td>
<td>Microsoft Foundation Class</td>
</tr>
<tr>
<td>OLP</td>
<td>Off-Line Programming</td>
</tr>
<tr>
<td>OOP</td>
<td>Object Oriented Programming</td>
</tr>
<tr>
<td>PC-ORC</td>
<td>PC-based open robot control</td>
</tr>
<tr>
<td>RGB</td>
<td>Red, Green, and Blue</td>
</tr>
<tr>
<td>S/N</td>
<td>Signal/Noise</td>
</tr>
<tr>
<td>TCP</td>
<td>Tool Centre Point</td>
</tr>
<tr>
<td>WMM</td>
<td>Weighted Mean Method</td>
</tr>
</tbody>
</table>
List of Figures

Figure 2.1: Generic intelligent gripper system [7][8] ................................................................. 14
Figure 2.2: Structure of cell mechanism facilitating OLP ............................................................ 15
Figure 2.3: General components in a robot vision system ............................................................. 17
Figure 2.4: Four basic illumination schemes [2] ........................................................................ 20
Figure 2.5: Typical two peaks intensity histogram [22] ............................................................. 24
Figure 2.6: Thresholding to a gray-level image .......................................................................... 24
Figure 2.7: The result of contour detection by applying Laplacian operator ............................... 26
Figure 2.8: (a) 4-directional chain code. (b) 8-directional chain code ....................................... 28
Figure 2.9: Hierarchical structure of features ........................................................................... 30
Figure 2.10: A general paradigm in knowledge-based feature recognition .................................. 31
Figure 2.11: Basic architecture of a fuzzy system ..................................................................... 33
Figure 2.12: Triangular membership function ............................................................................ 35
Figure 2.13: Result of fuzzification with reference to a triangular membership function .......... 36
Figure 2.14 The data structure to represent B-Rep ................................................................. 41
Figure 2.15: Interaction between grasp choice and kinematic limits ......................................... 42
Figure 2.16: Requirements of a gripper [59] ............................................................................. 44
Figure 2.17: Gripper classification (a) Pivoting movement gripper, (b) Linear or translational gripper ........................................................................................................ 44
Figure 2.18: (a) The Karlsruhe Dextrous Hand. (b) the Karlsruhe Dextrous Hand II ............... 45
Figure 2.19: Two-fingered and three-fingered modular grippers ............................................... 46
Figure 2.20: Trajectory planning process ................................................................................... 47
Figure 2.21: Reference and body-attached coordinate system ................................................... 48
Figure 2.22: Rotating coordinate systems ................................................................................... 48
Figure 2.23: Various components of a typical industrial system ............................................... 51
Figure 2.24: Basic structure of the software for robot applications ........................................... 52
Figure 3.1: The experimental setup for the intelligent gripping system .................................... 55
Figure 3.2: The assembly of actuator .......................................................................................... 57
Figure 3.3: The 3D shaded actuator ............................................................................................ 57
Figure 3.4: Torque-inertia curve ............................................................................................... 58
Figure 3.5: Mechanics model of lead screw-nut mechanism [81] ........................................... 60
Figure 3.6: The three-fingered gripper assembly ................................................................. 62
Figure 3.7: 3D shaded model of the gripper ......................................................................... 63
Figure 3.8: Kinematic concept diagram of the gripper mechanism .................................... 64
Figure 3.9: Force analysis diagram of slider (a) and crank (b) ............................................ 66
Figure 3.10: The architecture of intelligent gripper system ................................................. 67
Figure 3.11: PC bus interface cards ....................................................................................... 69
Figure 3.12: Wiring diagram for robot vision system ........................................................... 70
Figure 3.13: Setting and calibration of the frame grabber .................................................... 71
Figure 3.14: Setting and calibration of CCD camera ........................................................... 72
Figure 3.15: Wiring diagram of tactile array sensor ............................................................. 73
Figure 3.16: (a) Press on the sensor, (b) 2D touch image, (c) 3D touch image ............... 73
Figure 3.17: The wiring diagram of actuator control ............................................................ 74
Figure 3.18: PMAC PID and NOTCH Servo Filter ............................................................... 75
Figure 3.19: System software components for the PC-based intelligent gripping system ... 78
Figure 3.20: Architecture of 32-bit ODBC MDB access ..................................................... 82
Figure 3.21: Finger configuration of the gripper ................................................................. 83
Figure 3.22: Output image of gripping planning .................................................................. 85
Figure 4.1: Schematic architecture of robot vision system ................................................. 91
Figure 4.2: Typical captured image ...................................................................................... 93
Figure 4.3: Image through smoothing and thresholding ...................................................... 96
Figure 4.4: (a) Discontinuity from A to B ........................................................................... 100
Figure 4.5: Block diagram of interpolation principle .......................................................... 101
Figure 4.6: Flow chart of edge following and edge thinning .............................................. 102
Figure 4.7: Profile of the object after edge linking, edge following and edge thinning .... 103
Figure 4.8: Definition of length L and width W ................................................................. 104
Figure 4.9: Schematic diagram of vector tracing process .................................................... 107
Figure 4.10: Profile reconstruction by means of the closed chain of vectors .................... 107
Figure 4.11: Definition of CFC input variables Len and Prj .............................................. 108
Figure 4.12: Triangle membership functions of Len, Prj, and Node .................................. 109
Figure 4.13: Closed chain of vectors by means of CFC .................................................... 112
Figure 4.14: Object classification ................................................................. 113
Figure 4.15: Calculation of angle between two adjacent vectors ................. 113
Figure 4.16: Profile with four edges ............................................................. 114
Figure 4.17: Profile with six edges ............................................................... 114
Figure 4.18: Profile with two parallel edges ................................................. 115
Figure 4.19: Definition of isosceles-triangle, equilateral-triangle, rectangle, and hexagon... 115
Figure 5.1: Flow chart of inference of grasp mode ........................................ 120
Figure 5.2: Grasp features for cylinder and sphere ....................................... 122
Figure 5.3: Grasp feature for wedges ........................................................... 122
Figure 5.4: Grasp feature for block .............................................................. 123
Figure 5.5: Grasp feature for Parallelepiped ............................................... 124
Figure 5.6: Grasp feature for Polyhedron .................................................... 124
Figure 5.7: Configuration of the gripper while gripping round object ........... 126
Figure 5.8: Configuration of the gripper while gripping wedge with profile of equilateral triangle ................................................................. 126
Figure 5.9: Configuration of the gripper while gripping wedge with profile of isosceles triangle ............................................................................. 127
Figure 5.10: Configuration of the gripper while gripping block object .......... 128
Figure 5.11: Configuration of the gripper while gripping Parallelepiped object ................................................................. 129
Figure 5.12: Configuration of the gripper while gripping Polyhedron object ... 129
Figure 5.13: Computation diagram for gripping planning for two parallel edges 130
Figure 5.14: Grasp model for a polyhedron ................................................ 132
Figure 6.1: S4 robot communication protocols ......................................... 134
Figure 6.2: IRB1400 robot manipulator and its axes .................................... 136
Figure 6.3: Coordinate frame chain ........................................................... 136
Figure 6.4: Various coordinate system relationships in intelligent gripping system 138
Figure 6.5: Structure of RAPID program .................................................. 142
Figure 6.6: Definition of TCP for the three-fingered gripper ....................... 143
Figure 6.7: Digital I/O Communication between the host computer and robot controller ... 145
Figure 6.8: Gripping process control flow chart ....................................... 146
List of Tables

Table 2.1 The elements used in the design of a basic fuzzy inference algorithm .......... 34
Table 2.2 Defuzzification Methods ........................................................................... 38
Table 3.1: P2HNRXC-LDN-PD-00 hybrid stepper motor principal specification ........ 58
Table 4.1: Rules for fuzzy reasoning ......................................................................... 110
Glossary

A

Actuator: A motor or transducer that converts electrical, hydraulic, or pneumatic energy into power for motion or action.

Anthropomorphic Robot: Also known as a jointed-arm robot. A robot with all rotary joints and motions similar to a person’s arm.

Application (Computer): A program which is designed to facilitate the user to perform prescribed tasks.

Articulated Robot: A robot arm which contains at least two consecutive revolute joints acting around parallel axes resembling human arm motion. The work envelop is formed by partial cylinders or spheres.

Artificial Intelligence: The ability of a machine system to perceive anticipated or unanticipated new conditions, decide what actions must be performed under the conditions, and plan the actions accordingly. The main areas of application are expert system and vision.

B

Binary Image: A digitized image in which the brightness of the pixel can have only two different values, such as white and black.

Binarization: A process which converts a grayscale image into binary image.
Cell: A manufacturing unit consisting of two or more work stations or machines, and the material transport mechanisms and storage buffers that interconnect them.

Centre of Gravity: That point in a rigid body where the entire mass of the body could be concentrated and produce the same gravity resultant as for the body itself.

Chain Codes: A set of straight line segments of specified length and direction which are used to represent a boundary. Typically, this representation is established on a rectangular grid using 4- or 8-connectivity.

Classification: A process of grouping objects together into classes (subpopulations) according to their perceived likenesses or similarities.

Closed-Loop Control: The use of a feedback loop to measure and compare actual system performance with desired performance. This allows the robot control to make any necessary adjustment.

Computer Vision: Also known as machine vision. The use of computers or other electronic hardware to acquire, interpret, and process visual information. It involves the use of visual sensors to create an electronic or numerical analog of a visual scene, and computer processing to extract intelligence from this representation.

Configuration (of Gripper): The description and specification of mechanism, including the kinematic and/or structural features, the number of degree of freedom, the joint travel range, and the type of drive for the robot.
**Contour Following:** A feature of robot control program permitting the robot to move along a desired surface that is not defined completely. The robot gripper along with associated sensors is positioned at the beginning of the contour to be tracked. As the gripper is moved along the contour, the sensors feed data back to the control unit to ensure that constant is maintained with the contour.

**Coordinate Transformation:** In robotics, a $4 \times 4$ matrix used to describe the positions and orientations of coordinate frames in space. It is a suitable data structure for the description of the relative position and orientation between objects. Matrix multiplication of the transformations establishes the overall relationship between objects.

**D**

**Degree of Freedom:** The number of independent ways the end effector can move. It is defined by the number of rotational or translational axes through which motion can be obtained. Every variable representing a degree of freedom must be specified if the physical state of the manipulator is to be completely defined.

**E**

**Edge Detection:** An image analysis technique in which information about a scene is obtained without acquiring an entire image. Locations of transition from black to white and white to black are recorded, stored, and connected through a process called connectivity to separate objects in the image into blobs. The blobs can then be analyzed and recognized for their respective features.
**Encoder:** A transducer used to convert linear or rotary position to digital data.

**End Effector:** Also known as end-of-arm tooling or, more simply, hand. The subsystem of an industrial robot system that links the mechanical portion of the robot (manipulator) to the part being handled or worked on, and gives the robot the ability to pick up and transfer parts and/or handle a multitude of differing tools to perform work on parts.

**End-of-Arm Tooling:** A device, commonly made up of four distinct elements, which provide for (1) attachment of the hand or tool to the robot tool mounting plate, (2) power for actuation of tooling motions, (3) mechanical linkages, and (4) sensors integrated into the tooling.

**Feature Extractor:** A program used in image analysis to compute the values of attributes (features) considered by the user to be possibly useful in distinguishing between different shapes of interest.

**Feedback:** The signal or data sent to the control system from a controlled machine or process to denote its response to the command signal.

**Flexibility (Gripper):** The ability of a gripper to conform to parts that have irregular shapes and to adapt to parts that are inaccurately oriented with respect to the gripper.
G

**Grasp Feature:** The geometric attributes (features) which may be used to construct the grasp model in terms of object recognition resultant, such as position and orientation of the object.

**Gripper:** The grasping hand of the robot which manipulates objects and tools to fulfill a given task.

**Grayscale Image:** A digitized image in which the brightness of the pixels can have more than two values which are typically 128 or 256. A grayscale image requires more storage space and more sophisticated image processing than a binary image.

**Gripping Planning:** A capacity which determines where to grasp objects in order to conduct stable grasp and avoid collision during grasping or moving. The grasp configuration is chosen so that objects are stable in the gripper.

H

**Hand:** A fingered gripper sometimes distinguished from a regular gripper by having more than three fingers, and more dexterous finger motions resembling the human hand.

**Heuristic Problem Solving:** In computer logic, the ability to plan and direct actions to steer toward higher-level goals. This is the opposite of algorithmic problem solving.
**Homogeneous Transform:** A $4 \times 4$ matrix which represents the rotation and translation of vectors in the joint coordinate systems. It is used to compute the position and orientation of any coordinate system with respect to any other coordinate system.

**Image Analysis:** The interpretation of data received from an imaging device.

**Imaging:** The analysis of an image to derive the identity, position, orientation, or condition of objects in the scene. Dimensional measurements may also be performed.

**Intelligent Robot:** A robot that can be programmed to execute performance choices contingent on sensory inputs.

**Interface:** A shared boundary which might be a mechanical or electrical connection between two devices; it might be a portion of computer storage accessed by two or more programs; or it might be a device for communication with a human operator.

**Joint:** A rotary or linear articulation or axis of rotational or translational (sliding) motion in a manipulator system.

**Kinematics (Robot):** The study of the mapping of joint coordinates to link coordinates in motion, and inverse mapping of link coordinates to joint coordinates in motion.
**L**

**Linear Interpolation:** A computer function automatically performed in the control that defines the continuum of points in a straight line based on only two taught coordinate positions. All calculated points are automatically inserted between the taught coordinate positions upon playback.

**M**

**Machine Intelligence:** The study of how to make machines learn and reason to make decisions, as do humans.

**Manipulator:** A mechanism, usually consisting of a series of segments, or links, jointed or sliding relative to one another, for grasping and moving objects, usually in several degrees of freedom. A manipulator refers mainly to the mechanical aspect of a robot.

**Mathematical Modeling:** Using mathematics, computers and engineering to describe, simulate, analyse and improve processes and systems.

**Modular Programming:** A software design methodology which requires components to be developed in isolation so as to facilitate the integration of different modules.

**O**

**Orientation:** Also known as positioning. The consistent movement or manipulation of an object into a controlled position and attitude in space.
**P**

**Part Classification:** A coding scheme, typically involving four or more digits, which specifies a discrete product as belonging to a part family according to group technology.

**Path:** A series of positions in space that a robot manipulator or grasped object moves through.

**Pattern Recognition:** A field of artificial intelligence, in which image analysis is used to determine whether a particular object or data set corresponds to one of several alternatives or to none at all. The analysis system is provided in advance with the characteristics of several prototype objects so that it can classify an unknown object by comparing it with each of the different prototype.

**Payload:** The maximum weight that a robot can handle satisfactorily during its normal operations and extensions.

**Pixel:** Also known as photo-element or photosite. This is a digital picture or sensor element. Pixel is short for picture-cell.

**R**

**Reasoning:** A process of applying general rules, equations, relationships, and so on, to an initial collection of data, facts, and so on, to deduce a result or decision.

**Recognition:** A labeling process, that is, is the function of recognition algorithms is to in a scene and to assign a label to that object.
Robot: A robot is a reprogrammable, multifunctional manipulator designed to move material, parts tools, or specialized devices through variable programmed motions for the performance of a variety of tasks.

Robot Calibration (for vision): The act of determining the relative orientation of the camera coordinate system with respect to the robot coordinate system.

Robotics: The science of designing, building, and applying robots.

Sensing: The feedback from the environment of the robot which enables the robot to react to its environment. Sensory inputs may come from a variety of sensor types including proximity switches, force sensors, tactile sensors, and machine vision systems.

Sensor: A device such as a transducer that detects a physical phenomenon and relays information to a control device.

Servo-Actuator (Gripper): An actuator which is equipped with a control system, in which the control computer issues motion commands to the actuators, internal measurement devices measure the motion and signal the results back to the computer. The process continues until the gripper reaches the desired position.
**T**

**Tactile Sensor:** A touch sensor which is capable of detecting contact of touch, force, pattern slip, and movement. It may give information on the local shape, orientation, and feedback forces of a grasped workpiece.

**Teach Pendant:** Also known as teach box. A portable, hand-held programming device connected to the robot controller containing a number of buttons, switches, or programming keys used to direct the controller in positioning the robot and interfacing with auxiliary equipment. It is used for teach pendant programming.

**Thresholding:** A procedure of binarization of an image by segmenting it to black and white regions (represented by ones and zeroes). The gray level of each pixel is compared to a threshold value and then set to 0 or 1 so that binary image analysis can then be performed.

**Tool Centre Point (TCP):** A tool-related reference point that lies along the last wrist axis at a user-specified distance from the wrist.

**Trajectory:** A sub-element of a cycle that defines lesser but integral elements of the cycle. A trajectory is made up of points at which the robot performs or passes through an operation, depending on the programming.

**Translation:** A movement such that all axes remain parallel to what they were (i.e. without rotation).
Vision, 2D: The processing of 2D images by a computer vision system to derive the identity, position, orientation, or condition of objects in the scene.

Vision System: A system interfaced with a robot which locates a part, identifies it, directs the gripper to a suitable grasping position, picks up the part, and brings the part to the work area. A coordinate transformation between the camera and the robot must be carried out to enable proper operation of the system.
Chapter 1 Introduction

At present, industrial robots have found wide applications in a large number of areas, such as assembly, material handling and machine tending, packing, picking, palletizing, gluing and sealing, arc welding, spot welding, painting and coating, foundry applications, and waterjet cutting. Recently, the product life cycles are becoming shorter, and the change of assembly tasks occur frequently. But most of them implement their repetitive tasks using preprogrammed techniques and without the least intelligence. So far as industrial robots are concerned, present-day industrial robots are inflexible and costly to apply. Each robot application is a custom-designed collection of jigs, fixtures, parts presentation mechanisms and special tooling. One of the items which is usually custom-made for each application is the end of arm tooling [1]. However, literature on directly incorporating functionality of intelligent gripping to practical industrial robots is not addressed abundantly.

Recently, with development of flexible automation, with the presence of robots in manufacturing, realization of flexibility and intelligence presents a challenging task for most industrial robots. The future intelligent manufacturing system must be highly adaptable to unanticipated change that implies the ability to learn. The system must exhibit a high degree of autonomy in dealing with change, which implies the ability to deal with significant complexity, and is expected to possess much greater flexibility than it does now. Such systems would be sparse and of a hierarchical nature. Herein intelligent robots are able to satisfy the requirements.
Intelligent robotic operations may be achieved by means of the use of intelligent grippers which accomplish tasks by interacting with their environment in terms of automated part recognition, monitoring of the status of the object being grasped and control of grasping process based on on-line acquisition of sensory data. As is true in humans, vision capabilities endow a robot with a sophisticated sensing mechanism that allows the machines to respond to its environment in an “intelligent” and flexible manner [2].

To endow a gripper mounted on an industrial robot with intelligence, the algorithm for automated part recognition; intelligent algorithm for the gripper control, and on-line learning should be developed on the basis of the investigation into the intelligent gripping.

- Various grippers available are closely related to applications and robots. They are not suitable and applicable to this research. Therefore, design of a gripper suited to serve as a research platform for a specific robot, with which intelligent gripping can be investigated, is fundamental.

- In order to investigate into the intelligent gripper application, a generic framework integrating acquisition of sensory data, part recognition, decision-making and gripper control to achieve intelligent gripping is required. With the framework intelligent gripping may be formed as a hierarchical structure involved in sensing, decision making, gripping planning and gripper control.
• Gripping process is in accordance with the identification of part position and orientation as well as its geometry based on sensory data. For a multi-sensor system, sensory data might be of inconsistency in some cases. An algorithm for automated part recognition on-line is crucial.

• The geometric formation of objects to be grasped varies from object to object. The selection of the gripping surfaces and positions has a great influence on the stability and reliability of gripping process.

1.1 Aim

To construct a multi-sensor based intelligent gripper system working on the platform of an industrial robot, intelligent part recognition, decision making, intelligent gripping planning, gripping state monitoring, and manipulator trajectory planning systems are incorporated and integrated to implement intelligent gripping.

1.2 Objectives

The following objectives were accordingly specified for this project:

• To establish a generic framework which integrates sensory data, part recognition, decision-making, gripping planning, robot trajectory planning, and gripper control to achieve intelligent gripping based on an designed gripper featuring precision position control, grasp status identification and monitoring
• To develop an intelligent algorithm for automated part recognition on-line which includes identification of part position and orientation as well as its geometric shapes based on sensory data

• To develop an algorithm for gripping part intelligently based on part recognition, which means the optimal gripping surfaces and positions of the part, as well as the adjusting of the gripping

• To plan robot trajectory in terms of the kinematics of the manipulator and workcell scene setup, and implement the control scheme through Ethernet

• To develop a control scheme to control the gripper and gripping process intelligently in terms of sensory data, robot and gripping knowledge and information fusion

• To develop a windows-based object oriented software application framework with appropriate user and communication interfaces that integrate the overall system components
1.3 Hypothesis

Gripping intelligence may be achieved through gripper design, automated part recognition, gripping planning, intelligent algorithm for control of the gripper, and on-line learning and decision making based on sensory information or information fusion.

Visual C++ with its rich set of Microsoft Foundation Classes (MFC) and object oriented language features may be utilized to develop an object-oriented software framework to integrate and coordinate sensory data acquisition and processing, monitoring and implementation of all sorts of intelligent algorithms and control scheme. The application framework will further enhance flexibility, modularity, promote user-process interaction and software re-configurability.

Intelligent gripping features on-line part recognition, and gripping planning, gripping status monitoring and adjusting, which may be achieved by exploiting fuzzy set, fuzzy controller, and extracting object features through mathematical models.

1.4 Methodological Justification

In order to accomplish the objectives, the fundamental research issues covered in this project include:

- Digital Image Processing

The use of vision is motivated by the continuing need to increase the flexibility and scope of applications of robotic systems [2]. Robot vision may be defined as the process of
extracting, characterizing, and interpreting information from images of a three-dimensional world. This process also commonly referred to as machine or computer vision, may be subdivided into sensing, preprocessing, segmentation, description, recognition, and interpretation. Digital image processing is pivotal in the robot vision. Although the number of techniques available for digital image processing is significant, only subset of these methods satisfies the requirements of computational speed and low implementation cost. Digital image is processed in spatial-domain. And convolution technique is exploited to implement various image processing.

• **Fuzzy Sets and Fuzzy Logic**

Fuzzy set theory, compared to other mathematical theories, is perhaps the most easily adaptable theory to practice. The main reason is that a fuzzy set has the property of relativity, variability, and inexactness in the definition of its elements. Instead of defining an entity in calculus by assuming that its role is exactly known, we can use fuzzy sets to define the same entity by allowing possible deviations and inexactness in its role. This representation suits well the uncertainties encountered in practical life, which makes fuzzy sets a valuable mathematical tool [3]. In pattern recognition, we may find a better solution to geometric classification by means of fuzzy set theory.

Fuzzy logic is one of the fastest growing technologies in the world since the beginning of the computer era. There are well-established elements of the fuzzy IF-THEN structure, including fuzzy variables, membership functions, fuzzy rules, implication process, and decomposition. Putting these elements together forms a fuzzy inference algorithm. Fuzzy
pattern recognition is a very active branch in the field of pattern recognition, which may result in systems with high computational efficiency, and desired accuracy.

- **Reconfigurable System**

Many emerging applications in communication, computing and consumer electronics industries demand that their functionality stays flexible after the entire system has been manufactured. Such flexibility is required in order to cope with changing user requirements, improvements in system features, changing protocol and data-coding standards, demands to support variety of different user applications, location-specific user services, etc. Such flexibility can also provide novel implementation approaches leading to performance improvements, reduction of the system's cost or reduction of the overall power consumption.

### 1.5 Delimitations

This research will concentrate on the construction of a generic platform framework and intelligent algorithm with which investigation into gripper intelligence can be carried out, and methodology for intelligent gripping is to be developed. The implementation of intelligent gripping in this research project, however, will be limited to an ABB IRB-1400 with a S4 controller. The intelligent gripping planning algorithm will focus on implementation of intelligent gripping with reference to a three-fingered servo gripper developed in this research project. In addition, the object of the research is individual part oriented, and barrier-avoidance is beyond the research.
1.6 Significance of Research

1.6.1 Within the Technikon

It would be an initiating Technikon based research project that would expand the research areas of the Technikon. The fundamental research framework in robotics will be constructed.

A research platform for further investigation into robotics will be set up, on which researches on computer vision, pattern recognition, machine intelligence, etc., are able to be launched in the future. As an awareness of the research on robotics in Technikon context is being created and reinforced, it is certain that the outcome of the project will contribute to the development of the field of Robotics and Mechatronics within Technikon.

1.6.2 General

Flexible automation, usually robot based, has resulted in widespread interest and rapid development in robotics for recent years. Researches on robot intelligence have led to a variety of approaches to the problem-solving systems in artificial intelligence. The industrial robots are playing an increasingly important role in manufacturing industries.

As the majority of present industrial robots perform their tasks using preprogrammed techniques and without the aid of sensory feedback, to meet the requirements of the flexible automation, on some occasions where industrial robots working on preprograms fail to function appropriately, it is important that the industrial robots available be endowed with intelligence.
The project will make a contribution to the research fields, such as robot intelligence, control of intelligent gripper, on-line learning of robot, automated part recognition, etc.

Outside researchers and industry can benefit from the intelligent gripper and methodology developed by the project.

1.7 Organisation of Thesis

Objectives, hypotheses, delimitations, and significance of this research project are introduced in chapter 1. Chapter 2 analyzes the relevant theories, corresponding components, related technology, and up-to-date development in the field of intelligent robotics in terms of literature survey. Chapter 3 describes the overall system setup, hardware architecture, software components, implementation of subsystems, and integration of individual subsystems to form a platform of intelligent gripping platform in detail. Chapter 4 involves the architecture of robot vision system, digital image processing techniques, algorithms of extraction and expression of geometric features by means of fuzzy logic controller, as well as their implementation. Chapter 5 includes model-based gripping planning, in which algorithm of object geometry modeling based on fuzzy set classification is discussed and the approach that gripper configuration is reasoned out is encompassed. Chapter 6 describes robot trajectory planning in terms of its kinematics and intelligent gripping approach, and intelligent gripping control scheme, as well as touch image analysis. Conclusions are made in chapter 7, in which future research and development are put forward.
Appendix A describes the all procedures for this intelligent gripping system with regard to an object. Appendix B contains method and procedures for robot calibration so as to build the transformation relationship between robot coordinate system and image coordinate system. Appendix C contains system configuration files and fuzzy rule database for the CFC (fuzzy continuous controller). Appendix D encompasses the general robot RAPID program generated automatically during the intelligent gripping process. Appendix E gives gripper design drawings.
Chapter 2  Automated Part Recognition and Intelligent Gripping: Relevant Concepts, Framework and Technology

PC-based intelligent gripping systems feature, to a large extent, sensor-based perception and fusion, incorporation of intelligent algorithms into the control systems, as well as integration of individual subsystem modules. A robot that can “see” and “feel” is easier to train in the performance of complex tasks while, at the same time, requires less stringent control mechanisms than preprogrammed machines. A sensory, trainable system is also adaptable to a much larger variety of tasks, thus achieving a degree of universality that ultimately translates into lower production and maintenance costs.

The kernel of an intelligent robotic system consists of a multiple set of decision-making systems which need to

- generate robot trajectory;
- control gripper system equipped with a variety of sensors;
- perform object geometry modeling, object recognition, object geometric features extracting to determine gripper configuration, gripping strategies, real-time gripping parameters (position, orientation, force), and gripping adaptation in terms of sensory data.

The intelligence of a robot is realized by extracting, analyzing, and modeling of sensory data with the view of deriving effective decision-making schemes. Therefore, sensors play a crucial role in robot intelligence. The function of robot sensors may be divided into
two principal categories: internal state and external state. Internal state sensors deal with the detection of variables such as arm joint position, which are used for robot control; while external state sensors cope with the detection of variables, such as vision, range, proximity, touch, force and torque. In the light of perception of sensory data, an intelligent robot system is composed of a vision system which can be defined as the process of: Extracting, characterizing and interpreting information from images of a three dimensional world; range sensing which is utilized to measure the distance from a reference point to objects in the field of operation of the sensor; proximity sensing which yields an estimate of the distance between a sensor and a reflecting object; tactile sensor which is exploited to obtain information in association with the contact between an end-effector, also known as gripper, and objects in the workspace; and force and torque (F/T) sensing which is employed primarily to measure the reaction forces developed at the interface between mechanical assemblies [2]. Groover M.P. [4] divided the uses of sensors in robotics into four basic categories

- safety monitoring;
- interlocks in workcell control;
- part inspection for quality control; and
- determining positions and related information about objects in the robot cell.

This chapter discusses the relevant techniques and fundamentals to PC-based intelligent gripper systems, such as the overall system architecture, image processing, part recognition, gripping planning, robot trajectory planning, and intelligent gripping control strategies.
2.1 Intelligent Robot Overview

A generic intelligent gripper system based on industrial robot is shown in

Figure 0.1. The generic intelligent gripper system architecture is generally made up of five principal modules that perform part recognition, gripping planning, manipulator trajectory planning, gripping status monitoring and tracking and gripper control. The gripping system is capable of responding to the changing scene intelligently by means of decision making systems which usually comprise sensory data acquisition, processing, analyzing, modeling, as well as reasoning out handling schemes. The core of this type of system features machine vision and gripping intelligence.

The intelligent robotic systems may be divided into two categories in terms of the robots on which they work. One is based on the industrial robot while the other is based on the specially designed robot for specific purposes. The former is a relatively closed system, which, as a result, consists of two independent systems, robot system and intelligent gripper system that are under control of a host computer; the latter, which is usually an integrated system, the intelligent gripper is incorporated into the robot system [5]. The system architecture and implementation of intelligent gripping differ from each other to large extent.

From the intelligent robotics point of view, a variety of robot platforms have been proposed and developed based on different robot systems, gripper designs, sensing
methodologies and implementation, as well as different control schemes. Intelligent robot systems can cope with environmental changes and uncertainties automatically and intelligently [6].

As far as an industrial robot is concerned, a human operator usually guides the robot with teach pendant to the desired location. This procedure is an on-line teaching method. With increasingly shorter product life cycle, assembly tasks frequently change. Thus, a more flexible teach method is demanded. That method must enable a robot to rapidly adapt to
variation of circumstances. In order to cope with these demands, modern automation
trends have increasingly placed an emphasis on sensor-guided robots and off-line
programming (OLP).

OLP has been explored and various techniques have been developed so far. Virtual robot
system is a possible solution in facilitating OLP. In this system, vision sensors such as
charge-coupled device (CCD) cameras are employed to detect differences between actual
and desired part locations, generated off-line using pre-modeled virtual robot systems.
The programmed task, in such a case, can be greatly simplified with the aid of interactive
computer graphics, to simulate the effects of the planned motions, without actually
running the robot. Pusan National University [8] developed an automatic off-line
teaching method for teaching a robot task using OLP with the calibration function which
uses vision information. Image information of a workpiece is supplied to an operator
which teaches the desired locations on the image, and hence perform OLP directly. Thus,
teaching time is reduced. The structure of a cell mechanism facilitating OLP is shown in
Figure 2.2.
In the case of automatic assembly and autonomous assembly [9], robots play an important role in the production lines. Machine intelligence, which is able to adapt to the varying environments and accommodate all kinds of system and random errors, may be much more significant [10].

To achieve a flexible robot system, an open architecture control system is capable of integrating manufacturing components into a single platform. Therefore, a particular component can be easily added and/or replaced. Hong et al [11] proposed a modular object-oriented software architecture implemented on a PC to control a robot. The PC-based open robot control (PC-ORC) system is able to reconfigure its control system for various production environments. As a result, it provides a development environment to rapidly build a new robot control system. In addition, PC-ORC allows easy integration and reuse of hardware and software. With the development of high switching microelectronic devices and computer technology, high performance PCs can be employed to accomplish computation-intensive task. PC-based robot control system is able to endow robots with increased flexibility and intelligence.
2.2 Robot Vision System (Image Processing, Model-based Geometry Recognition, Geometric Feature Coding)

The use of vision is motivated by the continuing need to increase the flexibility and scope of robotic applications. Robot vision may be defined as the process of extracting, characterizing, and interpreting information from images within a three-dimensional world on the platform of a robot system. This process, also commonly referred to as machine or computer vision in general applications, may be subdivided into the following six principal areas: (1) sensing, (2) preprocessing, (3) segmentation, (4) description, (5) recognition, and (6) interpretation.

Robot vision structure and general components are shown in Figure 0.3. Part recognition is a pivotal process in the multi-sensor robot system. Recognition is a labeling process. The function of recognition algorithms is to identify each segmented object in a scene and to assign a label to that object. Recognition approaches in use today can be divided into two principal categories namely: decision-theoretic and structural. Decision-theoretic methods are based on quantitative descriptions (e.g. statistical texture) while structural methods rely on symbolic descriptions and their relationships (e.g. sequences of directions in a chain-coded boundary). Recognition is the basis, as
well as the core of computer vision or machine vision.

Figure 0.3: General components in a robot vision system

2.2.1 Digital Image Processing

Digital image processing has grown into a subject in its own right with applications spanning all areas of human endeavour. Typically, a specific captured image is viewed as representative of some underlying ideal image. In the vernacular, often the obtained image is degraded by some form of noise. The noise could have occurred in capture or transformation, or perhaps the physical image itself was degraded prior to capture. The ultimate goal of digital image processing is to provide a consistent and accurate representation of an object that is free from all kinds of interference in the scene, noise, and inconsistent illumination which might be present spatially and temporally. At present, a variety of techniques have been developed by means of different mathematic methods, ranging from convolution to fuzzy logic [12][13][14] to neural networks [15]. Image processing is always prior to component recognition and perception.
2.2.1.1 Image Acquisition

Nowadays, the sources of image data are CCD cameras that act as transducers that take light reflected from objects in a scene and convert the incident light patterns, focused on to an imaging plane, into analogue electrical signals [16]. A CCD camera can convert image to electrical signals at a high speed. When sampled spatially and quantised in amplitude, these signals yield a high S/N ratio digital image.

In addition, laser range finding, infrared and ultrasonic methods may also be used to create the real world image.

As far as a color CCD camera is concerned, the captured image, which is made of a pixel array \( I_k(x, y) \) (where \( k = 1, 2, 3; x = 1, 2, ..., m; y = 1, 2, ..., n \)) which size is \( m \times n \), contains the color information and intensity of three color channels. This color pixel array can be represented as

\[
I_k(x, y) = \begin{bmatrix}
I(1,1) & I(1,2) & \ldots & I(1,n) \\
I(2,1) & I(2,2) & \ldots & I(2,n) \\
\vdots & \vdots & \ddots & \vdots \\
I(m,1) & I(m,2) & \ldots & I(m,n)
\end{bmatrix} \quad (k = 1, 2, 3) \quad (0-1)
\]

The representation of pixel varies from the purposes of image processing. RGB (red, green, and blue) method, HSB (hue, saturation, and brightness) method, and CMYK (cyan, magenta, yellow, and black) method are commonly used. In the field of machine vision, RGB method is generally employed. In this case, each pixel is represented by three channels which denote red, green, and blue intensity respectively.
2.2.1.2 Illumination Techniques

In a vision system the illumination of a scene has a tremendous influence on the captured image and the complexity of vision algorithm [17][18]. Well-designed and consistent illumination, which is free from low-contrast image, specula reflection, shadows, and extraneous details, is crucial to the robustness of an image processing system. Four of the principal schemes used for illuminating a robot workspace are shown in Figure 2.4.

- **Diffuse-lighting**
  - The diffuse lighting approach can be exploited for objects characterized by smooth, regular surfaces. This lighting scheme is generally employed in applications where surface characteristics are important.

- **Backlighting**
  - Backlighting produces a black and white (binary) image. This technique is ideally suited for applications in which silhouettes of objects are sufficient for recognition or other measurements.

- **Structuring-lighting**
  - Structuring lighting consists of projecting points, stripes, or grids onto the workspace. This lighting technique has two important advantages. First, it establishes a known light pattern on the workspace, and disturbances of this pattern indicate the presence of an object, thus simplifying the object detection problem. Second, by analyzing the way in which the light pattern is distorted, it is possible to gain insight into the 3D approach.

- **Directional-lighting**
  - Directional lighting is useful primarily for inspection of object surfaces. Using a highly directed light beam and measuring the amount of scattered light beyond defects on the surface, such as pits and scratches.
2.2.1.3 Image Pre-processing

Plenty of image preprocessing techniques are available in the field of robot vision. The method used for image preprocessing range from spatial-domain and frequency-domain. Only a subset of them is suited for real-time image processing if the processing speed plays a predominant role in this process.

Convolution technique is one of the spatial-domain techniques used most frequently (also referred to as templates, windows, or filters). The desired image \( f(x, y) \) can be obtained by convoluting the original image \( i(x, y) \) with a convolution mask \( h(x, y) \).

\[
f(x, y) = h(x, y) \ast i(x, y) \tag{0-2}
\]

In robot vision system, the convolution masks are usually a 3×3, 5×5 or 7×7 matrices. For the sake of computational speed a 3×3 matrix is widely utilized in real-time systems. A typical convolution mask is given as

\[
H_{3\times3} = \begin{bmatrix}
    h_{11} & h_{12} & h_{13} \\
    h_{21} & h_{22} & h_{23} \\
    h_{31} & h_{32} & h_{33}
\end{bmatrix} \tag{0-3}
\]

The implementation of \( H_{3\times3} \) convolution can be defined as

\[
f(x, y) = \sum_{j=1}^{3} \sum_{k=1}^{3} i(x + j - 2, y + k - 2) \cdot h(j, k) \tag{0-4}
\]

The commonly used preprocessing techniques, which are also employed in this project, are the following

- **Smoothing**
Smoothing operations are used for reducing noise and other spurious effects that may be present in an image as a result of sampling, quantization, transmission, or disturbances in the environment during image acquisition. The following convolution masks are used:

- **High Pass Filtering**

  High pass filtering is utilized to sharpen images that are out of focus or fuzzy. Its convolution mask is [19]:

  \[
  H_{\text{High-pass}} = \begin{bmatrix}
  -1 & -1 & -1 \\
  -1 & 9 & -1 \\
  -1 & -1 & -1 
  \end{bmatrix} \quad (0-5)
  \]

- **Median Filtering**

  Median filtering ranks the current set of nine pixel intensities in order of magnitude and places the median intensity value into the destination image at the central point. The whole image is processed in turn by sliding the window over the entire image.

- **Low Pass Filtering**

  Low pass filtering is exploited to smooth out a sharp image. Its convolution mask is given as:

  \[
  H_{\text{Low-pass}} = \begin{bmatrix}
  1 & 1 & 1 \\
  1 & 2 & 1 \\
  1 & 1 & 1 
  \end{bmatrix} \quad (0-6)
  \]

- **Noise Cleaning**
Noise cleaning is employed to remove random noise spikes on the captured image. Its convolution mask is given as:

\[
H_{\text{Noise-cleaning}} = \begin{bmatrix}
1 & 2 & 1 \\
2 & 4 & 2 \\
1 & 2 & 1
\end{bmatrix}
\]  \hspace{1cm} (0-7)

➢ **Averaging**

Averaging can be used to remove random noise spikes and clean edge features in the image. Its convolution mask is given as:

\[
H_{\text{Averaging}} = \begin{bmatrix}
1 & 1 & 1 \\
1 & 0 & 1 \\
1 & 1 & 1
\end{bmatrix}
\]  \hspace{1cm} (0-8)

• **Thresholding**

Digital image thresholding is a crucial process in robot vision system, which is used to binarize the captured image. To separate and extract the object from the background in terms of an image array \( f(x, y) \), a threshold of \( T \) is normally utilized. The thresholding technique is not limited to a fixed value \( T \). Mutillevel thresholding, thresholding techniques developed by using neural network [20] and image fuzziness [21] can also be employed on some occasions. A thresholded image can be acquired by

\[
g(x, y) = \begin{cases} 
1 & \text{if } f(x, y) > T \\
0 & \text{otherwise}
\end{cases}
\]  \hspace{1cm} (0-9)

In the case of dark objects on a light background, thresholding takes the selected grayscale value \( T \) and compares each pixel intensity in the image. If the intensity at pixel
$f(x, y) < T$ that pixel is replaced by a logic 0 value. If the intensity $f(x, y) > T$ that pixel is replaced by a logic 1 value. A typical image intensity histogram with two peaks is shown in Figure 2.5.

![Intensity Histogram](image)

**Figure 0.5: Typical two peaks intensity histogram [22]**

The binarized image after thresholding is illustrated in Figure 2.6.

![Thresholding Image](image)

**Figure 0.6: Thresholding to a gray-level image**

In general, thresholding falls into two categories, which are manual thresholding and adaptive thresholding. Adaptive thresholding takes a histogram of all the pixel intensities in the images, detects the pixel intensity most frequent in the image and follows the
histogram curve down to identify the minimum. An adaptive thresholding is capable of figuring out an optimal thresholding value.

**Contour Detection**

Contour detection plays a central role in robot vision. Using the information from the contours means a considerable reduction in the volume of data to be processed in image analysis. In addition, using the contours obtained from the image is their relative stability under fluctuations in the lighting of the scene. The standard approaches to contour detection are implicitly based on a very simple model in which the image is regarded as ideally composed of essentially constant region separated by step edges [23]. The classical approach [24][25][26][27] to contour detection makes use of digital (finite-difference) versions of standard isotropic derivative operators, such as the gradient or *Laplacian.*

The first derivative of a contour modeled is zero in all region of constant intensity. The second derivative is zero in all locations, except at the onset and termination of an intensity transition.

➢ **Laplacian Edge Detection**

The *Laplacian* is a scalar second derivative operator for functions of two dimensions, given by:

\[
\nabla f(x, y) = \frac{\partial^2}{\partial x^2} f(x, y) + \frac{\partial^2}{\partial y^2} f(x, y)
\]  
(0-10)

The digital *Laplacian* at point \((x, y)\) can be defined as:
Gradient Edge Detection

The gradient of an image \( f(x, y) \) at location \( (x, y) \) is defined as the two-dimensional vector:

\[
G[f(x, y)] = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}
\]  

(0-12)

It’s common practice to approximate the gradient by absolute values:

\[
|G(x, y)| = \sqrt{G_x^2 + G_y^2} \approx |G_x| + |G_y|
\]  

(0-13)

For digital image, the component of the gradient vector in the \( x \) direction is given as:

\[
G_x = [f(x+1, y-1) + 2f(x+1, y) + f(x+1, y+1)] - [f(x-1, y-1) + 2f(x-1, y)+ f(x-1, y+1)]
\]  

(0-14)

\[
G_y = [f(x-1, y+1) + 2f(x, y+1) + f(x+1, y+1)] - [f(x-1, y-1) + 2f(x, y-1)+ f(x+1, y-1)]
\]  

(0-15)

The result of contour detection to an Allen key is shown in Figure 2.7:

![Contour Detection](image)

Figure 0.7: The result of contour detection by applying Laplacian operator

- Edge Linking
In practice, the set of pixels generated by contour detection seldom characterizes a boundary completely because of noise, breaks in the boundary due to nonuniform illumination and other effects that introduce spurious intensity discontinuities. Thus contour detection is typically followed by linking other boundary detection procedures designed to assemble edge pixels into a meaningful set of object boundaries.

Local analysis, one of the simplest approaches for linking edge points, is used to analyze the characteristics of pixels in a small neighbourhood about every point in an image that has undergone a contour-detection process. All points that are similar are linked, thus forming a boundary of pixels that share some common properties. Global analysis via the Hough transform and global analysis via Graph-Theoretic techniques are also utilized for this purpose [22].

- *Edge Following*

The objective of edge following is to link together the chain of pixels extracted by edge operators using local pixel neighbourhood searching. Chain code is a practical and widely used edge following method. The chain of vectors is acquired by tracking the interior profile of the symbol under investigation. Chain codes are used to represent a boundary as a set of straight line segments of specified length and direction. Typically, this representation is established on a rectangular grid using 4- or 8-connectivity as shown in Figure 2.8. In general, *update, minimum chain, and noise reduction* are employed to improve the chain to extract the relevant features [28].
2.2.2 Model-based Geometry Recognition

In the application of intelligent robot gripper system, gripper configuration, gripping planning and gripping process control (such as gripping force, real-time adjustment of gripping configuration, etc.) rely on the extraction of geometry features of objects to a large extent. Features of geometry are physically distinguishable regions and invariant on parts. Geometric feature recognition is viewed as the kernel of the robot vision system.

Typically three types of knowledge are needed with respect to geometric features: Feature parameters (size of feature), feature relationships (distance and angle between two features) and feature interactions (recursion between two features) [29]. Features are the aggregation of geometric entities and include attributes such as dimensions, types, orientations etc. They are defined in terms of high-level terms such as height, width and radius. The current approach to obtaining features is through feature recognition. Heuristic analysis and reasoning based on various mathematical modeling algorithms dominate the area. Plenty of feature extraction schemes have been developed, whereas, computationally efficient algorithms are still being explored.
CSG and B-Rep are the two commonly used approaches to representing solid models. Roy et al [30] proposed a feature-based representation scheme based on the hybrid CSG/B-Rep data structure, which exploits the advantages of both CSG and B-Rep. Chen et al [31] proposed a framework for feature based part modeling to provide high-level part models to support geometric reasoning. Feature types used in defining mechanical products are form features that are groups of geometric entities. Primary features and sub-features are used to represent the shape of the part, precision features and material features [32]. In the boundary representation an object is described by using the following information:

- A finite set of vertices, usually specified by Cartesian coordinates in Euclidian space, an ordered set of incident faces and vertex type
- A set of edges, where each edge is specified by a pair of incident faces, the enclosing angle object and edge type
- A set of faces, where each face is specified by its face equation, a normal vector pointed to the "outside" of the object, geometric classification of the face and tool approach information

2.2.2.1 Geometric Feature Extraction and Recognition

A feature is also regarded as a geometric form or entity that is used in reasoning in one or more design or manufacturing activities (i.e. fit, function, manufacturability evaluation, analysis interfacing, tool and die design, inspectability, and serviceability) [33]. The dimension of each feature, their relative positioning and spatial relationships are extracted
and organized into a feature graph, which is then transferred to a feature instantiation and model reconstruction environment for feature based model construction. Nitchke et al [34] summarized four major techniques that have been used to perform feature recognition, which are tree structure recognition, pattern recognition, production rules and graph/grammar matching. The shape of a feature is defined parametrically in terms of a set of dimensions. Primary features are used to form the major shape of a part, while secondary features are used to modify the shape of the part. Hierarchical structure of features can be described as shown in Figure 2.9.

![Hierarchical structure of features](image)

**Figure 0.9: Hierarchical structure of features**

In the field of knowledge-based intelligent robot handling systems, the creation of data structure and the construction and representation of B-Rep model are key issues in the implementation of feature extraction and recognition. A suitable representation scheme must, therefore, be used to describe the data and the model. A representation is desirable if it is unambiguous, unique, not sensitive and convenient to access [35]. Various
schemes, such as boundary-based, surface-based and volumetric-based representations have been proposed or under investigation [36]. A general paradigm in knowledge-based feature extraction and recognition is shown in Figure 2.10

![Figure 2.10: A general paradigm in knowledge-based feature recognition](image)

At present, part recognition algorithms have been extensively investigated. So far as recognition algorithms are concerned, they must be powerful enough to uniquely identify the object. Practically and computationally efficient algorithms for industrial applications still need further investigating. Object recognition techniques used in industry today may be classified into two major categories: Template-matching techniques [37] and structural techniques [38][39]. Template-matching is to match model template with a stored pattern feature set defined as a model template. Plenty of moment invariants techniques are exploited in vision systems [40][41][42]. Structural techniques, also known as syntactic pattern recognition, consider relationships between features or edges of an object. At present, recognition techniques based on neural network and fuzzy logic are being investigated widely. Fuzzy set theory has been successfully utilized in geometry classification system.
2.2.3 Foundations of Fuzzy Feature Extraction and Recognition

In recent years, Fuzzy Logic (FL) has been applied to develop new image processing and part recognition algorithms. Certain uncertainties and non-linearities always exist in feature extraction. The reasons for the uncertainty and non-linearity result from not only the physical process but also changing conditions, such as inconsistent illumination, shallow, and occlusion of objects, as well as spatially varying imaging process. As a result, it may be difficult to model the process and extract features by means of formulae. Fuzzy inference may be more applicable on these occasions. There are well-established elements of the fuzzy IF-THEN structure, including fuzzy variables, membership functions, fuzzy rules, implication process, and decomposition. Putting these elements together forms a fuzzy inference algorithm. The manner in which this algorithmic flow is implemented is also well established in the literature and in industrial applications [43].

As a matter of fact, most of the commercial and industrial applications to date are based on the fuzzy IF-THEN structure because of its expressive power and simplicity. As a striking technique in the field of artificial intelligence, FL has found wide use in heuristic exploration and computationally intensive recursion in real-time applications. FL is a convenient way to map an input space to an output space. FL is flexible, tolerant of imprecise data. Besides, FL, which is based on natural language, can model nonlinear functions of arbitrary complexity, and can be built up on top of the expertise of experts.

2.2.3.1 Fuzzy Inference System Design and Its Elements

The essential part of fuzzy system design is the application of fuzzy sets and fuzzy logic to a solution, or to a method of solution. The design challenge is to translate the
knowledge (natural language, numerical data, or closed-form mathematical formula) into fuzzy IF-THEN form. Expertise articulated in natural language is readily compatible with fuzzy IF-THEN rules. Fuzzy inference is the actual process of mapping from a given input to an output using fuzzy logic. Schematic fuzzy system architecture is illustrated as shown in Figure 2.11.

Figure 0.11: Basic architecture of a fuzzy system

Fuzzy system design in the form of IF-THEN rules consists of deciding what form the design elements should take. When these options in Table 2.1 are determined, the fuzzy system design is considered complete.
Table 0.1 The elements used in the design of a basic fuzzy inference algorithm

<table>
<thead>
<tr>
<th>Category</th>
<th>Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Antecedents</td>
<td>Universe of discourse, Name convention, Membership function, Threshold,</td>
</tr>
<tr>
<td></td>
<td>Linguistic hedge, Linguistic input libraries, Input fuzzifier</td>
</tr>
<tr>
<td>Consequents</td>
<td>Universe of discourse, Name convention, Membership function, Threshold,</td>
</tr>
<tr>
<td></td>
<td>Normalization, Output Processor</td>
</tr>
<tr>
<td>Rules</td>
<td>Logic operators, Rule formation strategy, Implication operators, Aggregation</td>
</tr>
<tr>
<td></td>
<td>operators, Defuzzification, Importance weights</td>
</tr>
</tbody>
</table>

- **Fuzzify Inputs**

The first step is to take the inputs and determine the degree to which they belong to each of the appropriate fuzzy sets via membership functions. The input is always a crisp numerical value limited to the universe of discourse of the input variable and the output is a fuzzy degree of membership. Fuzzification means adding uncertainty by design to crisp...
sets or to sets that are already fuzzy. The triangular membership function, shown in Figure 2.12, is most commonly used to fuzzify the input values.

**Figure 0.12: Triangular membership function**

PL, PS, Z, NS, and NL in Figure 2.12 stand for positive large, positive small, zero, negative small, and negative large respectively. For fuzzy positive large, its membership function can be expressed as:

\[
\mu_{PL} = \begin{cases} 
\frac{x}{x_1} & 0 \leq x < x_1 \\
\frac{x_2 - x}{x_2 - x_1} & x_1 \leq x < x_2 \\
0 & \text{otherwise}
\end{cases}
\]  

(0-16)

In general, if finite singletons are encompassed in the universe of discourse, a fuzzy set can be expressed as:

\[
A = \bigcup_{i=1}^{n} \mu_{A_i}(x)/x_i
\]  

(0-17)

The result of fuzzification can be illustrated in Figure 2.13 [44].
Apply fuzzy operator

Once the inputs have been fuzzified, we know the degree to which each part of the antecedent has been satisfied for each rule. If the antecedent of a given rule has more than one part, the fuzzy operator is applied to obtain one number that represents the result of the antecedent for that rule. This number will be then applied to the output function. The input to the fuzzy operator is two or more membership values from fuzzified input variables. The output is a single truth value. The intersection operation (the AND operation) or the union operation (the OR operation) is usually utilized. The intersection operation is defined as [45]

$$\mu_{A \cap B}(x) = \mu_A(x) \land \mu_B(x) = \min\{\mu_A(x), \mu_B(x)\}, \ \forall x \in X \quad (0-18)$$

The union operation is defined as

$$\mu_{A \cup B}(x) = \mu_A(x) \lor \mu_B(x) = \max\{\mu_A(x), \mu_B(x)\}, \ \forall x \in X \quad (0-19)$$

Apply implication method

We must take care of the rule’s weight, before applying the implication method. Every rule has a weight, which is applied to the number given by the antecedent. The implication method is defined as the shaping of the consequent (a fuzzy set) based on the
antecedent (a single number). The input for the implication process is a single number given by the antecedent, and the output is a fuzzy set. For a fuzzy proposition,

\[
If \ x \ is \ A \ then \ y \ is \ B
\]

by assuming \( P: x \ is \ A \), \( Q: y \ is \ B \), we can simplify it as

\[
If \ P \ is \ true \ then \ Q \ is \ true
\]

The membership function of fuzzy implication can be defined as

\[
\mu_{a \rightarrow b}(x, y) = [\mu_a(x) \land \mu_B(y)] \lor [1 - \mu_a(x)]
\] (0-20)

or it can be expressed by relation matrix \( R_{A \rightarrow B} \)

\[
R_{A \rightarrow B} = (A \times B) \cup (A \times E)
\] (0-21)

Where the symbol \( \times \) denotes Cartesian product, set \( E \) contains all elements of the above fuzzy set \( Y \).

Assume the IF-THEN rule is given as: If \( x \) is \( A \), \( y \) is \( B \), then \( z \) is \( C \), once the relation matrix \( R \) is known, the consequent can be calculated as

\[
C' = (A \times B') \odot R
\] (0-22)

Where \( A' \) and \( B' \) are the antecedent expressed by the vectors and \( C' \) is the consequent.

The symbol \( \odot \) is composition operation, which is defined as

\[
\mu_{R_1 \odot R_2}(x, z) = \lor_{y \in Y} [\mu_{R_1}(x, y) \land \mu_{R_2}(y, z)], \ \forall (x, z) \in X \times Z
\] (0-23)

- Aggregate all outputs
Aggregation is when we unify the outputs of each rule by joining the parallel threads. The input of the aggregation process is the list of truncated output functions returned by the implication process for each rule. The output of the aggregation process is one fuzzy set for each output variable.

\[
Result\ of\ Aggregation = \sum_{i=1}^{n} \mu_i(x) \tag{0-24}
\]

- **Defuzzify**

The input for Defuzzification process is a fuzzy set (the aggregate output fuzzy set). The final output for each variable is generally a single crisp number. So, given a fuzzy set that encompasses a range of output values, we need to return one number, thereby moving from a fuzzy set to a crisp output. Table 0.2 illustrates the two basic mechanisms for Defuzzification: Centroid and maxima.

**Table 0.2 Defuzzification Methods**

<table>
<thead>
<tr>
<th>Centroid Methods</th>
<th>Maxima Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Centre of Gravity</td>
<td>Mean of Maximum</td>
</tr>
<tr>
<td>Centre of Weight</td>
<td>Maximum Possibility</td>
</tr>
<tr>
<td>Centre of Largest Area</td>
<td>Left-right Maxima</td>
</tr>
<tr>
<td>Centre of Mass of Highest Intersected Region</td>
<td></td>
</tr>
</tbody>
</table>
A practical and approximate technique to calculate centre of gravity is:

\[ x' = \frac{\sum_{i=1}^{n} x_i \mu_i(x)}{\sum_{i=1}^{n} \mu_i(x)} \]  \hspace{1cm} (0-25)

### 2.2.4 Fuzzy Set Classification

Fuzzy set classification (or identification) use fuzzy membership rules to classify images.

**Membership rule** is described as [46]:

Assume fuzzy subset \( A_i \subset S, i=1,2,…,n \), and \( x \in S \). If \( A_i(x) = \max \{ A_1(x), A_2(x), \ldots, A_n(x) \} \), then \( x \) is regarded as a member of \( A_i \).

The technique that is utilized to create membership function is the key to fuzzy set classification. As far as the triangle classification is concerned, triangles can be classified as isosceles triangle, equilateral triangle, right-angle triangle and normal triangle. We can use fuzzy set \( I, E, R \), and \( IR \) to describe the above triangles respectively.

Assume the universe of discourse of triangles is \( S=\{(A,B,C)|A+B+C=180^\circ, A\geq B \geq C \geq 0\} \), A, B, C are three angles of a triangle. Therefore, the MBFs of \( I, E, R \), and \( O \) can be defined as

\[
F_I(A, B, C) = 1 - \frac{1}{60} \cdot \min \{|A-B|, |B-C|, |C-A|\} \]  \hspace{1cm} (0-26)

\[
F_E(A, B, C) = 1 - \frac{1}{180} \cdot \min \{|A-B|, |B-C|, |C-A|\} \]  \hspace{1cm} (0-27)

\[
F_R(A, B, C) = 1 - \frac{1}{90} \cdot \min \{|A-90|, |B-90|, |C-90|\} \]  \hspace{1cm} (0-28)

\[
F_O(A, B, C) = F_I \cup F_E \cup F_R = F_I \cap F_E \cap F_R = 1 - F_{I \cup E \cup R} = 1 - \max \{ F_I, F_E, F_R \} = \min \{ 1 - F_I, 1 - F_E, 1 - F_R \} \]  \hspace{1cm} (0-29)
Any input \((A, B, C)\) can be classified with reference to the above \textit{membership rule}.

2.3 \textbf{Intelligent Gripping}

An automatic gripping planning system must be capable of reasoning about the shape of objects within the workspace [47][48]. All possible faces must be reasoned out based on the boundary model extracted from the part recognition module. A gripping model is constructed in terms of those faces. Mating faces are identified and optimal mating faces are reasoned out accordingly. Efficient geometric feature coding plays a very important role in intelligent gripping system. Heuristic approaches are effective and efficient to reason out the optimal faces to be grasped. An intelligent gripping process is usually accomplished with the involvement of sensors, such as F/T sensor and tactile sensors so that the grasping status is kept in a stable condition.

Nnaji and Liu [49] proposed a boundary description method (polyhedral representation), which is used for automatic robotic assembly. The boundary entities are defined as \textit{vertex, edge, loop, plane, and face}. The data structure is shown in Figure 2.15.

Generally speaking, there are two classical approaches. The first method has no pre-stored information about the object and obtains the relevant data from various sensors. A large amount of computation has to be performed in such a method. The second method works with a model of the object and analyzes the entire model for choosing a suitable grasp. This method allows one to know the features of the object exactly and do a lot of the complicated geometrical computations off-line. But the analysis has to be restricted to modeled environments [50]. The effective and stable grasp is dependent on enormous
factors, such as face pairs to be grasped, centre of gravity, inertia moments, etc. Principal issues on the grasp are involved in stability of grasping, which can be achieved by maintaining the force equilibrium at contact points. Nguyen [51] proposed a fast and simple algorithm, which solved the problem of how to construct the independent regions of contact for grasps on polygonal and polyhedral objects. It consists of a general framework for analyzing the dynamic stability of the grasped object with multiple pure rolling contacts, where the fingertips are modeled as elastic structures and the object as a rigid body [52]. Analysis on grasping a rectangle and a round object with a three-fingered gripper was proposed, and relevant equilibrium contact forces are derived respectively.

![Figure 0.14 The data structure to represent B-Rep](image)

For multi-fingered grippers and dexterous hands a method for stably grasping 2 dimensional polygonal objects, which gave a solution to basic constraints on object vertex angles for feasible grasping with two fingers is presented by Fearing [53]. Spatial stability and contact grasp stability must be maintained in the whole grasping process,
which can be achieved on the basis of distinguishing these two types of grasp stabilities [54]. Also the centre of the positions of the points of contact as it evolves in time on the surface of a grasped object in the absence of any external force or active feedback was derived.

In robot handling application the pick-and-place operations involve the choice of grasp and the choice of trajectory to reach the grasp. It must be implemented by satisfying the pick-and-place constraints [55]. Interaction between grasp choice and kinematic limits must be taken into consideration, as shown in Figure 2.15.

![Figure 0.15: Interaction between grasp choice and kinematic limits](image)

In most cases, one more feasible mating edge pairs exist. Therefore, optimal grasping is another issue of intelligent gripping. Li and Sastry [56] proposed three quality measures: A minmax measure, a volumetric measure, and a task-oriented measure. For evaluating a grasp with multifingered robot hand screw theory and elementary differential geometry are used. Making sensing and acting techniques to cooperate in order to achieve a given manipulation task in a partially structured environment, is one of the major issues in robotics. A combination of partial geometric models using vision data is discussed [57] in
the context of automatic grasping using a guided decisional process. To guide the grasping movements, three processing phases respectively are applied: Selecting a viewpoint to avoid occlusions, modeling the local environment of the object to be grasped and determining the grasping parameters.

The intelligent gripping aims at

- achieving a collision-free, kinematically feasible trajectory;
- stable grasp with no twisting or slipping relative to the gripper while picking up an object; and
- no part of robot in collision with any obstacle at either the object’s pickup or putdown position.

### 2.4 Intelligent Robot Gripper (Multi-sensor Based Gripping Applications, Gripper Structure & Design)

A robot arm by itself can serve no purpose until a load or a tool is suspended from or attached to it. Devices connected between the robot wrist and the load for grasping the object to be handled is known as grippers. The robot gripper, also known as end-of-arm tooling, or end-effector, becomes a bridge between the computer-controlled arm and the world around it. The design of the gripper should reflect this role, matching the capabilities of the robot to the requirements of the task. The ideal gripper design should be synthesized from independent solutions to the three considerations shown in Figure 2.16. Grippers essentially replace the human hand. If the gripping abilities of a mechanical five-fingered “hand” are denoted as 100%, a four-fingered hand has 99% of
its ability, a three-fingered hand about 90%, and a two-fingered hand 40% [58]. The main function of a gripper is to grasp and release workpieces during the material transfer route.

Figure 0.16: Requirements of a gripper [59]

Various ways of classifying mechanical grippers and their actuating mechanisms have been put forward. One method is according to the type of finger movement used by the gripper. In this classification, the grippers can actuate the opening and closing of the finger by pivoting movement, or linear or translational movement, as shown in Figure 2.17.

Figure 0.17: Gripper classification (a) pivoting movement gripper, (b) linear or translational gripper
To improve the grasp flexibility, different types of dexterous grippers have been explored and developed for specific and complicated applications, as shown in Figure 2.18.

![Figure 0.18: (a) The Karlsruhe Dextrous Hand (b) The Karlsruhe Dextrous Hand II](image)

Recently, the manufacturing community has begun to use readily available, off-the-shelf components to build up automation systems in a modular way. This catalog based approach to machine design has many benefits including: ease of replacement of defective and worn-out parts, cost savings, shorter design times, more rapid implementation of the machine and leveraging the expertise of component builders [60][61]. Many gripper manufacturers have also taken this approach. Modular gripper design is shown in Figure 2.19.
Advanced robot systems require sensory information to enable them to make decisions and to carry out actions in a versatile and autonomous way. Intelligent grippers, such as hand-eye systems, are always equipped with various sensors, among which the CCD camera, proximity sensor, F/T sensor and the tactile sensor are exploited in most cases [62][63]. Proximity sensors, measuring the distance and orientation of an object relative to the gripper. It is necessary to bridge the uncertainty gap between the gross proximity-estimation of a vision system and the direct contact required for tactile sensing. On the other hand, humans make considerable use of information derived through touch. An emerging domain of robot sensing is tactile sensing. For reliable and robust gripping a tactile sensor is indispensable. Tactile image [64] can be employed to adjust contact state and contact force and prevent any possible slipping in the grasp process.

For multi-sensor gripper systems, sensor fusion [65][66] has being investigated extensively for the purpose of better extraction, perception and integration of different types of sensory data such that more reliable and multilateral recognition can be achieved.
2.5 Robot Trajectory Planning

In an intelligent robot handling system, dynamic and partially unpredictable environment is always expected. Hence, robot motion planning must be on-line. The planner receives a continuous flow of information about occurring events and generates new plans, while previously planned motions are being executed. Appropriate robot trajectories should be generated by on-line planners, transferring them to their respective goals, while avoiding collision with obstacles [67]. The schematic of trajectory planning process is shown in Figure 2.20.

Figure 0.20: Trajectory planning process

For robot handing system, the relationship between the robot coordinate system (tool coordinate system) and object coordinate system must be created. Transformation matrices are usually employed for this purpose.
2.5.1 Rotation Matrices

A 3×3 rotation matrix can be defined as a transformation matrix which operates on a position vector in a three-dimensional Euclidean space and maps its coordinates expressed in a rotated coordinate system $OUVV$ (body-attached frame) to a reference coordinate system $OXYZ$, as shown in Figure 2.21.

![Figure 0.21: Reference and body-attached coordinate system](image)

Figure 0.21: Reference and body-attached coordinate system

Figure 2.22 shows the $OUVV$ coordinate system rotated at an $\alpha$ angle about the $OX$ axis, then rotated an $\phi$ angle about the $OY$ axis, and then rotated an $\theta$ angle about the $OY$. The rotation matrices can be represented as the followings respectively.

![Figure 0.22: Rotating coordinate systems](image)

Figure 0.22: Rotating coordinate systems

The rotation matrices can be represented as the following respectively:
\[
R_{x,\alpha} = \begin{bmatrix}
1 & 0 & 0 \\
0 & \cos \alpha & -\sin \alpha \\
0 & \sin \alpha & \cos \alpha \\
\end{bmatrix} \tag{0-30}
\]

\[
R_{y,\phi} = \begin{bmatrix}
\cos \phi & 0 & \sin \phi \\
0 & 1 & 0 \\
-\sin \phi & 0 & \cos \phi \\
\end{bmatrix} \tag{0-31}
\]

\[
R_{z,\theta} = \begin{bmatrix}
\cos \theta & -\sin \theta & 0 \\
\sin \theta & \cos \theta & 0 \\
0 & 0 & 1 \\
\end{bmatrix} \tag{0-32}
\]

### 2.5.2 Composite Rotation Matrix

Basic rotation matrices can be multiplied together to represent a sequence of finite rotations about the principle axes of the OXYZ coordinate system. Since matrix multiplications do not commute, the order or sequence of performing rotations is important.

The rotation matrix representing a rotation of \(\alpha\) angle about the \(OX\) axis \((yaw)\) followed by a rotation of \(\theta\) angle about the \(OZ\) \((roll)\) followed by a rotation of \(\phi\) angle about \(OY\) \((pitch)\) axis is given by the resultant rotation matrix as:

\[
R = R_{y,\phi} R_{z,\theta} R_{x,\alpha} = \begin{bmatrix}
C\phi C\theta & S\phi S\alpha - C\phi S\theta C\alpha & C\phi S\theta S\alpha + S\phi C\alpha \\
S\theta & C\theta C\alpha & -C\theta S\alpha \\
-S\phi C\theta & S\phi S\theta C\alpha + C\phi S\alpha & C\phi C\alpha - S\phi S\theta S\alpha \\
\end{bmatrix} \tag{0-33}
\]

The rotation matrix representing a rotation of \(\phi\) angle about \(OY\) axis followed by a rotation of \(\theta\) angle about the \(OZ\) axis followed by a rotation of \(\alpha\) angle about the \(OX\), the resultant rotation matrix representing these rotations is:
2.5.3 Vision-based Path Planning

A very important step towards intelligent gripping system based on an industrial robot is
developing ways to generate motion plans for achieving certain goals while satisfying
environmental constraints with collision-free and optimal path. Classical motion planning
is defined on a configuration space which is assumed to be known, implying the complete
knowledge of both the robot kinematics as well as knowledge of the obstacles in the
configuration space. On the other hand, vision-based, or more general, sensor-based path
planning provides a more practical approach to robot control. To best utilize the sensor
feedback, a robot motion planner should incorporate constraints from the sensor system
as well as criteria for optimizing the quality of the sensor feedback [68][69].

2.6 PC-based Intelligent Robot Gripping Architecture

The PC-based open hierarchical architecture and hybrid control schemes tend to be
employed in sensor-based intelligent robot system. As an integrated system, the host
computer senses the environments by means of F/T sensor, imaging system, proximity
sensor, tactile sensor, etc. Perception and decision-making are based on sensor fusion and
hybrid control schemes are exploited to communicate with systemic components [70]. A
typical PC-based open architecture sensor-based intelligent robot system [11] is depicted
in Figure 2.23.
Pires and Costa [71] proposed a generic architecture of the robot applications based on an ABB industrial robot. The basic software structure is shown in Figure 2.23.

As far as the PC-based open hierarchical architecture intelligent robot system is concerned, the state-of-the-art application software module is decomposed into four sub-modules: Management, API, network and control objects. In addition, the functions of the application software module should be allowed to be extended or to be modified in terms of the requirements of implementation.
Figure 0.24: Basic structure of the software for robot applications

2.7 Conclusion

Plenty of issues related to intelligent robot systems have been reviewed, ranging from: Generic intelligent robot systems, state-of-the-art pattern recognition techniques, the analysis of intelligent gripping process, the latest gripper design and its applications, robot path planning, and PC-based intelligent robot gripping architecture on which my research project platform is based.
Robot vision system plays a pivotal role in intelligent robot systems. A complete robot vision system consists of the extraction of geometric features, data structures, describing and representing geometric features and model-based geometry recognition. As fuzzy logic has the features of modeling and solving the uncertain and nonlinear problem, flexible and robust part recognition can be constructed on the fuzzy inference system. Currently, various fuzzy image processing techniques have widened the conventional image processing approaches. Two kinds of object classification methods, template-matching techniques and structural techniques, are usually employed. Structural methods have been utilized in most applications recently. For the purpose of intelligent gripping, all possible faces must be reasoned out with reference to the boundary model extracted from the part recognition module. Gripping model is constructed eventually in terms of those faces. Heuristic approaches are effective and efficient to reason out the optimal faces to be grasped. Efficient geometric feature coding facilitates gripping planning. Syntactical description is an efficient way in gripping planning.

At present, different types of grippers have been designed for difference applications. Three fingered sensor-based grippers are capable of implementing intelligent gripping to a large variety of machine parts. For industrial robot based applications, robot trajectory planning is tightly associated with the gripper used. Intelligent gripping process is implemented with interaction of the manipulator and the gripper. The trajectory planning analyzes the grasp model, path constrains and other factors, and generates robot trajectory.
Chapter 3  System Setup: Gripper Design, Data Acquisition, Gripper Control, and Software Components

Intelligent gripping system based on industrial robot integrate hardware components, such as gripper control system, sensors, robot, and interface boards as well as software components. Modular system hardware formation and software design methodology is utilized to constitute the intelligent gripping system. The overall system is composed of relevant subsystems, each of which is able to be manipulated separately. Subsystems are synthesized seamlessly to form an integrated system. The flexibility and reconfigurability of the gripping system were developed in order to facilitate an application to adapt to different environments.

3.1  System Framework (Robot, Gripper, Sensory System)

The experimental setup for the intelligent gripping system is shown in Figure 3.1. The intelligent gripping system is made up of the following:

- An ABB industrial robot IRB-1400 and controller, which can be accessed through Ethernet or serial communication [72][73]

- An Servo linear actuator mounted on the wrist of the robot manipulator, which is driven by a Powermax hybrid stepper motor [74] under closed-loop control. The stepper motor controller and its optical encoder is interfaced to a PMAC board [75] through a relevant voltage-frequency converter [76]
Figure 0.1: The experimental setup for the intelligent gripping system
• A three-fingered gripper with one flat finger and two round fingers is attached to the linear actuator such that the position and speed of each finger can be controlled accurately.

• A robot vision which consists of a CCD camera [77] and an AGP bus frame grabber [78]. The camera is fixed over the workcell and takes images of the object. It is interfaced to the frame grabber which provides live images and is controlled through API’s.

• Robot touch which is achieved by attaching the tactile sensor to the flat finger of the gripper, which is capable of detecting the grasp force, tactile image, and slipping during the procedure of handling.

Aiming at constructing an open system, an IBM compatible PC is utilized as the host computer, the PC bus interface cards are used as the interface, modular software design based on object-oriented programming (OOP) facilitates the compatibility and reconfigurability. Thus, different applications can be set up with ease.

3.2 Linear Servo Actuator Design

Different types of actuators, ranging from electrical, hydraulic, pneumatic, to servo actuator are suitable for different applications. In this research, a three-fingered intelligent gripper will be developed, which requires that the finger position and speed be controllable. With reference to the specifications of the intelligent gripper and the payload of the robot manipulator, a compact linear servo actuator should be taken into account. For the reason of cost and appropriate control precision, stroke, and forces, etc., no commercial servo linear actuators suitable for this application have been found. Thus, a customized stepper motor driven linear servo actuator was developed.
3.2.1 Actuator Structure Design

To drive a three-fingered intelligent gripper, a servo linear motion source is needed. To convert motor rotary motion into linear motion, leadscrew and nut, gear and rack, cam and follower, are often exploited. For the sake of the compact structure, the combination of lead screw and nut is the most preferable structure, which was capable of increasing thrust force. In this research, 3D modeler Solidworks 2001 is utilized to make the mechanical design. The mechanical structure of the actuator is shown in Figure 3.2 with solid model in Figure 3.3. The detailed actuator design is referred to Appendix D.

![Figure 0.2: The assembly of actuator](image)

![Figure 0.3: The 3D shaded actuator](image)
Stepper motor, as the force source of the actuator, plays a crucial role in the actuator. The performance of the actuator is determined by the motor to large extent. In this application, Pacific Scientific model P2HNRXC-LDN-PD-00 is utilized. This model is a compact size, bipolar series, half stack, high-efficiency, and high-torque hybrid stepper motor. The principal specifications of this model are shown in Table 3.1.

**Table 0.1: P2HNRXC-LDN-PD-00 hybrid stepper motor principal specification**

<table>
<thead>
<tr>
<th>Holding Torque (Nm)</th>
<th>Detent Torque (Nm)</th>
<th>Rotor Inertia (Kgm²)</th>
<th>Weight (Kg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.43</td>
<td>0.018</td>
<td>7×10⁻⁶</td>
<td>0.45</td>
</tr>
</tbody>
</table>

The typical performance curve of this motor is shown in Figure 3.4.
In terms of specifications given in Table 3.1 and Figure 3.4, this model is able to provide high holding torque, high detent torque, lower inertia and light weight. With reference to the performance of this stepper, it is suitable for this application.

As is seen in Figure 3.1, an optical encoder is mounted on the shaft of the stepper motor on the backside, which ensures a position feedback. The stepper motor is connected to the leadscrew through a coupling which is attached by using setscrews on both sides. On the other hand, the stepper motor is mounted and positioned on the motor-connector with its shoulder fitting in recess. Two ball bearings are fitted in the motor-connector and support the leadscrew. The other end of motor-connector is mounted and positioned on the key-base in the same way as the fitting between the stepper motor and the motor base. A guide key and the keyway in the key-base assure a linear motion of the nut. The leadscrew-nut mechanism magnifies the drive force of the actuator and ensures that the object can still be safely held in the case of sudden power failure or motor failure. In addition, a finer positioning accuracy is also achieved. The left end of the housing is mounted on the robot wrist, while the right end is fitted in the key-base. The output shaft is mounted on the nut.

3.2.2 Analysis of Actuator Operation

The actuator must produce enough gripping force to secure the grasp. In addition, its stroke should cover reliable grasp of general size objects. The stroke of this linear actuator is 35mm. With reference to the mechanism of the designed gripper in Appendix D, the output range of radius is 21mm~55mm.
The output force of the actuator is generated by the stepper motor through the leadscrew-nut transmission mechanism [79][80]. The force analysis is conducted by using the static force analytical method. The simplified mechanics model of leadscrew-nut mechanism is depicted in Figure 3.5.

![Figure 0.5: Mechanics model of leadscrew-nut mechanism [81]](image)

- **Speed**

Motor speed (m/s): \[ W_M = V_L \cdot p \] (0-1)

- **Torque**

Load torque reflected to motor shaft (m-s): \[ T_L = \frac{1}{2\pi} \cdot \frac{F_L}{pe} + \frac{1}{2\pi} \cdot \frac{F_{FL}}{p} \times 0.2 \] (0-2)

Frictional force (N): \[ F_F = u \cdot W \] (0-3)

Friction torque (N): \[ T_{FL} = \frac{1}{2\pi} \cdot \frac{F_F}{pe} \] (0-4)

- **Inertia**

Total system inertia (N-m-s²): \[ J_T = \frac{W}{g} \left( \frac{1}{2\pi p} \right)^2 \cdot \frac{1}{e} + J_{LS} + J_M \] (0-5)

Where: \[ V_L \] — motor speed (m/s)

\[ p \] — lead screw pitch (revs/in)
In terms of Equation 3-2 and Equation 3-4, the output torque of motor should be greater than or equal to

\[ T_M \geq T_L + T_F \] (0-6)

Thus the load force is given as

\[ F_L \leq 2\pi peT_M - F_F - 0.2eF_{PL} \] (0-7)

In accordance with Equation 3-6, the maximum driven force that the actuator is able to produce is \((2\pi peT_M - F_F - 0.2eF_{PL})\).

### 3.3 Gripper Design

As is described in section 2.4, a three-fingered gripper is in possession of about 90% of the gripping abilities of a mechanical five-fingered “hand”. Thus, a three-fingered gripper is developed. It is desired that reliable and stable grasp can be conducted, objects with different size and different geometry can be handled, the gripper components should be easy to manufacture while maintenance, and the manufacturing cost would be low.
3.3.1 Gripper Structure Design

In this design, symmetric linkage mechanism was utilized to achieve the desired motion of three fingers. We take advantage of the function of magnifying output range of linkage mechanism. And furthermore, the configuration of the gripper is capable of being extended by changing relevant links so as to adapt to specific applications. Figure 3.6 shows the gripper assembly.

![Gripper Assembly Diagram]

**Figure 0.6: The three-fingered gripper assembly**

The three-fingered gripper structure comprises one actuator, three symmetrical crank-slider mechanisms and parallel four-bar linkages, and three fingers. The linear actuator drives the crank-slider mechanism in order that the linear motion of the slider is converted into the rotary motion of each crank. While the parallel four-bar linkage converts the rotary motion into translation of each finger [82]. To reduce the weight of the gripper, fingers are made of plastic while other components are made of aluminum.
The 3D shaded model of the gripper is shown in Figure 3.7. And the detailed gripper design is given in Appendix D.

The three fingers are composed of one flat finger where a tactile sensor is mounted and two round fingers. The three fingers, which are symmetrically distributed, move inwards or outwards about the centre simultaneously. Hence, the gripper is able to pick up mechanical parts with normal geometry, such as round, equilateral triangle, isosceles triangle, rectangle, hexagon and shapes with two parallel edges, etc.

3.3.2 Analysis of Gripping Process

The gripper is a mechanism which has one DOF. The function of grasp and release is implemented by converting the actuator’s linear motion into the finger's translation through mechanical transmission. With the help of the developed linear actuator, the
gripper is capable of conducting reliable and stable gripping in the process of gripping or delivering, even in the case of power failure.

3.3.2.1 Analysis of Gripper Kinematics

The schematically kinematic concept diagram is shown in Figure 3.8.

![Diagram of Gripper Kinematics](image)

**Figure 0.8: Kinematic concept diagram of the gripper mechanism**

According to Figure 3.8, the relationship between the position of fingertip and displacement of the actuator can be determined by Equation 3-8.

\[
\begin{align*}
    x_g &= l_g + 2l_1 \cos \theta \\
    y_g &= y + r + 2l_l \sin \theta - l_1
\end{align*}
\]  

(0-8)

where \(x_g\) denotes the fingertip position in radial direction, while \(y_g\) denotes its position in axial direction.

\[\theta = \gamma - \omega\]  

(0-9)
\[
\cos \gamma = \frac{l_1^2 + l_2^2 - l^2}{2l_1 \cdot l} \quad (0-10)
\]
\[
l' = \sqrt{x^2 + y^2} \quad (0-11)
\]
\[
\omega = \tan^{-1} \frac{y}{x} \quad (0-12)
\]

The speed of fingertips in radial and axial direction can be calculated in Equation 3-13.

According to Equation 3-12, we have
\[
\frac{d\omega}{dt} = -\frac{\sin \omega}{l'} \frac{dx}{dt} \quad (0-13)
\]

According to 3-10, we get
\[
\frac{d\gamma}{dt} = (\cos \gamma \cos \omega - \frac{x}{l_1}) \frac{1}{l \sin \gamma} \frac{dx}{dt} \quad (0-14)
\]

According to Equation 3-9, we obtain
\[
\frac{d\theta}{dt} = \frac{d\gamma}{dt} - \frac{d\omega}{dt} \quad (0-15)
\]

According to Equation 3-8, the velocity of fingertips is derived
\[
\begin{align*}
\frac{dx_s}{dt} &= -2l_1 \sin \theta \frac{d\theta}{dt} \\
\frac{dy_s}{dt} &= 2l_1 \cos \theta \frac{d\theta}{dt}
\end{align*} \quad (0-16)
\]

### 3.3.2.2 Analysis of Gripping Force

The force analysis diagram of slider and crank are shown respectively in Figure 3.9 (a) and (b). \(F\) is the actuating force generated by the linear actuator (see Equation 3-7), \(M_g\) is the moment reflected to the joint, \(F_I\) is the internal force of the link, while \(F_g\) is the grasping force applied to the object to be grasped.
The grasping force \( F_g \) can be derived from Equation 3-17, 3-18, 3-19, 3-20, 3-21, 3-22, and Equation 3-23, in accordance with Figure 3.9.

\[
\begin{align*}
\alpha &= \beta + \omega \quad (0-17) \\
F_{lx} &= F \quad (0-18) \\
F_{ly} &= F_{lx} \cdot \tan \theta \quad (0-19) \\
Mg &= F_g \cdot l_g \quad (0-20) \\
F_{lx} \cdot l_1 \sin \theta + F_{ly} \cdot l_1 \cos \theta &= M_g + F_g \cdot 2l_1 \cdot \cos \theta \quad (0-21) \\
F \cdot l_1 \sin \theta + F \cdot \tan \alpha \cdot l_1 \cos \theta &= l_g F_g + F_g \cdot 2l_1 \cdot \cos \theta \quad (0-22) \\
F_g &= Fl_1 \frac{\sin \theta + \tan \alpha \cos \theta}{l_g + 2l_1 \cos \theta} \quad (0-23)
\end{align*}
\]

The friction force which exerted between the fingers and the grasped object can be expressed

\[
F_f = F_g \cdot u \quad (0-24)
\]

where \( u \) is the coefficient of friction.
3.4 Implementation of System Control

The architecture of intelligent gripper system is depicted in Figure 3.10.

Figure 0.10: The architecture of intelligent gripper system
In this research, the intelligent gripper system architecture is composed of the following five principal modules:

- **Image processing module** — Performs image preprocessing which makes use of different filters, such as: Low pass, high pass, median, noise cleaning, averaging, and smoothing, etc. It then performs thresholding to produce a consistent binary image, contour detection, edge linking and edge following. Edge following is utilized to detect and extract the boundary of the object robustly.

- **Part recognition module** — Performs edge thinning to produce one pixel thick boundary and then extracts the geometric features from the boundary. It represents the boundary of the object with a chain of vectors and eventually describes the object with an Object_ID.

- **Gripping planning module** — Extracts geometric features, such as edge length, the geometric relationship of two edges (perpendicular, parallel, or angle between them), perimeter, area and centre of gravity (COG), from a database which is created in part recognition module. Then reasons out all possible combinations of faces which are suitable for grasp in terms of the configuration of the three-fingered gripper. Eventually derives the optimal combination of faces and pertinent reasonable gripping point for stable and reliable grasp.

- **Trajectory planning module** — Creates the relationship between image coordinate frame, robot coordinate frame and gripper coordinate frame. Then it builds up the transformation matrices to calculate each robtarget [83] data on the gripping path, and eventually generates the robot program. This program is transferred to the robot controller for execution in real-time.

- **Gripping control module** — Generates appropriate control schemes to optimize the control process. It implements robot control, touch image analysis and actuator control, accomplishes the intelligent gripping by adjusting the relevant control parameters in terms of the touch image feedback and actuator’s position feedback.
In Figure 3.10, the system hardware (CCD Camera, robot, tactile sensor, and actuator) are interfaced to the host PC through frame grabber, Ethernet card, data acquisition card, and PMAC card respectively. Figure 3.11 shows the PC bus interface cards in the host computer.

![PC bus interface cards](image)

**Figure 0.11: PC bus interface cards**

### 3.4.1 Robot Vision System Control

In robot vision system, the quality of captured image has a tremendous impact on the system performance. In this project, the hardware of robot vision system consists of a CCD camera and a frame grabber. The wiring diagram for robot vision system is depicted in Figure 3.11. Composite video in NTSC (12.27MHz at resolution of 640×480) is captured by the frame grabber. Both the frame grabber and the CCD camera are controllable, such as brightness, contrast, saturation, and so forth, through software.
3.4.1.1 Frame Grabber Control

A programmable frame grabber is important to the vision system. Flashpoint 3D frame grabber with PCI bus is employed in this application. This is a high-performance, low-cost PCI frame grabber which is able to capture and display full-frame color and video in real time to VGA display memory. It supports pixel format of 8/16/32 bits per pixel. It supports non-destructive overlay on live video. In addition, one optically isolated output trigger can be used to trigger illuminants, while RS-232 port can be used to communicate with the camera. Imaging can be controlled through API. Figure 3.13 shows setting and calibration of the frame grabber.
3.4.1.2 CCD Camera Control

CCD camera is the hardware core of vision system. In this research, SRC-503HP CCD camera is utilized. It is a high performance CCD camera. It supports high-resolution output up to 752×582V (440,000 pixels) and its scanning system provides 15,625 kHz (H) and 50 Hz (V). In addition, this model supports auto backlight compensation, auto white balance (AGC) and zoom. Especially, it provides a RS422C port for communication with the host computer. Therefore, all functions of this camera can be controlled through serial communication. To interface the camera to the frame grabber, A RS422 to RS232 converter is employed to convert the RS422 signal from the camera to RS232 signal which the frame grabber uses, as seen in Figure 3.12. Figure 3.14 shows setting and calibration of this CCD camera.
3.4.2 Tactile Sensor Interfacing

Combining vision and tactile sensing may lead to more precise and robust grasp. An intelligent robot equipped with vision and tactile sensing may perform tasks in a more flexible way. Tactile array signal is processed to provide a great deal of information about contact kinematics and precise tactile information. In robotics, a tactile sensor is used to obtain touch image and secure robust grasp [84]. Tactile sensors can compensate for uncertainties in object orientation angles and minimize forces on the object. Tactile sensing results in a gentler grasp in comparison to F/T sensing [85]. In this project, the
tactile sensor used is capacitive-based for outstanding sensitivity and durability. It consists of an 8×8 array of elements at 2mm spacing. Since the sensor is mounted on the fingertip, the fingertip area is very limited. Hence, the bias and output at small applied pressures must be repeatable and stable. The tactile array sensor system is shown in Figure 3.15.

![Diagram of tactile array sensor system]

**Figure 0.15: Wiring diagram of tactile array sensor**

The touch image is shown in Figure 3.16, while a finger presses on it. The magnitude of the applied force on each element is expressed by the color.

![Touch images](a) Press on the sensor, (b) 2D touch image, (c) 3D touch image

**Figure 0.16: (a) Press on the sensor, (b) 2D touch image, (c) 3D touch image**
3.4.3 Servo Actuator Control

The servo actuator is driven by a hybrid stepper motor with an optical encoder attached to the motor shaft at the rear end. The wiring diagram of actuator control is shown in Figure 3.17. The stepper motor control system is made up of five units, namely, power supply, PMAC card, voltage-to-frequency card, stepper motor drive, and optical encoder.

![Wiring Diagram of Actuator Control](image)

Figure 0.17: The wiring diagram of actuator control

The standard PMAC controller provides a PID position loop servo filter. The performance of the closed loop system is determined by the PID parameter setting of the PMAC to large extent. Figure 3.18 shows the diagram of the PMAC PID and NOTCH servo filter.
The proportional gain ("P" — Ix30) provides the stiffness of the system; the differential gain ("D" — Ix31) provides the damping for stability; while the integral gain ("I" — Ix33) eliminates steady-state errors. Ix34 determines whether the integral gain is active all the time or just during periods when the commanded velocity is zero. Therefore, the system I-variable Ix30~Ix35 must be appropriately set for a specific system.

The PMAC card sends analog voltage signal through its DAC (DAC1 and DAC1/) to implement the speed control of the stepper motor. Since the drive of stepper motor can only accept digital pulses, the voltage-to-frequency card is employed to convert voltage signal from the PMAC card into a pulse width modulated signal which is compatible with the stepper motor drive input. The frequency of this signal is proportional to the
magnitude of the velocity control voltage [86]. On the other hand, the direction control signal is generated through the amplifier enable bit (AENA1/DIR1), which is connected to the direction input of the voltage-to-frequency (DIR1).

The outputs of the voltage-to-frequency card on connector TB1 are the direction signal and the pulse width modulated pulse signal at TTL levels. The frequency of the output pulses is directly proportional to the magnitude of the control voltage from the DAC1 and DAC1/ of the PMAC card. The pulse and direction output of the voltage-to-frequency is tied to STEP- and DIR- of the stepper motor drive respectively. When logic 0 is present on the STEP-, the opto-isolator goes ON. Every transition of the opto-isolator from OFF to ON generates the execution of a single step of the stepper motor.

The outputs of the stepper drive are a pair of differential signals that provide excitation voltage to the windings of a stepper motor. The stepper motor is of bipolar stepper motor. The outputs (A, A, B, B, and GND) of the motor drive are tied to the corresponding wires of the motor respectively.

The incremental optical encoder, which outputs 4 pulse signals in quadrature mode, performs the feedback from the stepper motor. A resolution of 2048 pulses per revolution is achieved in this application. Outputs from the encoder are fed directly to the PMAC quadrature encoder inputs. The PMAC decodes the inputs to set its relevant system variables to indicate the current speed and position of the stepper motor.
3.5 Software Components for Experimental Setup

As a real-time automated part recognition and gripping system, the system must be highly integrated so as to handle objects at the scene in process. The software is developed using Microsoft Visual C++ on the Windows environment (Win9x, WinMe, WinNT, Win2000, or WinXP). Object oriented and modular software design techniques are utilized to develop the application. Microsoft Foundation Class (MFC) is used as the development platform [87][88]. Windows-based application guarantees friendly graphics user interface (GUI). MFC programs are highly portable, because the interface provided by the MFC library is largely independent from the details of its underlying implementation. The efficient and reliable code generated by Visual C++ ensures the reliability and stability of the software, and applicability in real-time system. Figure 3.19 depicts system software components for the PC-based intelligent gripping system.

The mainframe of the application is a single document-view application. All functional modules are integrated in the application. Class CIelligentGripperDoc, derived from class CDocument, organizes and maintains all system data, and updates data through MFC document-view structure. All system configuration files are accessed through this class. Class CIelligentGripperView, derived from CView, serves as the host user interface, which initiates all kinds of system function modules.
Figure 0.19: System software components for the PC-based intelligent gripping system
The five principal modules (image processing, part recognition, gripping planning, manipulator trajectory planning, gripping status monitoring, tracking and gripper control) are encapsulated and implemented in relevant classes.

3.5.1 Image Processing Module — CImageProcessing

CImageProcessing is responsible for taking image of the object at the scene. The output of this module is the profile of the object which is represented by chain code, as well as geometric features (area, perimeter, etc.). The input image is RGB color image. The memory that saves the image is allocated dynamically. The data structure RGBIMAGE is defined as follow:

```c
typedef struct tagRGBIMAGE {
    CPoint point;
    CSize size;
    LPBYTE lpImage;
    LPBYTE AllocateMemory();
    void FreeMemory();
    LPBYTE GetImage() const;
} RGBIMAGE;
```

The profile of an object is stored in a CList object, which behaves like double-linked list. The CList class supports ordered lists of nonunique objects accessible sequentially or by value. Unlike array access, element insertion is very fast at the list head, at the tail, and at a known position. Hence, the computation time can be reduced significantly. The one pixel thick profile of the object is defined as

```c
CList<CPoint, CPoint&> m_profile;
```
Convolution technique is the key to image processing in spatial domain. The convolution function is implemented as:

```cpp
void CImageProcessing::Convolution(const int nConvolution[3][3], int nConstant) {
    int i, j;
    int nRow, nCol;
    int nTemp;
    for(i = 1; i < m_imageSize.cy - 1; i++) {
        for(j = 1; j < m_imageSize.cx - 1; j++) {
            nTemp = 0;
            for(nRow = -1; nRow <= 1; nRow++) {
                for(nCol = -1; nCol <= 1; nCol++) {
                    nTemp += *(m_lpImage + (i + nRow) * m_imageSize.cx + j + nCol) * nConvolution[nRow + 1][nCol + 1];
                }
            }
            nTemp /= nConstant;
            *(m_lpImageTemp + i * m_imageSize.cx + j) = nTemp;
        }
    }
}
```

To adapt to different environments, apart from the default edge operator — Laplacian operator, some other edge operators are also provided. The detailed setting of active edge operator, image filters, etc. is referred to Appendix A.

### 3.5.2 Part Recognition Module — CObjectRecognition

To represent the object efficiently and effectively, we take advantage of the information implicated in the profile of the object. Image processing conducts intensive computation. Extracting the useful information from the profiles means a considerable reduction in the
volume of data to be processed, object profiles are relatively stable under different circumstances. The part recognition module works on object profile that is the output of image processing module. The profile of an object is represented by a closed chain of vectors. A continuous fuzzy controller is developed to accomplish the creation of the closed chain of vectors. CFuzzy_CFC class is used to implement the operation of this fuzzy continuous controller.

class CFuzzy_CFC
{
    public:
        CFuzzy_CFC();
        virtual ~CFuzzy_CFC();
    private: // Data
        // the maximum and minimum values for inputs and output
        double m_dMinX, m_dMaxX, m_dMinY, m_dMaxY, m_dMinZ, m_dMaxZ;
        // normalized inputs and output
        double m_dNormX, m_dNormY, m_dNormZ;
        // Degree of membership regarding each x, y, z;
        double m_dX_DM, m_dY_DM, m_dZ_DM;
        double m_dZ1, m_dZ2, m_dZ;

        // KNOWLEDGE BASE
        // Rule Base, 25 rules in all
        int* m_nRules[25];
    public: // Functions
        void SetupRuleSet(int nRuleSet[25][3]);
        // Interface
        double Interface_CFC(double x0, double y0, int nDefuzzyMethod = 0);
    private:
        // Normalisation
        void XNormalisation(double x0);
        void YNormalisation(double y0);
        // FUZZIFICATION
        // Membership Functions
        void MFX_NL(); void MFX_NS(); void MFX_Z(); void MFX_PS(); void MFX_PL();
        void MFY_NL(); void MFY_NS(); void MFY_Z(); void MFY_PS(); void MFY_PL();
        void MFZ_NL(); void MFZ_NS(); void MFZ_Z(); void MFZ_PS(); void MFZ_PL();
        void TransformZValueToNaturalState();
        // INFERENCE ENGINE
        void InferenceEngine_WMM();
        void InferenceEngine_MOM();
};
The fuzzy IF-THEN set is stored in a MS Access database. When part recognition module initializes, the program access to the fuzzy IF-THEN set by using the Microsoft Open Database Connectivity (ODBC) standard. We take advantage of the ODBC Software Development Kit (SDK), included with Visual C++, containing 32-bit drivers for MS Access MDB databases. ODBC’s unique DLL-based architecture makes the system fully modular. A small top-level DLL, ODBC32.DLL, defines the API. ODBC32.DLL loads database-specific drivers, during program execution. Figure 3.20 depicts the architecture of 32-bit ODBC MDB access [87].

![Figure 0.20: Architecture of 32-bit ODBC MDB access](image)

The output data structure of part recognition module is defined as

```c
typedef struct tagGEOMETRYDATA {
    int     m_nGeometryID;
    double  m_radius;
    CVector m_orientation;
    CPoint  m_COG;
    double  m_dArea;
    int     m_nVectorCount, m_nWorkVectorCount;
    CList   <CVector, CVector&> m_vectorChain;
    CVector* m_workChainV;
    CVector  m_mainV, m_auxV1, m_auxV2;
    BOOL    m_bGripping;
    int     m_nVectorChainDirection; // CW/CCW
} GEOMETRYDATA;
```

The chain of vectors that represent the profile of the object is stored in CList template derived object m_vectorChain. Therefore, the efficiency of access to the profile can be increased
dramatically. This module extracts all necessary geometric features of the object, such as geometric ID, radius if it is of round object, position expressed by centre of gravity (COG), orientation, area, direction of the vector chain (CW/CCW), and the vector chain. The detailed setting about part recognition is referred to Appendix A.

3.5.3 Gripping Planning Module — CGrippingPlan

For intelligent gripping system, handling objects with different geometry in terms of the configuration of the gripper appropriately and robustly is the core of the system. Gripping planning module analyzes and extracts the geometric features of the object to be grasped, which is saved in the struct tagGEOMETRYDATA. The target of this module is to determine the grasp centre and position of each fingertips that is represented by using the fingertip vector. The data described above are defined as:

```cpp
CVector m_mainV, m_auxV1, m_auxV2;
POINT3D m_centrePt;
```

m_mainV, m_auxV1, m_auxV2 represent the flat finger on which the tactile sensor is mounted and two round fingers respectively. Figures 3.21 shows finger configuration of the gripper when gripping two objects with circle and rectangle profile respectively.
The function that deals with equilateral triangle gripping planning is implemented as follows.

The finger vectors, \( m_{\text{main}}V \), \( m_{\text{aux}}V1 \), and \( m_{\text{aux}}V2 \), are worked out in terms of the geometric features.

```cpp
void CGrippingPlan::EquilateralTrianglePlanning(CObjectRecognition* pObjectRecognition,
                                              GRIPPINGPLANPARAM* pParam)
{
    CVector v;
    double dFingerR = pParam->m_dFingerRadius * pObjectRecognition->m_dVisionScale;
    double dCompensation = pParam->m_dProfileCompensation *
                          pObjectRecognition->m_dVisionScale;
    double dActiveR = dFingerR + dCompensation;
    int nDir = pObjectRecognition->m_geoData.m_nVectorChainDirection;

    m_mainV = pObjectRecognition->m_geoData.m_mainV;
    v = m_mainV;
    m_mainV.Normalize();
    m_mainV = m_mainV.RotateVector(nDir == CW ? 90 : -90);
    v = v.MultipledByConstant(0.5);
    m_mainV = CVector(v.EndPoint(), m_mainV.x, m_mainV.y, TRUE);
    v = m_mainV;
    m_mainV.Normalize();
    m_mainV = CVector(v.EndPoint(), m_mainV.x, m_mainV.y, TRUE);

    m_auxV1 = pObjectRecognition->m_geoData.m_auxV1;
    v = m_auxV1;
    m_auxV1.Normalize();
    m_auxV1 = m_auxV1.RotateVector(nDir == CW ? 90 : -90);
    v = v.MultipledByConstant(0.5);
    v = CVector(v.EndPoint(), m_auxV1.x, m_auxV1.y, TRUE);
    v = v.MultipledByConstant(dActiveR);
    m_auxV1 = CVector(v.EndPoint(), m_auxV1.x, m_auxV1.y, TRUE);

    m_auxV2 = pObjectRecognition->m_geoData.m_auxV2;
    v = m_auxV2;
    m_auxV2.Normalize();
    m_auxV2 = m_auxV2.RotateVector(nDir == CW ? 90 : -90);
    v = v.MultipledByConstant(0.5);
    v = CVector(v.EndPoint(), m_auxV2.x, m_auxV2.y, TRUE);
    v = v.MultipledByConstant(dActiveR);
    m_auxV2 = CVector(v.EndPoint(), m_auxV2.x, m_auxV2.y, TRUE);

    CalculateCentreFromGrippingVectors();
    m_bGrippingPlan = TRUE;
}
```
The gripping planning output image for the above two objects is shown in Figure 3.22.

![Output image of gripping planning](image)

**Figure 0.22: Output image of gripping planning**

The object geometry which is suitable for gripping, including circle, triangle (isosceles-triangle, equilateral-triangle), rectangle, profiles with two parallel edges, and hexagon, is dealt with individually in different functions.

### 3.5.4 Robot Trajectory Planning Module — CTrajectoryPlanning

Trajectory planning is to create robot trajectory in terms of the data that gripping planning module generated. To make trajectory planning, object position and orientation, gripper configuration, robot kinematics and its instruction system must be taken into consideration. The transformation matrices are worked out in terms of the relationship of the camera coordinate system, robot vision coordinate system and robot coordinate system. To organize the data that is in association with the relationship mentioned above, data structure must be
well defined in order to facilitate the computation of robot path. Gripper data structure is defined as [73]:

typedef struct tagTOOLDATA {
    BOOL robhold;
    double trans_x, trans_y, trans_z;
    double rot_q1, rot_q2, rot_q3, rot_q4;
    double toolmass;
    double cog_x, cog_y, cog_z;
    double aom_q1, aom_q2, aom_q3, aom_q4;
    double ix, iy, iz;
} TOOLDATA;

In TOOLDATA, the grasp centre point of the gripper is regarded as the tool centre point (TCP), the tool coordinate system is attached to the wrist coordinate system of the robot. 4-point TCP is utilized to define gripper coordinate system.

Apart from the tool coordinate system, the object coordination system must be acquired as well. The data structure of the object coordinate system is defined as:

typedef struct tagWOBJDATA {
    BOOL robhold;
    BOOL ufprom;
    CString ufmec;
    double uframe_trans_x, uframe_trans_y, uframe_trans_z;
    double uframe_rot_q1, uframe_rot_q2, uframe_rot_q3, uframe_rot_q4;
    double oframe_trans_x, oframe_trans_y, oframe_trans_z;
    double oframe_rot_q1, oframe_rot_q2, oframe_rot_q3, oframe_rot_q4;

    void CreateObjFrame(double x, double y, double z,
                        double q1, double q2, double q3, double q4) {
        oframe_trans_x = x;  oframe_trans_y = y;  oframe_trans_z = z;
        oframe_rot_q1 = q1;  oframe_rot_q2 = q2;  oframe_rot_q3 = q3;
        oframe_rot_q4 = q4;
    }
} WOBJDATA;

cxii
In WOBJDATA, the translation and rotation data are defined as variable names imply. User coordinate system is also encompassed.

To facilitate creation of manipulator path, ROBTARGET is defined to describe all data that is needed to generate robot path.

```c
typedef struct tagROBTARGET {
    double trans_x, trans_y, trans_z;
    double rot_q1, rot_q2, rot_q3, rot_q4;
    int robconf_cf1, robconf_cf4, robconf_cf6, robconf_cfx;
    double eax_a, eax_b, eax_c, eax_d, eax_e, eax_f;

    void Create(double x, double y, double z,
                double q1, double q2, double q3, double q4, int cf1, int cf4, int cf6) {
        trans_x = x;    trans_y = y;    trans_z = z;
        rot_q1 = q1;    rot_q2 = q2;    rot_q3 = q3;    rot_q4 = q4;
        robconf_cf1 = cf1;  robconf_cf4 = cf4;  robconf_cf6 = cf6;
        robconf_cfx = 0;
    }
} ROBTARGET;
```

3.6 Conclusion

The intelligent gripping system is capable of responding to the changing scene intelligently. The proposed PC-based intelligent gripping system is developed under Microsoft Visual C++. Object-oriented techniques are employed in the software development. All data and functions are encapsulated in relevant C++ classes to form the system architecture. System modules are integrated and implemented by making use of modular software design technique.

PC-bus interface cards (frame grabber, PMAC card, data acquisition card, and Ethernet card) are utilized, as well as ActiveX techniques are widely exploited to build up a configurable
system. To maximize the applicability to grasping objects various geometry, three-fingered gripper is developed, which is able to grasp commonly-seen mechanical parts. In addition, it is capable of handling soft objects with the presence of the tactile sensor. 3D modeler Solidworks 2001 is used to develop the three-fingered gripper so as to speed up the design period and validate the motion mechanism during designing. The intelligent gripper is powered with the developed servo actuator such that the position and speed of the gripper can be controlled precisely and robustly. To get the fingertip position and speed under precise control, the relationships between the actuator and the fingertip with regard to the actuator motion are generated. The proposed architecture in Figure 3.10 puts forward a generic framework for intelligent gripping system, which organizes the system hardware and software components and reveals their relationships.
Chapter 4  Robot Vision System

Robot vision plays a critical role in robot intelligence. It starts with visual data collection, ends with description and interpretation of object geometric features. In robot vision systems, geometric feature extraction and representation are the two most important issues to which we must find solution in terms of the application requirements. A good number of object recognition and feature representation methods and algorithms have been investigated in different applications and for different purposes. To make the present robot vision systems suitable for various eye-hand applications, further researches and improvement still need to be done.

To constitute a gripping-oriented robot vision system, the following issues must be taken into consideration:

- **Image acquisition** — includes selection of visual sensors in terms of image resolution of which the vision system must be in possession, illumination and other specifications.

- **Image processing techniques** — encompasses methodology for image processing in terms of the requirements for real-time, complexity of computation and precision. Furthermore, whether object’s contour or entire image should be exploited to identify the object. In addition, whether color or grayscale image will be employed to parse the feature of the object.

- **Representation and description of object’s geometric features** — is dependent on the application to which the vision system is going to be applied. Common techniques are B-
Rep, CSG or feature-based representation. In addition, classification or model-based object matching may be made with regard to the object model.

- Irregularity handling functions — deal with uncertain irregularities that may occur on the scene or during the computation.

This research focused on extraction, description and interpretation of those features that are significant to the gripping-oriented application and capable of facilitating gripping planning. As an intelligent gripping system, the object position and orientation as well as the object’s profile must be identified and represented accurately. Hence, a high resolution CCD camera is utilized to acquire visual information of the object to be grasped. A real-time intelligent gripping system needs to overcome the bottleneck of intensive computation of a vision system. Contours are exploited to represent the topology of faces, since the contours have the merit of their relative stability under fluctuations in the lighting of the scene and are able to provide the concise topology of faces. To represent the objects efficiently and effectively, edge vector expression method is developed such that computational efficiency is increased dramatically. To extract edge vectors, fuzzy vector tracing method based on a continuous fuzzy controller and heuristic vector tracing method are developed. A closed chain of vectors is generated. Thereto, profile traversing and feature extracting are conducted with respect to the chain. Classification is based on structural techniques, also known as syntactic pattern recognition, which consider relationships between features or edges of an object, and deduce the geometry ID in terms of the topology of edge vectors.
4.1 Architecture of Robot Vision System

The architecture of robot vision system may be considered a hierarchical architecture. The robot vision functionalities may be partitioned into firstly low-level vision. It includes those processes which are primitive and requires no intelligence such as image acquisition and image preprocessing. Secondly, high-level vision includes extracting, modeling, recognition, description, and cognition. In this study, the objectives of vision system are to produce data that represent the gripping-oriented profile and identify the geometry. The architecture of robot vision system implemented in this research is shown in Figure 4.1.

![Diagram of Robot Vision System Architecture](image)

**Figure 0.1: Schematic architecture of robot vision system**

In this study, the execution of high-level vision supplies necessary geometric features, such as area, length, width, COG, chain of vectors and object geometry ID for the successive gripping planning module.
4.2 **Image Processing**

Image processing starts with image sensing, preprocessing the captured image by means of various filters and templates to alleviate the distortion of original image to acquire a clear and accurate image. Then proceeds to extract the profile by edge operators, rehabilitate any drawbacks on the profile and concludes with a one-pixel-wide output profile.

4.2.1 **Image Sensing**

The image quality captured has a tremendous effect on the vision system. A high-resolution CCD camera along with a frame grabber is employed to guarantee imaging. Apart from controlling the camera to ensure an ideal image, the programmable frame grabber can also be used to adjust the imaging parameters. This is important to acquire an image with moderate brightness, contrast and sharpness, etc. To facilitate image processing and increase precision of vision system, a clear contrast between the object and the background is always needed. In this study, dark objects are placed on a white background. Light objects placed on a black background are also allowed as an alternative. In addition, the illumination is crucial to the vision system in some cases. To eliminate shallow and nonuniform illumination and all drawbacks that cannot be compensated by adjusting work parameters of camera and frame grabber, an illumination system is designed and set up to form a perfect image sensing system. Figure 4.2 shows the captured image of an object.
The captured image is stored in the VGA memory of the computer. RGB color image is then saved in program buffer for further processing. The detailed data structure is described in Section 3.5.1.

4.2.2 Image Preprocessing

To extract the profile of the object, image preprocessing is employed to apply filtering, noise cleaning, averaging, thresholding and edge operator to the captured image sequentially. The spatial convolution technique of digital image is the basis for this purpose.

4.2.2.1 Image Filtering

Three types of filters (high-pass filter, median filter, and low-pass filter) are available in this system. These filters can be combined to yield ideal image.

- High-pass filtering

High-pass filtering is employed to sharpen images that are out of focus or fuzzy. The function uses a $3 \times 3$ convolution filter, which emphasizes differences in gray level in the 3 by 3 neighborhood about the central pixel in the window. After high-pass filtering, the image is sharpened and low frequency components in the image are removed. The template used in this application is described in Section 2.2.1.3.
• Median filtering

Median filtering is exploited to remove random noise spikes on very noisy images. Unlike the other operations, median filtering makes use of a non-linear sorting of intensity levels in a $3 \times 3$ window to smooth out ‘salt & pepper’ noise. After median filtering, the image is smoothed and random noise becomes less apparent. The implementation of median filtering selects windows of pixel data from a $3 \times 3$ array of pixels. It ranks the current set of nine pixel intensities in order of grayscale magnitude and replaces the central point with median intensity value. The whole image is processed in turn by sliding the window over the entire image.

• Low-pass filtering

Low-pass filtering is utilized to smooth out a sharp image and high frequency components in the image are removed. A $3 \times 3$ convolution filter is used so as to remove the high frequency components.

4.2.2.2 Noise Cleaning

Noise cleaning technique is used to remove random noise signal on the captured image. A $3 \times 3$ convolution function is utilized to smooth out this noise. The smoothed and random noise clear image is yielded through this technique.

4.2.2.3 Averaging

Averaging technique is utilized to remove random noise spikes and clean edge features in the image. The ‘Averaging’ function uses a $3 \times 3$ sliding convolution window to smooth out noise.
4.2.2.4 Thresholding

Thresholding is a particularly useful technique for establishing boundaries in images which contain solid objects resting on a contrasting background. The process of thresholding is that a threshold greylevel is selected first and then all pixels below that greylevel are assigned to the background while all pixels with greylevel at or above the threshold are considered the object to be recognized. In this research, both manual thresholding and adaptive thresholding are put into use. The greylevel ranges from 0~255. That is to say 256 greylevel might be specified as the threshold theoretically. But only a subset of the 256 greylevel can be used as the threshold. It is found that values through 100 to 200 can be used for this purpose. As to the manual thresholding, the threshold is determined with regard to the illumination and the contrast between the object and the background, if the color of background is not constant, and object contrast varies within the image. In such cases, a threshold that works well in one area might work poorly in other areas of the image. In these cases, adaptive thresholding may be more effective and practical.

Figure 4-3 shows a clear and accurate binary image achieved through image smoothing (filtering, noise cleaning, and averaging) and thresholding.
4.2.3 Boundary Detection

The use of boundary detection in vision systems may simplify and tremendously accelerate the process of image processing. The boundary features a sharp greylevel transition. If the edges are reliably strong, and the noise level is low, one can threshold an edge magnitude image and thin the resulting binary image down to single-pixel-wide closed, connected boundaries. But certain discontinuities at the boundaries result from noise, shallow, and other interferences. In addition, unfortunately all known edge operators fail to result in ideal single-pixel-wide boundaries. Hence, edge linking and edge thinning must be applied after edge detection.

4.2.3.1 Edge Detection

Edge detection is implemented by examining each pixel neighbourhood and quantifying the slope and the direction as well in some cases by means of edge detection operators. Spatial convolution templates are always employed. In this study, options of edge detection are offered. The edge detection operators are convolution kernels that implement directional derivatives. The main edge detection operators implemented in this application are:

- Laplacian edge operator

The Laplacian operator is a second derivative operator. It can be implemented digitally by convoluting the whole image with the following 3×3 template.

\[
H_{\text{Laplacian}} = \begin{bmatrix}
-1 & -1 & -1 \\
-1 & 8 & -1 \\
-1 & -1 & -1 \\
\end{bmatrix}
\]
• **Sobel edge operator**

The *Sobel* operator uses two $3 \times 3$ edge convolution operators. One detecting X-gradient $G_x$, the second Y-gradients $G_y$, then uses the sum of squares of each gradient $G$ to determine the magnitude and orientation of an edge. The two templates are defined as

$$H_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad \text{and} \quad H_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

$$G = \sqrt{G_x^2 + G_y^2} \quad (0-1)$$

• **Kirsh edge operator**

Eight $3 \times 3$ convolution templates are used, one for each of the compass directions. The maximum edge gradient evaluated is retained for each pixel in the new edge enhanced image. Edge maps yielded by means of this operator are rather thick.

$$H_1 = \begin{bmatrix} 3 & 3 & -5 \\ 3 & 0 & -5 \\ 3 & 3 & -5 \end{bmatrix} \quad H_2 = \begin{bmatrix} 3 & -5 & -5 \\ 3 & 0 & -5 \\ 3 & 3 & 3 \end{bmatrix} \quad H_3 = \begin{bmatrix} -5 & -5 & -5 \\ 3 & 0 & 3 \\ 3 & 3 & 3 \end{bmatrix} \quad H_4 = \begin{bmatrix} -5 & -5 & 3 \\ -5 & 0 & 3 \\ 3 & 3 & 3 \end{bmatrix}$$

$$H_5 = \begin{bmatrix} -5 & 3 & 3 \\ -5 & 0 & 3 \\ -5 & 3 & 3 \end{bmatrix} \quad H_6 = \begin{bmatrix} 3 & 3 & 3 \\ -5 & 0 & 3 \\ -5 & -5 & 3 \end{bmatrix} \quad H_7 = \begin{bmatrix} 3 & 3 & 3 \\ 3 & 0 & 3 \\ -5 & -5 & -5 \end{bmatrix} \quad H_8 = \begin{bmatrix} 3 & 3 & 3 \\ 3 & 0 & -5 \\ 3 & -5 & -5 \end{bmatrix}$$

• **Compass gradient edge operator**

Eight $3 \times 3$ convolution templates are used, one for each of the compass directions. The maximum edge gradient evaluated is retained for each pixel in the new edge enhanced image. Edge maps yielded by means of this operator are rather thick as well. These templates are
oriented more towards detecting corners in the eight main directions indicated by a compass
deck (hence its denomination), than to detecting straight-line boundaries.

$$H_1 = \begin{bmatrix} -1 & -1 & -1 \\ 1 & -2 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad H_2 = \begin{bmatrix} -1 & -1 & 1 \\ 1 & -2 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad H_3 = \begin{bmatrix} -1 & 1 & 1 \\ -1 & 2 & 1 \\ -1 & 1 & 1 \end{bmatrix} \quad H_4 = \begin{bmatrix} 1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & -1 & 1 \end{bmatrix}$$

$$H_5 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & -1 & -1 \end{bmatrix} \quad H_6 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & -1 \\ 1 & -1 & -1 \end{bmatrix} \quad H_7 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & -1 \\ 1 & -1 & 1 \end{bmatrix} \quad H_8 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & -1 \\ 1 & 1 & -1 \end{bmatrix}$$

- **Simple** edge operator

Simple edge operator is utilized to emphasize edge detail in any orientation from a cleaned
image by emphasizing the intensity gradient across the image map. It is equivalent to a 2×2
template. Assume that the pixel $x_{ij}$ is being processed, the pixel array is as follows:

$$\begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{bmatrix} \quad (0-2)$$

Then the gradient $G$ is defined as:

$$G = \sqrt{(x_{11} - x_{22})^2 + (x_{12} - x_{21})^2} \quad (0-3)$$

- **Roberts** edge operator

Roberts is utilized to emphasize edge detail in any orientation from a cleaned image by
emphasizing the intensity gradient across the image map. It is equivalent to a 2×2 template.
The pixel array is shown in Equation 4-2. The gradient $G$ is defined as:

$$G = \sqrt{(x_{i1} - x_{i2})^2 + (x_{i1} - x_{i2})^2} \quad (0-4)$$

cxxiv
In general, 2×2 operators respond best and fast to sharp transitions in low-noise images. While 3×3 operators better handle more gradual transitions and noisier images. In this research, *Laplacian* operator is found to work well with most objects and relevant environments and yields thin boundaries. As a result, *Laplacian* operator is set to the default operator in this application.

### 4.2.3.2 Edge Linking

Ideally, the techniques of contour detection should yield only pixels lying on the boundary between objects and the background. In practice, this set of pixels seldom characterizes a boundary completely because of noise, breaks in the boundary due to nonuniform illumination and other effects that introduce spurious intensity discontinuities. In this application, linear interpolation is employed to create a continuous boundary. Figure 4.4 (a) shows discontinuity from point A to point B on boundary, while (b) shows the principle of linear interpolation.

![Figure 0.4: (a) Discontinuity from A to B  (b) Principle of linear interpolation](image)

The discontinuity from A to B is expressed with line segment *OP* in Figure 4.4 (b). Assume the distance from origin *O* to point *P*(X<sub>e</sub>, Y<sub>e</sub>) is X<sub>e</sub> pixels in x axis, Y<sub>e</sub> pixels in y axis. The linear interpolation method is employed to join point *O* and point *P*. Assume that the current
coordinate of the point being interpolated is saved in coordinate variable. \( X, Y, X_R \) and \( Y_R \) stand for registers that store the interpolation variables in \( X \) and \( Y \) axes respectively. The block diagram that explains the interpolation principle is shown in Figure 4.5.

**Figure 0.5: Block diagram of interpolation principle**

Edge linking gives boundaries without discontinuity. The profile is then traced and thinned to single-pixel-wide to facilitate extraction of geometric features.
4.2.3.3 **Edge Following and Thinning**

Chain code is an efficient method that is used to represent the profile. In this research, chain codes are employed to represent a boundary as a set of straight-line segments of specified length and direction. Chain codes are organized by means of double-linked list to increase the addressing and computation speed a great deal. Edge thinning is performed while edge following is being executed, so that single-pixel-wide profile is acquired. The flow chart of edge following and edge thinning is shown in Figure 4-6. 8-connectivity chain code is used in this study.

Scan the image from lower-left corner to find out the start point $P_0$, and add it to the double-linked chain.

Initialize variables
Set searching square size $m=3$ pixels

Examine pixels on the searching square from upper-right corner at CCW direction

Edge Point $P_i$ found?

Y
Add $P_i$ to the chain by appending it to the double-linked list

Set all points except $P_i$ on the current searching square as background points to thin the boundary

Reset searching square size $m=3$ pixels

The chain

End
Figure 0.6: Flow chart of edge following and edge thinning

The chain code contains the start pixel address followed by a string of code words. And single-pixel-wide boundary is achieved. Figure 4-7 shows the profile of the object after edge linking, edge following, and edge thinning.

Figure 0.7: Profile of the object after edge linking, edge following and edge thinning

4.3 Object Recognition (Feature Extraction, Geometry Recognition, and Description)

Feature recognition and representation is core in object recognition. The objective of this research is to grip objects intelligently. Hence, object recognition must provide all necessary geometric features for intelligent gripping. Inherent features of objects are always invariants with regard to translation, rotation, and scale. To extract those inherent features intelligently is that needs to be solved in this section.

4.3.1 Object Features

Among features, COG is the most important feature that must be figured out in object recognition, since it has great effect on stability and reliability of gripping process. The area of an object has a close relation to the weight of object to be grasped, i.e. the gripping force that
should be applied. In addition, the length and width are also needed for gripping planning in some cases.

- Calculation of COG

In digital image processing, COG \((x_c, y_c)\) can be calculated simply by taking into consideration pixels that stand for the object through the following equation

\[
\begin{align*}
    x_c &= \frac{\sum_{i=1}^{N} x_i}{N} \\
    y_c &= \frac{\sum_{i=1}^{N} y_i}{N}
\end{align*}
\]

(0-5)

- Calculation of area

In digital image processing, area \(A\) can be calculated simply by counting pixels that stand for the object.

- Calculation of length and width

![Diagram](image)

**Figure 0.8: Definition of length \(L\) and width \(W\)**
Length $L$ is defined as the longest chord $l$ in the object. While the width $W$ is the addition of the two longest line segments on either side of the chord $l$, which are perpendicular to the chord $l$. Figure 4.8 depicts the definition of length $L$ and width $W$.

\[ W = W_1 + W_2 \quad (0-6) \]

### 4.3.2 Extraction and Representation of the Profile

Extraction and representation of the profile of objects are the crucial issues to which a gripping oriented application must find solution. In this project, the profile is represented by the proposed edge vector representation (EVR), i.e. using a closed chain of vectors to represent the object profile. In practice, objects with different types of geometry have been tested. The EVR is proved to be a very effective way to represent the object profile and moreover it is capable of providing necessary information for object geometry reconstruction that is crucial to gripping planning. To extract the closed chain of vectors, a continuous fuzzy controller (CFC) and vector tracing method (VTM) were developed to achieve this task.

#### 4.3.2.1 Vector Tracing Method

VTM is developed to represent the profile of the object with a closed chain of vectors, i.e. the profile is represented by a polygon. VTM allows for curves in the profile of objects. Curves are approximated by a group of short vectors. The precision of approximation can be adjusted through a system variable which is accessible through user interface. To reduce the approximation error to line segment, two adjacent vectors are merged if the angle between them is less than a given tolerance which is adjustable by means of interaction. The kernel of this method is to locate the corner points with given tolerance.
• Vector algebra fundamental [89][90]

Length of vector \( v = ix + jy \) is defined as

\[
|v| = \sqrt{x^2 + y^2}
\]

(0-7)

Unit vector \( v_0 \) of vector \( v \) is defined as

\[
v_0 = \frac{v}{|v|} = \frac{ix + jy}{\sqrt{x^2 + y^2}}
\]

(0-8)

Angle \( \theta \) between two vectors \( v_1 \) and \( v_2 \) can be calculated by

\[
\theta = \cos^{-1}(v_{10} \cdot v_{20})
\]

(0-9)

\( v_{10} v_{20} \) stands for the dot product (or inner product) of vector \( v_{10} \) and vector \( v_{20} \) which are the unit vectors of vectors \( v_1 \) and \( v_2 \) respectively.

Discriminator of two orthogonal vectors \( v_1 \) and \( v_2 \) is defined as

\[
v_{10} \cdot v_{20} = 0
\]

(0-10)

Discriminator of two parallel vectors \( v_1 \) and \( v_2 \) sharing the same direction is defined as

\[
v_{10} \cdot v_{20} = 1
\]

(0-11)

Discriminator of two parallel vectors \( v_1 \) and \( v_2 \), having opposite directions, is defined as

\[
v_{10} \cdot v_{20} = -1
\]

(0-12)

• Vector tracing method

This algorithm is to locate each corner (or node) within given tolerance. The algorithm is described as follows:

Step 1. Construct the base vector \( v_{\text{base}} \) by joining the first point and the next \( Nth \) \((N=5)\) point in the double-linked list that the image processing module established.
Step 2. Trace the next $M$ ($M=5-10$) points in the list. Calculate all the $M$ orthogonal vectors to the $v_{\text{base}}$. If the last point in the list is found, create the closed edge vector by joining the first point of $v_{\text{base}}$ to the last point, and the tracing process comes to an end.

Step 3. If any two of the $M$ vectors are opposite or the length of the $Mth$ vector $v_M$ less than a threshold $\delta$ (i.e. if $|v_M|<\delta$), construct the new $v_{\text{base}}$ by joining the first point to the $Mth$ point, as shown in Figure 4.9 (a), and go back to Step 2.

![Figure 0.9: Schematic diagram of vector tracing process](image)

Step 4. Create edge vector $v$ by joining the first point of the base vector and the $Mth$ point, as shown in Figure 4.9 (b), and add $v$ to the double-linked list. Set the $Mth$ point as the first point of the next base vector, and go back to Step 1.

Figure 4.10 shows profile reconstruction of the object by means of the closed chain of vectors using VTM. Points stands for the corners while lines represents the edge vectors.

![Figure 0.10: Profile reconstruction by means of the closed chain of vectors](image)
4.3.2.2 Continuous Fuzzy Controller

Though VTM gives a good solution to construct the closed chain of edge vectors, the precision of vector tracing by means of VTM is dependent on setting of the threshold δ and the number $M$ to some extent. Fuzzy logic is based on natural language, which can model nonlinear functions of arbitrary complexity and can be built up on top of the expertise of experts. The fuzzy controller is capable of giving excellent and robust solutions to complicated systems, if adequate precise and reliable expertise has been accumulated. As far as the corner tracing is concerned, a continuous fuzzy controller is developed in this application on the basis of testing various corners and collecting enough knowledge in this regard, which covers all known possibilities of corner appearance.

To judge whether a corner appears, there are two determinative factors. One is the length of orthogonal vector $Len$ to the base vector $v_{base}$ from the $Mth$ point; the other is the distance $Prj$ from the project point which is perpendicular to the base vector to the endpoint of $v_{base}$. Node, the output of this CFC, is the membership of corner point. Figure 4.11 shows the definition of $Len$ and $Prj$.

![Figure 0.11: Definition of CFC input variables $Len$ and $Prj$](image)
The algorithm of the developed CFC is described as follows:

Step 1. Normalization of universe of discourse

For the sake of simplification, \( x, y \) and \( z \) are used to represent \( \text{Len}, \text{Prj} \) and \( \text{Node} \) respectively. Assume \( x \in [x_1, x_2], \ y \in [y_1, y_2], \) and \( z \in [z_1, z_2] \). To confine \( \text{Len}, \text{Prj} \) and \( \text{Node} \) to \([-1, 1]\), the following normalization formulae are applied.

\[
\begin{align*}
\hat{f}_{\text{Len}}(x) &= \frac{2(x-x_1)}{x_2-x_1} + 1 \\
\hat{f}_{\text{Prj}}(y) &= \frac{2(y-y_1)}{y_2-y_1} + 1 \\
\hat{f}_{\text{Node}}(z) &= \frac{2(z-z_1)}{z_2-z_1} + 1
\end{align*}
\]

(0-13)  
(0-14)  
(0-15)

Step 2. Definition of fuzzy set and membership functions

For the normalized input and output variable \( x, y, \) and \( z \), five fuzzy sets are defined as: NL, NS, Z, PS, PL, which are represented by \( A_1 \sim A_5, \ B_1 \sim B_5, \ C_1 \sim C_5 \) respectively, symmetric, and entirely-overlapped triangle membership functions. Hence, \( \forall x \in [-1,1], \ \sum \mu_{A_i} = 1 \). Figure 4.12 shows membership functions.

Figure 0.12: Triangle membership functions of \( \text{Len}, \text{Prj}, \) and \( \text{Node} \)
The five membership functions can be expressed as

Negative Large (NL): \( \mu_{NL} = \begin{cases} 1 & x \leq -1 \\ -2x - 1 & -1 < x < -0.5 \\ 0 & x \geq -0.5 \end{cases} \) (0-16)

Negative Small (NS): \( \mu_{NS} = \begin{cases} 0 & x \leq -1 \\ 2(x + 1) & -1 < x < -0.5 \\ -2x & -0.5 \leq x \leq 0 \\ 0 & x \geq 0 \end{cases} \) (0-17)

Zero (Z): \( \mu_{Z} = \begin{cases} 0 & x \leq -0.5 \\ 2x + 1 & -0.5 < x < 0 \\ -2x + 1 & 0 \leq x < 0.5 \\ 0 & x \geq 0.5 \end{cases} \) (0-18)

Positive Small (PS): \( \mu_{PS} = \begin{cases} 0 & x < 0 \\ 2x & 0 \leq x < 0.5 \\ -2(x - 1) & 0.5 \leq x \leq 1 \\ 0 & x > 1 \end{cases} \) (0-19)

Positive Large (PL): \( \mu_{PL} = \begin{cases} 0 & x \leq 0.5 \\ 2x - 1 & 0.5 < x < 1 \\ 1 & x \geq 1 \end{cases} \) (0-20)

Step 3. Fuzzy rule base

Table 4.1 gives the rules for fuzzy reasoning, 25 rules in all.

**Table 0.1: Rules for fuzzy reasoning**

<table>
<thead>
<tr>
<th>Len</th>
<th>Proj</th>
<th>NL</th>
<th>NS</th>
<th>Z</th>
<th>PS</th>
<th>PL</th>
</tr>
</thead>
<tbody>
<tr>
<td>NL</td>
<td>PL</td>
<td>PL</td>
<td>PS</td>
<td>PS</td>
<td>PL</td>
<td>PL</td>
</tr>
<tr>
<td>NS</td>
<td>PS</td>
<td>PS</td>
<td>PS</td>
<td>PS</td>
<td>PL</td>
<td>PL</td>
</tr>
<tr>
<td>Z</td>
<td>Z</td>
<td>Z</td>
<td>Z</td>
<td>Z</td>
<td>PL</td>
<td>PL</td>
</tr>
<tr>
<td>PS</td>
<td>NL</td>
<td>NS</td>
<td>NS</td>
<td>NS</td>
<td>NS</td>
<td>PS</td>
</tr>
<tr>
<td>PL</td>
<td>NL</td>
<td>NL</td>
<td>Z</td>
<td>NS</td>
<td>NS</td>
<td>NS</td>
</tr>
</tbody>
</table>
Step 4. Fuzzy reasoning methods

According to Table 4.1, the fuzzy rule set can be described as

Rule 1 (R₁): IF \( x = NL \) and \( y = NL \), THEN \( z = PL \)

Rule 2 (R₂): IF \( x = NS \) and \( y = NL \), THEN \( z = PS \)

......

Rule 24 (R₂₄): IF \( x = PS \) and \( y = PL \), THEN \( z = NS \)

Rule 25 (R₂₅): IF \( x = PL \) and \( y = PL \), THEN \( z = NS \)

In terms of fuzzy logic theory [45], we have the following relations

\[
R = \bigcup_{i=1}^{n} R_i \quad (0\text{-}21)
\]

\[
\mu_{R_i} = \min\{\mu_{A_i}(x), \mu_{B_i}(x), \mu_{C_i}(x)\} \quad (0\text{-}22)
\]

\[
\mu_{C_i} = \max\{\min\{\mu_{A \times B}(x, y), \mu_{R}(x, y, z)\} \quad (0\text{-}23)
\]

If the input is singleton, we have the following consequents

\[
\mu_{\text{con}}(z) = \bigcup_{i=1}^{n} \alpha_i \wedge \mu_{\text{con}}(z) \quad (0\text{-}24)
\]

Where the firing strength \( \alpha_i = \min\{\mu_{A_i}(x_0), \mu_{B_i}(y_0)\} \).

Step 5. Defuzzification

Weighted Mean Method (WMM) is used in this study. The result of defuzzification \( z_0 \) is defined as:
\[ z_0 = \sum_{i=1}^{n} \alpha_i z_i / \sum_{i=1}^{n} \alpha_i \quad (0-25) \]

In this application, the proposed CFC gives a good solution to edge vector tracing and yielding the chain of vectors. In particular, curves can be approximated with adequate precision.

Figure 4.13 shows the closed chain of vectors by means of CFC. A portion of arc in the profile is approximated by small line segments. And the profile is reconstructed by the chain of vectors with precision.

![Figure 0.13: Closed chain of vectors by means of CFC](image)

4.4 Object Classification

In terms of the configuration of the three-fingered gripper, the geometry ID must be figured out so as to make appropriate gripping planning. In this study, syntactical pattern recognition is utilized to identify the object geometry. Geometry classification is shown in Figure 4.14.
4.4.1 Determination of Object Geometry ID

The configuration of the three-fingered gripper is determined by geometry. In section 2.2.4, triangle classification by means of fuzzy set theory has been discussed. Identification of edges can be achieved by counting the edges which satisfy certain criteria. To figure out the geometry ID, calculation of angle between two adjacent vectors must be done, as shown in Figure 4.15.

\[ \theta \]

Figure 0.14: Object classification

Figure 0.15: Calculation of angle between two adjacent vectors
Where $v^*_i$ is the opposite vector of $v_i$. The angle between the two adjacent vectors $v_i$ and $v_{i+1}$, can be figured out through Equation 4-9. The geometry that is suitable for grasp are discussed as follows:

- **Profile with four edges**

As gripping oriented classification, only rectangle and those with two parallel edges may be gripped with regard to the configuration of the three-fingered gripper. Rectangle can be defined as $A_1=A_3$, $A_2=A_4$, and $A_i \perp A_{i+1}$, as shown in Figure 4.16.

![Figure 0.16: Profile with four edges](image)

- **Hexagon**

Hexagon can be defined as $A_1=A_2=A_3=A_4=A_5=A_6$, and the angle between $A_1$ and $A_2$ is $60^\circ$, as shown in Figure 4.17.

![Figure 0.17: Profile with six edges](image)

- **Profiles with two-parallel edges**

This type of geometry can be defined any profile with two parallel edges $A_1 \parallel A_2$, as shown in Figure 4.17.
4.4.2 Calculation of Object Orientation

To obtain the location of an object, apart from its position COG, its orientation must be determined as well. As to the grasp of isosceles-triangle, equilateral-triangle, rectangle, and hexagon, orientation must be worked out in terms of the chain of vectors. Figure 4.16 shows the definition of the abovementioned geometry.

- The orientation $v_0$ of isosceles-triangle is through the COG and perpendicular to the bottom side.
- The orientation $v_0$ of equilateral-triangle is through the COG and perpendicular to the side and constitutes the minimum angle with the horizontal vector $v_H$.
- The orientation $v_0$ of rectangle is through the COG and perpendicular to the longest side and constitutes the minimum angle with the horizontal vector $v_H$.
- The orientation $v_0$ of hexagon is through the COG and perpendicular to the side and constitutes the minimum angle with the horizontal vector $v_H$. 
4.5 Conclusion

Vision system plays a crucial part in machine intelligence. In this chapter, gripping-oriented vision system is described. In general, vision systems are computationally intensive application. Real-time image processing is dependent on the relevant algorithms to a large extent. Extraction and representation of object features are the key issues to which a gripping oriented application must find solution. Since boundaries can supply adequate information for object description and representation, this application, as an intelligent gripping system, takes advantage of boundary information by means of chain code, and furthermore object geometric features are extracted from the chain code.

In this study, the profile is represented by EVR, that is to say a closed chain of vectors is exploited to represent the object profile. It has been proven that the chain of vectors can represent different objects appropriately. Moreover EVR is capable of providing necessary information for object geometry reconstruction which is necessary for gripping planning. CFC and VTM are developed to achieve this task.

In this application, the developed CFC finds a good solution to edge vector tracing and yielding the chain of vectors robustly. The profile is approximated and represented by a chain of vectors.

On the other hand, the developed VTM traces the chain code and checks out the corner points heuristically. And vector algebra is a logic and effective tool in this case.
Chapter 5  Model-Based Gripping Planning

Gripping planning provides a bridge that connects perception to action. The focus of gripping planning is “how to grasp” and “where to grasp”. Human hand grasping is the best example for gripping. An object representation suited for visually guided grasping has to integrate 2D visual features and 3D grip information about a known object, in order to apply a known grip when the situation requires it. Gripping planning is highly gripper configuration related. Various types of grippers mounted on different types of robots can be found. From simplest two-fingered grippers, to dexterous hands, as shown in Figure 2.18, to anthropomorphic robot system [91], gripping planning and corresponding algorithms are confined to each instance. But as to a certain gripper system, a relatively generic algorithm and modeling may help create the effective grasp model to a geometry family to some extent. In most intelligent gripping systems [92][93], certain interaction is needed whenever a novel object comes on the scene. To perform the grasp without detailed mathematical models, neural networks are used as a new set of biologically-inspired tools for the formulation of control strategies for learning mappings from perception to action directly [94].

Current eye-hand systems extract features of the object to be grasped by means of sensing and perception, no priori knowledge on this object is needed. Intelligent gripping system may also extract features which is gripping oriented from CAD models. These sort of systems are in conjunction with a database which stores geometric feature models and relevant grasp models. And the geometric features extracted from sensory data are matched to a specific model in the database.
In this research, model-based gripping planning algorithm is investigated in order that the grasp models can be derived from geometric features generated in robot vision module, provided that the object falls into a distinguished geometry category.

5.1 Object Grasp Modeling

Knowledge-based object grasp modeling depends on knowledge from three sources:

- The geometry of object to be grasped
- The gripper geometry and configuration
- The task to be performed

Characteristics that influence gripping planning come from two aspects:

- Physical characteristics — Weight, material, surface roughness and inertia
- Geometric characteristics — Shape, faces and COG

Physical characteristics influence the grasp force that the gripper should apply to the object through weight and friction coefficient and speed at which the object can be moved. Geometric characteristics influence the gripper configuration, grasp faces, as well as grasp position [95]. In this study, we focus on the geometric features of the object. Grasping by robot grippers requires that object geometry and hand geometry be appropriately matched and the desired grasp modes selected. The following general criteria on grasp stability and manipulability must be met when gripping planning is performed:

- Maximize the contact between gripper and object
• Keep COG within stable region
• Minimize the gripping force by optimizing the selection of gripping faces and contact area
• Minimize the torque and moments that exert on the gripper and robot wrist
• Minimize the deformation of object that may arise in the process of grasp
• Approach the main grasp face at normal direction to the face
• Check out any geometric interference while gripping and releasing the object.

Different grasp mode may be derived from the strategy that is applied to satisfy the above criteria. To facilitate the description and discussion of grasp modeling, we define some items. Grasp face and featured planar face pair are introduced here for this purpose. Grasp face is defined as all faces which are suitable for grasping; Featured planar face pair is defined as planar grasp face pair in possession of two opposite face normal vectors and satisfying grasp constraints, excluding curved surface (cylinder, sphere) and hexagon. The geometry that is recognized and classified in robot vision module is 2D image (top view). The corresponding distinguished objects in 3D space are cylinder, sphere, wedge with isosceles or equilateral triangle profile from the top view, block, Parallelepiped, and polyhedron which has at least two parallel faces with opposite normal vectors. A distinguished shape falls into one of these geometry families. Gripping schemes for each family are predefined. The grasp planner for each family is indexed in terms of the geometry ID produced in vision system.

In this study, a grasp model comprises one primary grasp face and one/two secondary grasp faces with reference to the configuration of the three-fingered gripper. The three-fingered gripper has one flat finger on which the tactile sensor is mounted, known as primary finger
and two round fingers, known as secondary fingers. The grasp model is constituted by extracting object grasp features from object geometric features. Knowledge base defines gripping related features. IF-THEN inference mechanism can be employed to explain the grasp models. For instance,

![Flow chart of inference of grasp model](image-url)

**Figure 0.1: Flow chart of inference of grasp model**
IF Geometry ID = 1 THEN round object grasp model should be applied.

Procedures for inference of grasp model is shown in Figure 5.1.

As to the inference of grasp model which is capable of being applied to polyhedron, first, build up a set which encompasses all featured planar face pairs, then work out the optimal featured planar face pair. The procedures are described as follows:

Step 1. Traverse the closed chain of edge vectors, figure out all featured planar face pairs $C_i$ ($i=1, 2, ..., n$)

Step 2. Run inference routine. Hence reason out the optimal featured planar face pair

The optimal pair of faces which means COG is within the stable region, all constraints are satisfied, no geometric interference would occur during handling process, minimal gripping force is applied and object equilibrium must be maintained in the whole gripping process.

5.2 Extraction of Grasp Features

In most cases there are some sorts of uncertainty about grasp faces. Extraction of grasp faces or featured planar face pairs is the key issue of this section. Object grasp models will be constructed on the basis of geometry of the object. The geometric features of object will be constructed and incorporated into the grasp model to form the necessary grasp features. In general, the three fingertips are positioned at 20mm below the top surface in order that enough contact length or area is ensured. The fingertip contact positions are crucial to gripping planning. The generalized methodology of model-based gripping planning may be derived from the grasp features.
5.2.1 Grasp Feature of Round Objects

Round object is defined as an object with circle profile from the top view, including cylinder and sphere. The COG of the object gives the position of grasp centre of the gripper. The fingertip contact circle is the most important grasp feature in this case. The grasp features for cylinder and sphere are shown in Figure 5.2 (a) and (b) respectively.

![Figure 0.2: Grasp features for cylinder and sphere](image)

5.2.2 Grasp Feature of Wedge

Wedge is defined as an object with triangle profile from the top view, including isosceles triangle and equilateral triangle. The fingertip contact positions should be constructed in terms of the profile. The grasp features for aforementioned triangles are shown in Figures 5.3 (a) and (b) respectively.

![Figure 0.3: Grasp feature for wedges](image)
The two triangles expressed by long dash-dot-dot lines, as well as the dimensions of edges are the grasp features. And the three grasp points will be worked out accordingly in Section 5.3.

### 5.2.3 Grasp Feature of Block

Block is defined as an object with its profile being a rectangle from the top view. The fingertip contact positions should be constructed with regard to the featured planar face pair with the longest edges. The grasp feature is shown in Figure 5.4.

![Figure 0.4: Grasp feature for block](image)

The three grasp points will be worked out with regard to the grasp features expressed by the two long dash-dot-dot lines and the distance $d$ between these two lines in Section 5.3.

### 5.2.4 Grasp Feature of Parallelepiped

Parallelepiped is defined as an object with hexagon profile from the top view. The fingertip contact positions should be constructed in terms of the profile. The grasp features are the constructed hexagon and its dimension, as shown in Figure 5.5.

![Figure 5.5: Grasp feature for parallelepiped](image)
5.2.5 Grasp Feature of Polyhedron

Polyhedron is defined as an object with two parallel edges in its profile from the top view. The fingertip contact positions should be constructed in terms of the profile. The grasp feature is shown in Figure 5.6.

![Figure 0.6: Grasp feature for Polyhedron](image)

The three grasp points will be worked out with regard to the grasp features described by two long dash-dot-dot lines and the distance $d$ between these two lines, as well as the dimensions of the two lines, in Section 5.3.

5.3 Model-based Gripping Planning

Detailed grasping modeling is discussed in this section. The position and orientation of the three-fingered gripper are completely determined by the configuration of the gripper which is represented by three finger vectors (primary finger vector $v_m$, and other two fingers vector $v_{g1}$, $v_{g2}$), grasp radius $r_g$, and grasp centre $C_g$. Grasp models may result from the grasp features
described in the previous section. When object geometry matches a predefined grasp model, the aforementioned configuration of the gripper is derived. With reference to the design of the three-fingered gripper, the grasp radius ranges from 22mm through 55mm, the following results are suitable to all grasp models.

- Any angle between two finger vectors is 120°
- \(|v_m| = r_g|v_{gI}| = |v_{g2}| = r_g + R_f\), where \(R_f\) is the radius of either round finger
- The two round finger vectors \(v_1, v_2\) start from \(C_g\), and end at the centre of each round finger respectively

5.3.1 Gripping Planning for Round Object

The grasp centre \(C_g\) coincides with the centre of the object, while the primary finger vector \(v_m\) points to the east. Figure 5.7 depicts configuration of the gripper while gripping round object.

![Figure 0.7: Configuration of the gripper while gripping round object](image)

Where \(r_g\) is equal to the radius of fingertip contact circle, as shown in Figure 5.2.
5.3.2 Gripping Planning for Wedge Object

- If the profile is an equilateral triangle, $C_g$ coincides with the COG of the object. The configuration of the gripper is shown in Figure 5.8.

![Figure 5.8: Configuration of the gripper while gripping wedge with profile of equilateral triangle](image)

Where $r_g = \frac{a}{2\sqrt{3}}$, the direction of $v_m$ coincides with the orientation of the object which is determined in object recognition module. And $v_1, v_2,$ and $v_3$ are edge vectors of the object.

- If the profile is an isosceles triangle. The configuration of the gripper is shown in Figure 5.9.

We can obtain the vector $v''$ which is orthogonal to the major vector $v_1$, $v'' \perp v'$, where $v' = v_1 / 2$. On the other hand, we can acquire the bisector vector $v_b$ through $v_1$ and $v_3$. Then we work out the intersection point between $v''$ and $v_b$, which also serves as the grasp centre $C_g$. $v_m$ is constituted by creating an orthogonal vector from $C_g$ to $v_1$. $r_g = |v_m|$. 
5.3.3 Gripping Planning for Block Object

Figure 5.10 depicts configuration of the gripper while gripping block object.

In Figure 5.10, we have $H = (r_g + R_f) \sin 30^\circ - r_g + R_f$.

Figure 0.9: Configuration of the gripper while gripping wedge with profile of isosceles triangle

Figure 0.10: Configuration of the gripper while gripping block object
\[ r_g = \frac{2H}{3} + \frac{R_f}{3} \]  \hspace{1cm} (0-1)

The corresponding pitch of two round fingers \( L \) in this case can be worked out as follows:

\[ L = 2(r_g + R_f) \cos 30^\circ \]

\[ L = (r_g + R_f)\sqrt{3} \]  \hspace{1cm} (0-2)

The minimum width \( \text{min}L \) of the object may be derived from Equation 5-1 and Equation 5-2.

\[ \text{min}L = (H + 2R_f) \frac{2\sqrt{3}}{3} \]  \hspace{1cm} (0-3)

### 5.3.4 Gripping Planning for Parallelepiped Object

Figure 5.11 depicts configuration of the gripper while gripping Parallelepiped object.

![Configuration of the gripper while gripping Parallelepiped object](image)

**Figure 0.11: Configuration of the gripper while gripping Parallelepiped object**

The grasp centre \( C_g \) coincides with the COG of the object. While the primary finger vector \( v_m \) coincides with the orientation, \( |r_g| \) is equal to the length of the edge.
5.3.5 Gripping Planning for Polyhedron Object

Figure 5.11 depicts configuration of the gripper while gripping Parallelepiped object.

![Figure 5.11: Configuration of the gripper while gripping Parallelepiped object](image)

Figure 0.12: Configuration of the gripper while gripping Polyhedron object

The computation of the configuration of the gripper in this case is illustrated in Figure 5.13.

![Figure 5.13: Computation diagram for gripping planning for two parallel edges](image)

Figure 0.13: Computation diagram for gripping planning for two parallel edges

Where $\mathbf{v}_s$ denotes the projection vector, while projecting $\mathbf{v}_I$ to $\mathbf{v}_2$. The primary finger always contacts the shorter edge $\mathbf{v}_I$, while the other two round fingers contact the longer edge $\mathbf{v}_2$. According to Figure 5.13, we have the following equation
\[ d_i + x = d_r + l_s - x - l_t \]

\[ x = (d_r - d_i + l_s - l_t) / 2 \]  \hspace{1cm} (0-4)

Since \( 0 \leq x \leq l_s - l_t \), IF \( x < 0 \) THEN \( x = 0 \).

ELSE IF \( x > l_s - l_t \) THEN \( x = l_s - l_t \)

Then we can figure out the dimension \( l_L \) from left side of the primary finger to the start point of \( v_2 \), and the dimension \( l_R \) from right side of the primary finger to the end point of \( v_2 \).

\[ \begin{cases} 
    l_L = d_i + x \\
    l_R = d_r + l_s - x - l_t 
\end{cases} \]  \hspace{1cm} (0-5)

Then we may work out the mid point \( P_t \) of the primary finger, which also serves as the end point of \( v_m \).

\[ P_t = d_i + x + \frac{l_L}{2} \]  \hspace{1cm} (0-6)

The length of \( v_m \), which is \( v_m \) is orthogonal to \( v_1 \), may be worked out by means of Equation 5-1. And \( C_g \) is the start point of \( v_m \). Therefore, the configuration of the gripper is determined completely.

Whether two parallel vectors \( v_1, v_2 \) can be used as a candidate for gripping or not, is another problem to which appropriate solution must be given. The following conditions must be satisfied:

- First of all, \( |v_1| \geq \text{minMajorEdgeLen} \), where \( \text{minMajorEdgeLen} \) denotes the minimum length of the major edge.
• Secondly, the length of projection vector $|v_i| \geq \text{minMajorEdgeLen}$

• The pitch $L$ of the two round fingers is $L = 2 \cdot \min\{l_L, l_R\} + l_t$. Hence the minimum length of $v_2$ is

$$\min L = 2 \cdot \min\{l_L, l_R\} + l_t \quad \text{(0-7)}$$

When the configuration is determined, the grasp stability will be examined. That is to say, the COG must be within the stable region, which is defined as a rectangle region $ABCD$, as shown in Figure 5.12. Otherwise certain a adjustment needs to be made to keep the COG within the stable region, with the abovementioned conditions still being met.

The three finger vectors are used to check out any possible geometric interference between the gripper and the object. A number of vectors are constituted in terms of the finger vectors $v_m$, $v_L$, $v_R$, then the finger vectors are extended at the normal directions pointing to the exterior of the object. If any intersection points, between these vectors and the edge vectors of the object, are checked out, that means certain interference exists. As a result, gripping planning should be remade, i.e. another featured planar face pair is examined, until no interference exists. Grasp model created for a polyhedron is shown in Figure 5.14.

![](image.png)
The gripping planning process is fully automated, no interaction is required during the process. The result of gripping planning is visualized by illustrating the three finger positions and orientations and grasp centre. Therefore, certain verification can be done through the aforementioned visualization.

5.4 Conclusion

Eye-hand systems extract grasp-related geometric features of the object to be grasped by means of sensing and successive perception. Model-based gripping planning is an effective solution to build up the grasp model. No priori knowledge on this object is needed. It’s a significant exploration and attempt to solve the gripping planning problem intelligently.

The proposed algorithm succeeds in generating the grasp model by extracting the geometric features coming from vision system and reasoning out the grasp features from the geometric features. The constraint examiner and interference examiner ensure that the gripping process will be performed reliably and robustly. In addition, visualized gripping planning model can be used as a complement to the interference examiner. The five geometry categories, which can be handled by this application, cover basic part geometry to some extent. The proposed system may find application in automatic assembly or component handling system.
Chapter 6  Trajectory Planning and Intelligent Gripping

Integrated intelligent gripping system combines the robot system with the gripper system. To incorporate gripping control system into robot system, the communication between the host computer and robot controller is supposed to be established. In integrated robot system, computers can be implemented to directly control machines via a machine controller. This implies that communication among machines can be handled by the computer level. Since most type of industrial robots support serial communication or Ethernet communication and digital or analog I/O as well, at present, industrial robot-based gripping/handling system usually make use of serial communication or TCP/IP-based network communication (LAN or internet). Figure 6.1 shows robot communication protocols of ABB robots.

![Diagram of robot communication protocols]

**Figure 0.1: S4 robot communication protocols**

Robot intelligence is significantly related to sensing system. F/T sensor, proximity sensor, tactile sensor, and vision system endow robot system with the ability to adapt to the environments. Robot trajectory planning is dependent on robot manipulator, gripper configuration and the object to be grasped. Gripping process modeling may extract object
grasp-related data either from gripping planning system or directly from robot workcell simulation system 0. In this research, gripping planning module offers necessary data for trajectory planning. The coordinate frame relationship between robot system and vision system must be created appropriately such that the position and orientation data may be transformed from vision coordinate frame to relevant robot coordinate frame. The robot paths generated automatically is downloaded to the robot controller through Ethernet. S4 controller is present in the ABB IRB1400 industrial robot. ActiveX technique facilitates the communication between the host computer and the robot controller.

6.1 Eye-hand Coordinate System Transformation

The relationship between vision system and robot system is crucial to eye-hand robot systems. Transformation matrices are powerful tools to establish this relationship which coordinates the vision system with robot manipulator. Gripping positioning accuracy is dependent on the aforementioned relationship. The robot platform employed in this research, an articulated robot manipulator, has six degrees of freedom (DOF). Figure 6.2 shows the robot manipulator and its axes.

6.1.1 Robot Coordinate Systems

All robot positions are related to a defined coordinate frame. This coordinate frame may in turn be related to another coordinate frame etc. in a chain. Some of these coordinate frames are embedded in the configuration of the robot system, and are not visible to the user, while others may be programmed by the user. The coordinate frame chain that the ABB series robots use is shown in Figure 6.3.
Figure 0.2: IRB1400 robot manipulator and its axes

Figure 0.3: Coordinate frame chain

- **Base coordinate system** — gives relation between world and base frame, and is related to world frame. It is usually attached to the robot base.

- **World coordinate system** — is intrinsic to the system, no definition is needed.
• *User coordinate system* — can be defined anywhere on the scene, and is related to world coordinate system.

• *Object coordinated system* — is usually attached to an object, and is related to user coordinate system.

• *Wrist coordinate system* — is implicit in the kinematic model of robot, and is related to base coordinate system.

• *Tool coordinate system* — is defined by means of tool calibration data, and is related to wrist coordinate system.

• *Program displacement coordinate system* — is set up by robot instructions in RAPID program, and is related to object coordinate system.

![Coordinate system relationships](image)

6.1.2 Transformation Matrices and Quaternion

The relationship between the tool coordinate frame and vision coordinate frame has direct effect on the robot programming. In robot program, robot paths (referred to as *robtarget* in ABB robot RAPID language) are TCP locations (i.e. the tool frame, referred to as *tooldata* in RAPID language) with reference to the object coordinate frame. The coordinate system relationships in intelligent gripping system are shown in Figure 6.4.
In this project we suppose that the desktop of the workbench where the object frame is located is parallel to the plane on which the robot base is mounted, i.e. the user frame or object frame, and vision frame are on the same plane, and parallel to the robot base frame.

To simplify programming, we have the user frame $O_uX_uY_uZ_u$ coincide with vision system $O_vX_vY_vZ_v$. The object frame $O_cX_oY_oZ_o$ is constructed with its $X$ axis $O_cX_o$ collinear to the object orientation vector.

Suppose that a rotation of $\phi$ angle about $O_uY_u$ axis followed by a rotation of $\theta$ angle about the $O_uZ_u$ axis followed by a rotation of $\alpha$ angle about the $O_uX_u$ transform the gripper frame $O_gX_gY_gZ_g$ to the user frame $O_uX_uY_uZ_u$, then the resultant rotation matrix representing these rotations can be represented by

$$ R = R_{\alpha,\phi} R_{\theta,\phi} R_{\gamma,\theta} = \begin{bmatrix} C\theta C\phi & -S\theta & C\theta S\phi \\ C\alpha S\theta C\phi + S\alpha S\phi & C\alpha C\theta & C\alpha S\theta S\phi - S\alpha C\phi \\ S\alpha S\theta C\phi - C\alpha S\phi & S\alpha C\theta & S\alpha S\theta S\phi + C\alpha C\phi \end{bmatrix} \quad (0-1) $$
where $C$ and $S$ stand for trigonometric functions $Cos$ and $Sin$ respectively.

### 6.1.3 Quaternion

The orientation of a coordinate system can be described by a rotational matrix, as aforementioned in previous section, which describes the direction of the axes of the coordinate system in relation to a reference system. The rotated coordinate systems axes ($x$, $y$, $z$) are vectors which can be expressed in the reference coordinate system as follows:

$$
\begin{align*}
x &= (x_1, x_2, x_3) \\
y &= (y_1, y_2, y_3) \\
z &= (z_1, z_2, z_3)
\end{align*}
$$

These three vectors can be put together in a matrix, a rotational matrix, where each of the vectors form one of the column:

$$
\begin{bmatrix}
x_1 & y_1 & z_1 \\
x_2 & y_2 & z_2 \\
x_3 & y_3 & z_3
\end{bmatrix}
$$

In ABB robot RAPID language, quaternion is used to represent rotational matrix. A quaternion is a more concise way to describe this rotational matrix; the quaternion is calculated based on the elements of the rotational matrix:

$$
\begin{align*}
q_1 &= \frac{\sqrt{x_1 + y_2 + z_3 + 1}}{2} \\
q_2 &= \frac{\sqrt{x_1 - y_2 - z_3 + 1}}{2} \\
q_3 &= \frac{\sqrt{y_2 - x_1 - z_3 + 1}}{2}
\end{align*}
$$

$$
\begin{align*}
\text{sign } q_2 &= \text{sign } (y_3 - z_2) \\
\text{sign } q_3 &= \text{sign } (z_1 - x_3)
\end{align*}
$$
\[ q_4 = \frac{\sqrt{z_3 - x_1 - y_2} + 1}{2} \] \[ \text{sign } q_4 = \text{sign} \ (x_2 - y_1) \] (0-5)

6.1.4 Computation of Object Frame

Since the z axis of the gripper frame (the tool frame) \( O_g Z_g \) is opposite to the z axis of the user frame \( O_u Z_u \), we have \( \phi = -180^\circ, \alpha = 0 \). Hence, we may work out the quaternion in this case.

\[
\begin{aligned}
q_1 &= 0 \\
q_2 &= \frac{\sqrt{2}}{2} \sqrt{1 - \cos \theta} \\
q_3 &= \frac{\sqrt{2}}{2} \sqrt{1 + \cos \theta} \\
q_4 &= 0
\end{aligned}
\] (0-6)

According to \( q_2 = \frac{\sqrt{2}}{2} \sqrt{1 - \cos \theta} \), we may derive the rotation angle

\[ \theta = \cos^{-1} (1 - 2q_2^2) \] (0-7)

Besides, the angle \( \beta \) between the X axis of object frame \( O_u Z_u \) may be figured out through the orientation vector of the object to be grasped. Thereby, the object frame is obtained by translating the user frame to the grasp centre \( (x_{gc}, y_{gc}) \) of the object and rotating Z axis by \( \beta \) angle. The following transformation matrix may be used to express the coordinate transformation.

\[
\begin{bmatrix}
x_o \\
y_o
\end{bmatrix} = \begin{bmatrix}
\cos \beta & -\sin \beta \\
\sin \beta & \cos \beta
\end{bmatrix} \begin{bmatrix}
x_u \\
y_u
\end{bmatrix} + \begin{bmatrix}
x_{gc} \\
y_{gc}
\end{bmatrix}
\] (0-8)

6.2 Robot Trajectory Planning

In this study, the task to handle the object is pick it up and place it in the right place in terms of its geometry ID. In this regard, robot program is supposed to be generated automatically.

clxiv
and downloaded to the robot controller through Ethernet. The intelligent gripping process is performed by coordinating the gripper control and execution of robot program in terms of sensory data through communication between the host computer and robot controller.

6.2.1 RAPID Program Structure

A RAPID program consists of instructions and data. The program is usually made up of three different parts:

- A main routine
- Several subroutines
- Program data

The program memory contains system modules. The main routine is the routine from which program execution starts. Subroutines are used to divide the program up into smaller parts in order to obtain a modular program that is easy to read and maintain. Data is used to define positions, numeric values (registers, counters) and coordinate systems, etc. The structure of RAPID program is shown in Figure 6.5.
Figure 0.5: Structure of RAPID program

System modules are programs that are always present in the memory. Routines and data related to the installation rather than the program, such as tools and service routines, are stored in system modules. The detailed RAPID program generated in this study is referred to Appendix D.

6.2.2 Robot Manipulator Path Planning

In ABB RAPID programming, the tool frame is specified by data structure tooldata; user and object frame are specified by data structure wobjdata; while the manipulator target position is specified by data structure robtarget. tooldata, wobjdata, and rottarget must be specified explicitly in each movement instruction. TCP of this gripper is defined at the centre of circle that the three fingertips make up, as shown in Figure 6.6

Figure 0.6: Definition of TCP for the three-fingered gripper

Here, tool frame is defined using 4-point TCP method [73]. tooldata is determined in terms of the tool frame. wobjdata is constructed with regard to the relationship between vision frame and user frame/object frame. robtarget for each movement instruction is accomplished in terms of result of object recognition and gripping planning.
Certain compensation is needed to keep the gripper right perpendicular to the workbench. Compensation method is worked out by figuring out the compensation vector in $XOY$ plane by means of calibration. Then the compensation vector is projected to $X$ and $Y$ axis respectively to get each component, which will be used to calculate the quaternion at each position on the paths.

6.3 Sensory Data Based Gripping Control and Analysis

The intelligent gripping system makes use of a hierarchical architecture that comprises three levels which encompass planning level, decision-making level, and implementation level (sensing, actuating). Control strategy is implemented through PC bus interfaces.

Planning level generates the manipulator’s path and control strategy which handles communication and interaction between the host computer, robot controller, and actuator. When gripping process is carrying on, decision-making level conducts program flow and control flow, while both tactile image and actuator status, and information from robot controller are analyzed and control strategy is deduced with regard to this information. Gripping force is adjusted by analyzing tactile image in order that any possible slipping between the object and the fingers can be prevented.

6.3.1 Tactile Image Analysis and Gripping Stability

Because of the limited accuracy, speed of visual feedback, stiffness of surfaces that the gripper fingers will contact, and the need to measure deformations, vibrations, slipping and forces during contact with objects, it is essential to endow the service robot with an action-perception cycle with tactile feedback. This tactile information has to be processed in a short period to
protect the object or the manipulator against damage. The tactile sensor employed in this study is an 8×8 array capacitive sensor. It is used to detect the spatial and temporal pressure distribution. The goal of the tactile gripping skill is to successfully apply a stable grasp on an unknown object.

In a real-time gripping system, the tactile image must be processed in a significantly short time so as to react to the touch state. In this application, multithreaded programming technique is utilized to ensure real-time processing of touch information. Tactile image is built up during the whole object handling process in through a thread execution. And the tactile image is updated in a certain period of time by means of communication with the thread data. In general, contact process with the grasp surface of the object is divided into two stages:

- **Transitional grasp stage** — the touch area with the surface of the object gets increasingly large. Thus, the amount of touched elements increases. And force exerted on the touched element also increases.

- **Stable grasp stage** — the touch area and the amount of touched elements are stabilized. The gripping force varies in a narrow range.

If some sort of singularity occurs during the gripping process, for instance, the amount of touched elements or the gripping force changes dramatically, that may result from slipping between the finger and the object being gripped. Then it leads to more force needed to be applied. The digital I/O communication between the host computer and robot controller is shown in Figure 6.7. The gripping process control is shown in Figure 6.7.
Figure 0.7: Digital I/O Communication between the host computer and robot controller

Figure 0.8: Gripping process control flow chart
6.4 Conclusion

Vision-guided trajectory planning generates robot RAPID programs in terms of data from gripping planning, kinematics of robot manipulator, and transformation matrices between relevant robot coordinate systems and vision coordinate system. Practical equations that build up the relationship between robot tool frame (represented by TCP of the three-fingered gripper) and object frame have been derived from the transformation matrices. An appropriate quaternion computation method derived in this project guarantees automated generation of robot programs. On this basis, modular RAPID programs are successfully generated and tested with regard to different shape of objects.

Digital I/O is employed as handshaking signal between execution of robot program and manipulation of the three-fingered gripper so as to coordinate these two relatively independent processes. Analysis of tactile image ensures a stable and robust gripping process. Singularity in the process of gripping may be handled, and proper strategy may be applied to solve the problem at the same time. Besides, compensation solution to errors of gripper and robot has been made to assure precise gripping process.
Chapter 7  Conclusion

Investigation into intelligent gripping system based on various types of industrial robots has been carried out since the last twenty years. Further research is required to improve the system performance and integration with other technologies. The significance of implementing intelligent gripping is that it enables an industrial robot, which usually executes preprogrammed programs, to handle singularity or abnormal conditions with operator flexibilities. Sophisticated sensing mechanism allows the robot to respond to its environment in an intelligent and flexible manner, in order to meet the requirements of flexible automation. The intelligence of a robot is realized by extracting, analyzing, and modeling of sensory data with the view of deriving effective decision-making schemes. Intelligent gripping systems may find a wide range of application within assembly process, production line and quality inspections. The computer integrated and reconfigurable intelligent gripping system developed in this project presented an advanced platform to perform research into robot vision and intelligence.

In this research project, an intelligent gripping system was integrated with an ABB industrial robot. Hardware components (servo actuator, sensors, robot and interface boards) and software components were used to perform the integration task in a modular fashion. Modular system hardware formation and software design assures reconfigurability of the system. The subsystems are fully functional and totally integrated. This minimize the cost and complexity of the use and maintenance of the system, it is able to adapt to different environments with ease, and facilitate its use in other applications. In addition, PC-bus interface and ActiveX techniques guarantee the applicability and compatibility of the system.
Hierarchical architecture for intelligent gripping is constituted, which partitions the robot’s functionalities into high-level (modeling, recognizing, planning and perception) layers, and low-level (sensing, interfacing and execute) layers that run asynchronously.

A robot vision system makes up of the most important portion of robot intelligence. In-process algorithms in vision systems require high computational efficiency in order to be applied to a real-time system. Object boundaries provide adequate gripping-oriented geometric features including the extraction, description and representation of object geometric features. These features constitute the core of a vision system. In this study, the object profile is represented by EVR. It is a closed chain of vectors employed to describe the object profile. The most important thing is that EVR can provide the necessary information for gripping planning. In this research, CFC was developed to solve the problem of edge vector tracing. It is capable of yielding the closed chain of vectors robustly. VTM method was also developed to trace the chain code and determine the corner points heuristically, may be used as an alternative to CFC method. Fuzzy inference engine was proven to be an ideal solution to trace out the edge vector. In addition, algorithms with high computational speeds are crucial in a real-time robot vision system, while consistent illumination is key to robustness of an image processing system. In this study, geometric relations are extracted from the closed chain of edge vectors, useful features such as COG, area, etc. are figured out, and a syntactic pattern recognition method is utilized to reason out the geometry ID of objects.

As a bridge that connects perception to action, gripping planning solves the problem of “how to grasp” and “where to grasp”. Gripping planning is highly related to gripper configuration and object geometry. A generic gripping planning algorithm for a specific gripper system able
to generate an effective grasp model for each geometry family was developed. Present eye-hand systems extract grasp-related features of the object to be grasped by means of sensing and perception with no priori knowledge on this object required. In this project model-based gripping planning is proposed. It provides an effective solution to create the grasp model for an object which falls into one of the grasp categories, extracting the geometric features coming from the vision system and reasoning out the grasp features. It is indispensable to examine relevant geometric constraints and any possible interference so as to ensure that the gripping process will be performed reliably and safely. In this application a visualized grasp model is displayed for intuitive inspection to geometric interference.

The goal of the project is to grasp objects intelligently. Vision-guided trajectory planning algorithm is proposed. It generates robot programs using data from the gripping planning module and kinematics of robot manipulator. Transformation matrices between relevant robot coordinate systems and vision coordinate system provide an effective method to generate robot programs automatically.

System integration over the robot controller is via Ethernet communication. Digital I/Os coordinate two relatively independent processes. One is the execution of the robot program and the other is the control of the three-fingered gripper. A stable and robust gripping process may be achieved by analyzing the tactile image and adapting the grasp state accordingly. The intelligent gripping process is conducted by taking into account singularities, compensation to errors that may influence the grasp, analysis of tactile image and generating appropriate control strategy. Various experiments have been performed on the developed platform. The experimental validation of the system is proven to be correct.
Besides, this application is built in the environment of Visual C++. Visual C++ is a remarkably powerful development platform to develop system-level industrial application. Object-oriented and based on MFC class programming techniques ensure the compatibility, expandability, reconfigurability and modular programming design.

The experimental validation of the intelligent gripping system was made on a variety of distinguished objects. The experiment result is proven to be successful and fulfill the desired objectives.

The accomplishments and contributions of this study can be summarized as follows:

- An integrated intelligent robot gripping system which integrates robot vision system, gripping planning, trajectory planning, intelligent gripping control scheme and ABB IRB1400 industrial robot is established and implemented successfully.

- The developed linear servo actuator is capable of supplying a significant source for intelligent gripper. This actuator implements precise control over position and speed.

- The designed three-fingered gripper equipped with the developed linear servo actuator is highly versatile and can handle a large variety of objects effectively.

- The generic algorithms developed in the robot vision system give practical and effective solutions to machine vision. The approximation to object boundaries by means of the closed chain of edge vectors is proposed in this research. EVR may be regarded as an efficient object geometric representation for gripping-oriented application.
• The uncertainty and non-linearity in the image processing and part recognition incur difficulty in modeling the process and extracting features by means of formulae. FL, which is based on natural language, can model nonlinear functions of arbitrary complexity, and can be built up on top of the expertise of experts. FL provides us with a mathematical framework for representation and processing of expert knowledge. FL is of computational intelligence. The successful use of fuzzy logic enhances the intelligence of the system. The generic CFC method may find its use in other applications.

• The model-based gripping planning system categorizes objects and successfully creates a grasp model for different objects.

• The robot manipulator trajectory planning algorithm succeeded in generating RAPID programs automatically.

• Seamless integration between individual modules. Serial data flow supports each module effectively.

Intelligent gripping system is an advanced system. Further development may enhance the performance and functionalities of the system. Future development may be summarized as follows:

• To improve the user interface, including 3D animation of intelligent gripping process, further integration of operation of system modules is proposed.
• To increase the intelligence of the system a proximity sensor may be integrated to determine the distance to objects. Sensor fusion may improve the performance of the system.

• To develop a multi-fingered gripper with more DOF. It will handle complicated objects more effectively.

• To develop 3D vision system enabling geometric and grasp features to be extracted more effectively. It may lead to more reasonable grasp schemes. Hence, the most reasonable grasp model is reasoned out.

To take into consideration the workcell within the robot manipulator trajectory planning module, the intelligent gripping system may then include full integration with workcell.
References


clxxxii


[63] Banks J., Design and Control of an Anthropomorphic Robotic Finger with Multi-Point Tactile Sensation, Research Report, Artificial Intelligence Laboratory, MIT.


[77] Color CCD Camera Operating manual, Eagle Technology.


[88] Schildt H., MFC Programming from the Ground Up, California, McGraw-Hill.


Appendix A  Procedures for Conductiong Intelligent Gripping

To intelligently grip the object (as shown in Figure A-1) by using the developed integrated intelligent gripping system, the following procedures are supposed to be followed:

1. Initialize the vision system by clicking the icon , or select menu Image->Initialize FP3D (Ctrl+F9).

2. Calibrate the vision system by placing the Calibration Disc on the scene and clicking the icon , or selecting menu Calibration->Vision Calibration (Ctrl+F2). The Image Processing and Object Recognition Dialog, shown in Figure A-2, will appear. Accept the default settings in this dialog in most cases, and click on OK button to start calibrating the vision system. The result of vision calibration that displays in the main window of the application is shown in Figure A-3.

Figure A - 1 Object with complex shape to be grasped
Figure A - 2 Image Processing and Object Recognition Dialog

Figure A - 3 Window that displays vision system calibration result
• Enter the height of the object to be grasped in Robot Calibration Dialog, as shown in Figure A-4, by clicking the icon 🔄, or select the menu Calibration->Robot Calibration (Ctrl+F3). The default height is 40mm.

![Robot Calibration Dialog](image)

**Figure A - 4 Robot Calibration Dialog**

• Recognize the object by clicking the icon 📸, or select the menu Recognition->Recognition (Ctrl+F12). The Image Processing and Object Recognition Dialog, as shown in Figure A-2, will appear. Accept the default settings in most cases, click on OK to recognize the object. The system will display the chain of vectors in the main window of the application, as shown in Figure A-5.
Figure A - 5 The window that shows the result of object recognition

- Do Gripping Planning, after the object has been recognized successfully by clicking the icon or select the menu Gripping->Gripping Planning.

- Generate the robot trajectory for intelligently gripping this object by clicking the icon or select the menu Gripping->Create Trajectory. The dialog, as shown in Figure A-6, will appear, click on Yes to create trajectory.
Start the gripping process by clicking the icon ![icon], or select menu **Gripping->Start**. The Intelligent Gripping Process Dialog, as shown in Figure A-7, will be brought up.

Click on the **Start** button to start the intelligent gripping process.
Calibration of the gripper

The objective of calibration of the gripper is to obtain the tooldata of the gripper which is essential to generate the RAPID robot programs. The procedure of the calibration may be described as

- Fit the gripper calibration tool, as shown in Figure B-1, on the gripper. Figure B-2 shows that the calibration tool is fitted on the gripper.

Figure B - 1 Gripper calibration tool

Figure B - 2 Gripper fitted with the calibration tool
• Use 4-point TCP method [73] to calibration the TCP of the gripper.

![Figure B - 3 The 4-point TCP method](image)

**Definition of user coordinate system and object coordination system**

Figure B-4 depicts the 3-point method of defining user coordinate system and object coordination system. To simplify the definition of object coordinate system, we can make the object coordinate system coincide with the user coordinate system.

![Figure B - 4 The 3-point methods of defining user coordinate system and object coordinate system](image)
Appendix C  Configuration Files and Fuzzy Rule Database

➢ Default Configuration file of Vision System

The default configuration file (FP3d_defaultParam.fgb) of vision system is as follow

```
[m_bCompMono] = 0
[m_bReplicate] = 0
[m_nAlign] = 2
[m_nBrightness] = 42
[m_nContrast] = 42
[m_nSaturation] = 23
[m_nHue] = 0
[m_nSharpness] = 0
[m_nTimeOutMS] = 2
```

➢ Fuzzy IF-THEN Rule Database

The Microsoft Access Database (RuleSet.mdb) is used to save the fuzzy IF-THEN rule database. The definition of the data structure in design view is shown in Figure C-1.

![Figure C - 1 Definition of the data structure in design view of MS Access](image)

```
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>AutoNumber</td>
<td>The distance between the projection point of the first vector and the projection point of the last vector</td>
</tr>
<tr>
<td>LEN</td>
<td>Number</td>
<td>The length of the last vector</td>
</tr>
<tr>
<td>TURN</td>
<td>Number</td>
<td>The discriminator of turning point</td>
</tr>
</tbody>
</table>
```

The fuzzy IF-THEN rules in the database is shown in Figure C-2.
<table>
<thead>
<tr>
<th>ID</th>
<th>PRJ</th>
<th>LEN</th>
<th>TURN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-2</td>
<td>-2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>-2</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>-2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>-2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>-2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>-1</td>
<td>-2</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>-1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>-1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>-1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>-2</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>-1</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>16</td>
<td>1</td>
<td>-2</td>
<td>-2</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>18</td>
<td>1</td>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>19</td>
<td>1</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>20</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>21</td>
<td>2</td>
<td>-2</td>
<td>-2</td>
</tr>
<tr>
<td>22</td>
<td>2</td>
<td>-1</td>
<td>-2</td>
</tr>
<tr>
<td>23</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>24</td>
<td>2</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>25</td>
<td>2</td>
<td>2</td>
<td>-1</td>
</tr>
</tbody>
</table>

Figure C - 2 Fuzzy IF-THEN rules in RuleSet.mdb
Appendix D  Robot Program for Intelligent Gripping of the Object

The robot RAPID program automatically generated in the trajectory planning is saved in file (Test.prg). Test.prg is shown as follows:

```plaintext
VERSION:1
LANGUAGE:ENGLISH

MODULE MY_TEST
'15-07-2002, 09:14:58
PERS tooldata
gripper1:= [TRUE,[[0.0000,0.0000,315.0000],[1.0000,0.0000,0.0000,0.0000]],[5.0,[85.0000,0.0000,65.0000],[1.0000,0.0000,0.0000,0.0000],[0.0000,0.0000,0.0000,0.0000],[0.01,0.01,0.01]]];
PERS wobjdata wobj1 := [FALSE,TRUE,[[301.9700,-1144.6500,423.2400],[1.0000,0.0000,0.0000,0.0000]],[[0.0000,0.0000,0.0000,0.0000],[1.0000,0.0000,0.0000,0.0000]]];
VAR robtarget homepos:= [[168.2209,144.0067,199.6798],[0.0184,-0.5831,-0.8121,-0.0132],[-1.0,-1.0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]]; VAR robtarget startpos:= [[168.2209,144.0067,149.6798],[0.0184,-0.5831,-0.8121,-0.0132],[-1.0,-1.0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]]; VAR robtarget readypos:= [[168.2209,144.0067,69.6798],[0.0184,-0.5831,-0.8121,-0.0132],[-1.0,-1.0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]]; VAR robtarget rotpos:= [[168.2209,144.0067,69.6798],[0.0215,-0.7947,-0.6066,-0.0073],[-1.0,0.0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]]; VAR robtarget grippos:= [[168.2209,144.0067,49.6798],[0.0215,-0.7947,-0.6066,-0.0073],[-1.0,0.0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]]; VAR robtarget pickuppos:= [[168.2209,144.0067,69.6798],[0.0215,-0.7947,-0.6066,-0.0073],[-1.0,0.0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]]; VAR robtarget safepos:= [[168.2209,144.0067,149.6798],[0.0215,-0.7947,-0.6066,-0.0073],[-1.0,0.0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]]; PROC haupt()
  main;
ENDPROC

PROC main()
  VAR num answer;
  TPeRase;
  TPWrite "TEST PROGRAM";
  MOVEJ homepos,v100,z20,gripper1\WObj=wobj1;
  MOVEJ startpos,v80,z20,gripper1\WObj=wobj1;
  MOVEJ readypos,v20,z20,gripper1\WObj=wobj1;
  MOVEJ rotpos,v20,z20,gripper1\WObj=wobj1;
  MOVEL grippos,v5,fine,gripper1\WObj=wobj1;
  TPReadFK answer, "Gripped?","Ok", "", "", "";
  MOVEJ pickuppos,v20,z20,gripper1\WObj=wobj1;
  MOVEJ safepos,v80,z20,gripper1\WObj=wobj1;
ENDPROC
ENDMODULE
```

excvii
Appendix E  Technical Drawings for the Actuator and the Gripper

Drawing List:

- Gripper Assembly
- Actuator Assembly
- Actuator
- Housing
- Coupling
- Motor Connector
- Nut
- Leadscrew
- Shaft
- Key Base
- Anti Rot. Key
- End Cap
- Gripper Assembly
- Grip-Base
- Bracket
- Crank
- Linkage
- Connecting-Rod
- Top-Bracket
- Finger-Support
- Finger
- Finger1
- Adjust Block
- Fingertip
The technical drawings is not available see cd for drawings