Title:
Fuzzy Optimisation Based Symbolic Grounding for Service Robots
Name:
Beisheng Liu

Fuzzy Optimisation Based Symbolic Grounding for Service Robots

Beisheng Liu

Ph.D

2013

UNIVERSITY OF BEDFORDSHIRE

Fuzzy Optimisation Based Symbolic Grounding for Service Robots

by

Beisheng Liu

A thesis submitted to the University of Bedfordshire in partial fulfilment of the requirements for the degree of Doctor of Philosophy

November 2013

ABSTRACT

Symbolic grounding is a bridge between task level planning and actual robot sensing and actuation. Uncertainties raised by unstructured environments make a bottleneck for integrating traditional artificial intelligence with service robotics. In this research, a fuzzy optimisation based symbolic grounding approach is presented. This approach can handle uncertainties and helps service robots to determine the most comfortable base region for grasping objects in a fetch and carry task. Novel techniques are applied to establish fuzzy objective function, to model fuzzy constraints and to perform fuzzy optimisation. The approach does not have the short comings of others' work and the computation time is dramatically reduced in compare with other methods. The advantages of the proposed fuzzy optimisation based approach are evidenced by experiments that were undertaken in Care-O-bot 3 (COB 3) and Robot Operating System (ROS) platforms.

DECLARATION

I declare that this thesis is my own unaided work. It is being submitted for the degree of Doctor of Philosophy at the University of Bedfordshire.

It has not been submitted before for any degree or examination in any other University.

Name of candidate: BEISHENG LIU

Signature: _____

Date: 05-Nov-2013

ACKNOWLEDGMENTS

I would like to express my gratefulness and deepest respect to my directors of studies, Dr.Dayou Li, and Prof. Yong Yue for their continuous encouragement, invaluable guidance and support.This thesis will not be completed without their mentorship.

I would like to express my gratitude to my parents for all their support and encouragement throughout my PhD. None of my achievements could have happened without their support.

My special thanks are due to all my friends whose support has been invaluable.

This research is sponsored by the EU FP7 ICT project “Multi-Role Shadow Robotic System for Independent Living” (Grant agreement no.: 247772).

LIST OF CONTENT

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AI	Artificial Intelligent
ARPLACE	Action-Related Place
CHM	Constraints Handling Method
CoM	Centre of Mass
DoF	Degree of Freedom
FRS	Fuzzy Reachability Space
FRBS	Fuzzy Rule Base System
FULDEK	Fuzzy Logic Development Kit
FIS	Fuzzy Inference System
FCF	Fuzzy Constraint Functions
GSM	Grounded Situation Model
GA	Genetic Algorithm
GrAM	Grounded Action-related Model
GSM	Generalised Success Model
HDZ	High Dexterity Zone
IK	Inverse Kinematic
LS	Least-Squares
KB	Knowledge Base
MA	Manipulability Area
PDM	Point Distribution Model
ROS	Robot Operating System
STRIPS	Stanford Research Institute Problem Solver
SRS	Shadow Robotic System
SISO	single-input-single-output
SPARK	Spatial Reasoning & Knowledge
SVM	Support Vector Machine
SQP	Sequential Quadratic Programming
STA	Simple Tuning Algorithm
SMACH	Space Machine
ODE	Open Dynamic Engine
OWL	Ontology Web Language
OF	Objective Function
OBP	Optimal Base Pose
OSI	Obstacle Spatial Information
QNA	quasi-Newton Algorithm
QLFIS	Q-learning Fuzzy Inference System
QLBGFC	Q-learning Based Genetic Fuzzy Controller
VSS	Variable Structure Systems
WGR	Workspace Goad Region

CHAPTER 1 INTRODUCTION

1.1 Background and Motivations

Service robots aim to ease people's everyday life by taking over chores such as cleaning up a table or serving drinks. They are especially useful for people that depend on assistance such as elders or people with disabilities. Several service robot systems are developed in recent years for assisting people in their daily lives. The notable examples include: 1) PR2 system developed by Willow Garage for perception and object fetching applications (Bohren et al., 2011); 2) ARMAR-III system for loading and unloading a dishwasher and a refrigerator (Asfour et al., 2008); 3) Justin system for manipulating objects on tables (Fuchs et al., 2009); 4) Herb system for opening and closing doors, draws and cabinets (Srinivasa et al., 2010); 5) Care-O-bot 3 system before the SRS project for detecting and placing bottles onto a tray (Meeussen et al., 2010). The common significant to these service robot systems is their capability of implementing tasks at certain autonomous level in domestic environments.

Considerable efforts have been made to enable service robots to accomplish multiple tasks autonomously in domestic environments. The technologies involve three main aspects: 1) symbolic action planning at task level; 2) actions implementation based on specific robot configurations at motion level; 3) symbolic grounding that bridges the task level planning and the motion level control. This research focuses on the symbolic grounding for transferring symbolic action commands generated at the task level into specific robot configurations in terms of robot base poses (locations and orientations) and the optimal robot base region used at the motion level.

When a service robot is given a task command, e.g. “get a milk box”, a sequence of symbolic action commands for accomplishing the task has to be planned at the task level. Each action command in the action sequence is normally composed of an individual function such as “move” or “search”, a robot’s body part such as “base” or “arm”, a target object such as “milk box” or “kitchen table”, and a symbolic term that indicates the relational condition of the robot’s body part with respect to the target object such as “near” or “close”. For example, a symbolic action command *move(base, near, MilkBox0)* represents the action of moving a robot base to a pose near to the target object *MilkBox0* for performing tasks, such as object manipulation or surface exploration. On the other hand, at the motion level, robots are controlled based on trajectories. Trajectories are the sequences of poses in a robot’s workspace, starting from its current pose to a destination pose. The robot will be driven to move from one pose to another according to planned trajectories. It is necessary to bridge symbolic action commands at the task level and specific robot poses at the motion level to enable a robot to understand the action commands at the higher level and to execute corresponding actions at the lower level. The bridging process is known as symbolic grounding.

Grounding task level commands to motion level controls has been proven to be not easy in domestic environments since these environments are often highly unstructured. On one hand, it is not possible to have exact and complete prior knowledge of these environments. For example, the location of objects is usually unknown and grounded robot base pose may be occupied by obstacles. This information has to be obtained through perception and interaction with the environment. On the other hand, knowledge acquired through sensing is affected by uncertainties and imprecision. The quality of sensory information is influenced by sensor noise and the limited sensor range. For example, the localisation and navigation of robot base suffered from sensor imprecision. A robot cannot exactly position itself to a desired pose and it doesn’t know precisely where it is currently located.

Symbolic grounding has drawn increasing attention. Approaches reported for addressing the above mentioned problems can be classified into optimisation based approaches, analytic model based approaches, vision based approaches, statistic based approaches and experience based approaches. In optimisation based approaches, uncertainties in determining the optimal robot configuration for implementing an action command is modelled by an objective function. The objective function is often defined based on certain criteria such as grasp quality, manipulability measure of robot arm, distance to obstacles, action execution time, etc. The optimal configuration can be determined by evaluating a solution space using the objective function (Zhao et al., 1992, Hsu et al., 1999, Mitsi et al., 2008, Berenson et al., 2008). Limitations in the optimisation based approaches can be summarised as: 1) the optimal robot configuration may not be reachable since the influences of obstacles are not taken into account (Zhao et al., 1992); 2) object grasping is not considered (Hsu et al., 1999); 3) time spent for searching the solution space is too long which may not be able to meet the requirement of real-time task implementation (Mitsi et al., 2008); uncertainty of object location is not considered which means the object location must be precisely known (Berenson et al., 2008).

In analytic model based approaches, robot's workspace is analysed and a model which measures how easily a specific robot configuration can be reached is generated. Suitable configuration for executing an action command can be found by searching the model (Seraji, 1995, Nagatani et al., 2002, Guan et al., 2006, Zacharias et al., 2007, Diankov et al., 2008, Detry et al., 2009, Berenson et al., 2009, Vahrenkamp et al., 2009). Limitations in these approaches can be summarised as: 1) the complexity for calculating the workspace of robot arm with higher Degree of Freedom (DoF) grows exponentially (Seraji, 1995); 2) algorithm can only be applied to particular tasks (Nagatani et al., 2002); 3) the process for generating the analytic model costs long computation time (Guan et al., 2006, Detry et al., 2009); 4) self-collision of robot arm and the influences of obstacles are not considered (Zacharias et al., 2007); 5) the destination pose of robot's gripper must be explicitly defined before action execution (Diankov et al., 2008,

Vahrenkamp et al., 2009); 6) suitable robot configurations may be discarded if they are not sampled (Berenson et al., 2009).

In vision based approaches, the grounding of goal configurations is guided by vision signals (Zhang et al., 1999, McGuire et al., 2002, Roy et al., 2003, Bower and Lumia, 2003, Popovic et al., 2010). The limitations in these approaches can be summarised as: 1) vision signal processing can be time consuming (Zhang et al., 1999); 2) user intervene is required during task implementation (McGuire et al., 2002); 3) the grounding relies on the visibility of target objects and that of the robot's gripper (Roy et al., 2003, Bower and Lumia, 2003, Popovic et al., 2010).

In statistic based approaches, uncertainty of target object location are often modelled as a probability distribution. The probability distribution is updated through perception of the environment or through symbolic instructions from a human user (Mavridis and Roy, 2006, Lemaignan et al., 2011). These approaches rely on sensor readings to gradually establish the probability distributuions which can be time consuming and not suitable for real-time task implementation.

In experience based approaches, robot base poses for grasping an object can be grounded from successful experience acquired by trail-and-error interaction with the environment or by observing human's motion data (Tenorth and Beetz, 2008, Stulp et al., 2009). The limitations of these approaches can be summarised as: 1) the approaches require the accumulation of the successful experience to a certain level; 2) the approaches become inefficient in unstructured environments since changes in environment will invoke a new learning process.

When dealing with uncertainties raised in unstructured environments, the above mentioned approaches often rely on sensor readings to gradually establish probability distributions or a trail-and-error procedure to accumulate successful experiences to a certain level. It is very time consuming to establish probability distributions for objects in the environment during task implementations and the successful experiences learned for implementing certain tasks may be no longer

suitable when the condition of the environment changes. In the real world, given symbolic commands, human can handle well the uncertainties using human reasoning. Fuzzy logic is one of the methods that mimics human reasoning and, therefore is the natural candidate to handle uncertainties in the process of symbolic grounding. In this research, uncertainties in determination of robot configurations in terms of robot base regions for implementing object fetching tasks are modelled by 3D Fuzzy Reachability Space (FRS). The FRS represents a fuzzy relation from robot base poses to the extents to which a robot comfortably grasps an object given the inexact and incomplete environmental information. According to human reasoning, the robot base pose for comfortably grasping an object may not be unique and it can be blocked by obstacles in practice. Therefore, finding such a pose is an optimisation under uncertainty and can be handled with fuzzy optimisation. An objective function is established based on FRS and obstacles in the environment are modelled as 3D fuzzy constraints. A novel algorithm is used to perform optimisation and calculate the most suitable pose for grasping the target object. In unstructured environments, it is very time-consuming for a robot to navigate to an exact pose since the robot will have to gradually calculate trajectory and velocity to position itself to the target pose. It is inspired by human reasoning that when humans approach to an object they do not turn to calculate the exact positions for themselves to pick up the object. To improve the efficiency of task implementation, symbolic action commands can be grounded as regions. When a robot is navigating to a target region, it only needs to decide whether its base is located within the region. This will save the time for exactly placing the robot base. It is unnecessary to have exact robot base poses due to the use of human reasoning. Therefore, a fuzzy objective function is defined and a novel algorithm is used to determine the optimal base region for implementing an object fetch task based on fuzzy objective function. The fuzzy optimisation based symbolic grounding for robot base pose does not fully rely on the previous successful experience, the establishment of probability distributions, the human intervention and the precise visibility of the target object.

This research is carried out based on Care-O-bot 3 (COB3), one of the two benchmark multiple-functional service robot platform (The other is PR2 developed by Willow Garage). COB3 is developed by Fraunhofer IPA and widely used in EU projects. Software is developed based on Robot Operating System (ROS), the open source robot-control software. It is used worldwide. This research is in conjunction with an EU FP7 ICT research project of Shadow Robotic System (SRS) for independent living. The aim of the project is to develop robust personal assistive robots using Robot Operating System (ROS) and Care-O-bot 3 as the initial demonstration platform. The core of SRS is an autonomous control framework which enables Care-O-bot 3 to implement object fetching tasks autonomously in domestic environments. The control framework includes an autonomous symbolic task planner which generates symbolic action sequences based on task commands given by a human user and a task coordination which controls the execution of action sequences based on specific robot configurations. The work presented in this thesis is used to bridge the symbolic task planner and the task coordination by grounding symbolic action commands into specific robot configurations in terms of robot base regions. The fuzzy optimisation based symbolic grounding algorithms presented in this thesis are realised as ROS service blocks and integrated with the autonomous control framework. Experiments were carried out to test the symbolic grounding algorithms where Care-O-bot 3 tried to implement object fetching tasks in domestic environments.

1.2 Problem Definition

Symbolic grounding for service robots working in unstructured environments involves challenges in several aspects:

1) **Handling uncertainties raised in unstructured environments:** Due to the unstructured and the dynamic features of the environments where service robots work, it is often hard for the robots to have the exact and complete information of the environment. Sensor readings, which can have measurement errors, can provide imprecise information about the size,

shape and location of objects in the environment. Objects can be introduced in or removed from the environment randomly. More important, the uncertainties exist among robot poses. Due to limited sensing and actuating capabilities, it is often the case where there are differences between the actual target pose and the one that the robots eventually reached. These uncertainties need to be handled in the symbolic grounding process.

2) **Grounding the optimal robot configuration in clustered environments:** The environments where service robots work are often clustered with obstacles. When the grounded robot base pose is close to obstacles, it is more difficult for a robot to navigate its arm. Some robot arm trajectories may even be blocked. In some cases, the grounded robot base pose may be occupied by an obstacle. Therefore, the influences of obstacles to the determination of the optimal robot base pose must be considered in the symbolic grounding process.

3) **Improving efficiency of task implementation:** In unstructured environments, it is very time-consuming for a robot to navigate to an exact pose since the robot will have to gradually calculate trajectory and velocity to position itself to the target pose. The efficiency of task implementation can be improved by grounding symbolic action commands into robot base regions. When a robot is navigating to a target region, it only needs to decide whether its base is located within the region. This will save the time for exact positioning the robot base. The problem of how to determine the optimal robot base regions for implementing symbolic object fetching commands needs to be addressed.

1.3 Aim and Objectives

This research aims to develop novel symbolic grounding algorithms that enable service robots to implement object fetching tasks autonomously in unstructured environments.

Specific objectives of the research are:

- **To conduct a comprehensive review of existing researches in symbolic grounding:** The existing approaches for grounding symbolic action commands into specific robot configurations will be investigated. Problems found in these approaches will be summarised.

- **To develop and implement a novel algorithm that handles uncertainties raised in unstructured environments:** This is important as the use of fuzzy reasoning will ensure the uncertainties to be handled and therefore, enable symbolic grounding to be conducted in unstructured environments.

- **To develop and implement novel fuzzy optimisation algorithms that determine the optimal robot configurations in terms of robot base poses for implementing object fetching tasks:** The environments where service robots work are often clustered with obstacles. The influences of obstacles to the determination of the optimal robot configuration will be taken into account.

- **To develop and implement novel algorithms for establishing fuzzy objective functions and performing fuzzy optimisation based on fuzzy objective functions:** The symbolic commands will be grounded as robot base regions using fuzzy objective function.. It takes less time for a robot to reach a region than a specific pose. Therefore, the efficiency of task implementation will be improved.

- **To design and carry out experiments in domestic environments:** The proposed symbolic grounding algorithms will be realised and integrated with the autonomous control framework developed in the SRS project. Experiments will be designed and carried out in domestic environments to test the proposed algorithms.

1.4 Thesis Organisation

The rest of the thesis is organised into 6 chapters. Chapter 2 gives a literature review on the state of the art of symbolic grounding researches. Chapter 3 describes a fuzzy reasoning based algorithm for establishing a Fuzzy Reachability Space (FRS) that handles uncertainties in unstructured environments. Chapter 4 presents methods for establishing an objective function and fuzzy constraints. A fuzzy optimisation algorithm that determines the optimal robot base pose based on the objective function and the fuzzy constraints is also presented. In Chapter 5, the method for establishing a fuzzy objective function and a fuzzy optimisation algorithm that determines the optimal base region are introduced. The integration of the proposed algorithms with an autonomous control framework and the experiment results and analysis are provided in Chapter 6. Finally, conclusions and further work are given in Chapter 7.

CHAPTER 2 LITERATURE REVIEW

Autonomous robots are aimed to accomplish useful tasks without human intervention in real-world environments. One of the fundamental tasks is object manipulation. When implementing such tasks, a robot first receives an action command from the user or from a task-level planner. The action commands often contain symbolic terms such as "*near*" or "*close*". For example, an action command "*move(base, near, Table_1)*"represents the action of moving the robot base to a suitable pose near to the object "*Table_1*" for performing object manipulation tasks. Another action command "*move(gripper, close, MilkBox0)*" represents the action of moving the gripper to a pre-grasp pose close to the target object "*MilkBox0*" in order to grasp it. The robot will need to know where to position itself so that the target object can be grasped comfortably and what is the suitable joint configuration for reaching or grasping the target object. Symbolic grounding is needed to translate symbolic task commands into specific robot configurations.

In the first section of this chapter, approaches to the problem of grounding symbolic object manipulation commands to specific robot base poses or joint configurations are investigated. Uncertainties in determining the target object location, suitable robot base pose or joint configuration for grasping objects and the influences of obstacles to the determination of robot configurations are handled by a variety of methods. Since the approach proposed in this work uses fuzzy optimisation to determine suitable base regions for grasping objects, in the second section of this chapter, the methods for learning fuzzy systems are also reviewed.

2.1 Grounding Symbolic Object Manipulation Commands into Specific Robot Configurations

In this section, approaches for grounding suitable base poses or joint configurations for object manipulation tasks are reviewed. Choosing a suitable base pose is essential to the success of grasping. An incorrect robot base pose will make the robot unable to reach the target object or increase the difficulty of arm control. Robot base pose also determines the difficulty of path planning. If the robot base is placed in a cluttered area, it will be more difficult or impossible to escape. Furthermore, the robot arm is more likely to collide with obstacles when grasping the object (Berenson et al., 2008). Uncertainties raised by real-world environments put challenges on determining the suitable robot base pose or joint configuration. On one hand, it is not possible to have exact and complete prior knowledge of these environments. For example, the exact location of the target object or the obstacles is often unknown by the robot and the suitable grasping configuration cannot be defined beforehand. This information has to be acquired through perception and interaction with the environment. On the other hand, knowledge acquired through sensing is affected by uncertainties and imprecision. The quality of sensor information is influenced by sensor noise, the limited field of view, the condition of observation, etc. (Hagras and Sobh, 2002). For example, the localisation and navigation of the robot base suffered from sensor impressions. The robot cannot be exactly navigated to a desired pose and it doesn't know precisely where it is currently located. To address the above mentioned problems, a variety of approaches have been proposed. The approaches can be classified into optimisation based, analytic model based, vision based, statistic based and experience based. The approaches are classified due the method used. In optimisation based approach, an objective function or evaluation function is often define based on certain criteria such as grasp quality, manipulability measure of the arm, distance to the obstacles, task execution time, etc. The optimal configuration can be determined by evaluating a solution space (Zhao et al., 1992, Hsu et al, 1999, Mitsi et al., 2008, Berenson et al., 2008). In analytic model based approaches, the workspace of the robot is analysed and a model which measures how easily a specific robot configuration can be reached is generated. The robot base or gripper pose can be grounded by searching the model (seraji, 1995, Nagatani et al., 2002, Guan et al., 2006, Zacharias et al., 2007, Diankov et al.,

2008, Detry et al., 2009, Berenson et al., 2009, Vahrenkamp et al., 2009). In vision based approaches, the grounding of goal gripper pose is guided by vision signals (Zhang et al., 1999, McGuire et al., 2002, Roy et al., 2003, Bower and Lumia, 2003, Popovic et al., 2010). In statistic based approaches, uncertainties in the determination of target object location are often modelled as a probability distribution. The probability distribution is updated through perception of environment or through symbolic instructions from the user (Mavridis and Roy, 2006, Lemaignan et al., 2011). In experience based approaches, robot base poses for grasping an object can be grounded from successful experience acquired by trial-and-error interaction with the environment or by observing human's motion data. Uncertainties in determining the robot base pose are often modelled as a probability distribution (Tenorth and Beetz, 2008, Stulp et al., 2009).

2.1.1 Optimisation based approaches

Zhao et al. (1992) were one of the first to use an optimisationalgorithm to deal with the uncertainty of the determination of a sequence of robot base poses and manipulator configurations for performing a sequence of given actions. A cost function is defined where the total cost of a sequence of robot configurations is the weighted sum of the cost of moving the base and the manipulator. The suitable sequence of robot base pose is found by exhaustive search of the robot's configuration space. However, obstacles and object grasping are not considered in this research.

Hsu et al. (1999) proposed an optimisation based algorithm that makes use of randomised motion planning techniques to compute a base location for a manipulator so that specified tasks are executed as efficient as possible. While the manipulator is in motion, the base remains stationary. A requirement for the base location is that the reachable workspace of the manipulator covers all the task points. Furthermore, the robot base should be placed to enable efficient task execution, that is, to minimise the execution time. Their work also takes into account the impact of obstacles in the environment.

The algorithm first computes a collision-free path for an initial base location, using standard randomised path planner. The path computed by the planner is then deformed to obtain a locally optimal path. Finally the robot base is moved to better locations iteratively. At each step of the iteration, a new collision-free path is recomputed. The path found in the previous iteration is used as a starting point for finding a new path in the current iteration. A collision-checker determines whether a path configuration is free or not. To check whether a path is collision-free, it will be partitioned into a sequence of configurations. If all the configurations are collision-free, it is regarded as a collision-free path.

The algorithm was tested on real-life data from the automotive industry. Experiment results show that the algorithm significantly reduce the task execution time by choosing a suitable base location. However, as this algorithm is designed for industrial robots, it only optimises the time for moving the manipulator. The grasping of objects is not considered. The problem of positioning error of the robot base is also not addressed in this work.

Mitsi et al. (2008) dealt the problem of determining the optimalbase location and joint angles for a robot to reach some prescribed end-effector poses with an optimisation algorithm. The optimisation is conducted through a hybrid heuristic method that combines the advantages of a genetic algorithm (GA), a quasi-Newton algorithm (QNA) and a constraints handling method (CHM). GA applied alone has the advantage of searching the whole space of solutions as well as not being trapped in a local minimum. However, GA is efficient only for limited number of variables. QNA has the advantage of detecting local minimums for higher number of variables but it is strongly depending on the initial searching point. CHM is applied in order to reduce the searching space and accelerate the procedure.

The problem of optimal placement of the robot base is solved by minimising the sum of the deviation squares between the prescribed poses and the calculated

poses of the end-effector, as well as by maximising the manipulability measure of each configuration. The proposed method is applied to a 6-DoF manipulator. The objective function can be described by

$$F = F_1 + \alpha \cdot F_2 \tag{2.1}$$

where

$$F_1 = \sum_{k=1}^{n} \sum_{i=1}^{3} \sum_{j=1}^{4} (A_{Sr}^{7}(i,j) - A_{Spr}^{7}(i,j))_k^2, \tag{2.2}$$

is the sum of deviations squares between the calculated end-effector poses and the corresponding prescribed poses,

$$F_2 = \sum_{k=1}^{n} (\frac{1}{w_k^2}) \tag{2.3}$$

is the sum of inversed manipulability measures square, n is the number of prescribed poses, $A_{Sr}^{7}(i,j)$ is the calculated value of the element (i,j) of the A_S^7 matrix, $A_{Spr}^{7}(i,j)$ is the prescribed value of the element (i,j), w_k is the manipulability measure for robot configuration of pose k and α is weighting factor.

The measure of manipulability is defined as

$$w = \sqrt{\det(J \cdot J^T)}, \tag{2.4}$$

where J is the Jacobian matrix. The Jacobian matrix of the manipulator is calculated by taking into account the joint angle values for each poses.

The optimisation method combines GA, QNA and CHM. The starting populations for the GA are randomly generated to set variable values. The variable values are used to calculate the fitness function value. The fitness function is defined by the objective function. GA uses selection, elitism, crossover and mutation procedures to create new generations. After some repetitions when the maximum generation number is reached, the variable values with the minimum fitness function value are selected. The optimum GA variable values are sent to QNA as the initial searching point. The QNA modifies the values using a finite-difference gradient method until a maximum iteration number or a local minimum is reached. Afterwards, CHM is used to reduce the bounds of the variables. Each bound reduction leads to a new round of GA and

QNA loops. The optimisation is finished when the maximum CHM loop number or the fitness function goal is reached. It is shown through experiment that the result obtained by using the hybrid method is better than the one obtained by GA or the combination of GA with CHM. However, the computational time of the hybrid method is around 10*mins* which may not be able to meet the requirement of real-time task implementation.

Berenson et al. (2008) used co-evolutionary algorithm to optimise robot base pose for implementing object pick-place tasks. It has been known that a suitable base pose will make the robot arm control easier. If the robot base is placed at a location far from obstacles, there will be more feasible arm configurations. Besides, the navigation of the robot base is more likely to meet a time constraint. These issues are addressed by a co-evolutionary optimisation algorithm. Three metrics are defined for evaluating a robot base pose configuration:

- **Grasp quality:** defined as force-closure which can be measured with a variety of matrices.
- **Configuration desirability:** refers to the cost of being in a certain configuration of the arm. The cost is robot dependent. It gives better scores to configurations that are far from singular configurations.
- **Configuration clutter:** measures the distance from each of the robot's links to the nearest obstacle.

The overall score for a robot base pose configuration is the weighted sum of the three matrices.

Objects in cluttered environments can be grasped in multiple ways. This makes the problem of finding the optimal base pose highly non-linear with many local minima. The co-evolutionary algorithm can efficiently avoid these local minima. The evolutionary structure consists of three populations of individuals: 1) the population of robot base poses for grasping the object placed at its initial location; 2) The population of robot base poses for placing the object at its goal location; 3) The population of grasp configurations. The individuals contain genes for an offset in X and Y-axis which allow the evolutionary algorithm to improve on

good solutions in later generations via mutation. Each population is associated with its own fitness function. The fitness of one population depends on the individuals in one or more of the other populations. The fitness function guides all three populations to a solution that has maximum score.

Simulation has been carried out with a Puma robot model. Results show that with co-evolutionary algorithm, the successful rate of the task implementation was improved and the time needed for function evaluation was reduced in comparing with the random function sampling. However, the uncertainty of the target object location is not considered in this approach which could cause failure in transferring to real-world.

2.1.2 Analytic model based approaches

Workspace analysis and generation has been an important issue in robotics, since this knowledge is essential for the grounding of suitable robot configurations. Many methods have been represented for the analysis, determination and generation of the workspace of manipulation.

Seraji (1995) proposed an off-line method to determine the appropriate base locations from which the robot can reach a target point. It has been known that proper placement of the robot base is crucial for successful execution of tasks. Base placement can be done manually by the operator based on visual data obtained from the worksite. These approaches are prone to error due to potential misjudgement of the operator. In the method proposed by Seraji, “arm reachability” is used as the basic criterion for base placement to ensure that the target is within comfortable reach of the arm. The method is one of the earliest to use an analytic model ofrobot arm to determine the appropriate location of robot base. Before addressing the base placement problem, the workspace boundaries of a 3-DoF arm Puma is investigated. The size of the reachable workspace is calculated from the angle constraints for the arm joints. When a set of randomly positioned target points is given, the base location for reaching the entire target

point set can be determined by the intersection of the robot arm's workspace and the target points. The percentage of the target points covered by the workspace is used as the "arm reachability" value for determining whether a robot base location is suitable for task implementation. By setting restrictions to the joint angles, the workspace of the robot arm becomes smaller. This allows undesirable over-extended or under-extended arm configurations to be avoided at the user's specification. The method can also be extended for finding a robot base location for reaching a target surface or volume. The limitation of this method is when it is applied to robots with higher DoF, the complexity for calculating the workspace grows exponentially. Besides, uncertainties in the robot's working environment are not considered.

Nagatani et al. (2002) tried to solve the problem of robot base path planning while keeping manipulability at the tip of the robot arm. The task of the robot is to draw a large object on a wall. A path planning algorithm is designed for finding a trajectory (a set of robot base pose) for implementing the task. One of the general approaches to this problem is to consider the robot base as extra joints of the arm. However, according to the implementation experience, the arm controller should be different from the base controller. Thus, in their approach, an analytic model is defined for representing how difficult the robot arm can be operated when the robot base is place at different locations. Assuming the tip of the robot arm is perpendicular to the wall, a valuation of manipulability can be calculated for each robot base location. The manipulability ω is defined by the Jacobian matrix, shown in this equation:

$$\omega = \sqrt{\det(J(q)J^T(q))}. \tag{2.5}$$

The distribution of manipulability is calculated from joint angles that are calculated by inverse kinematics. The shape of a manipulability distribution is illustrated in Fig. 2.1.

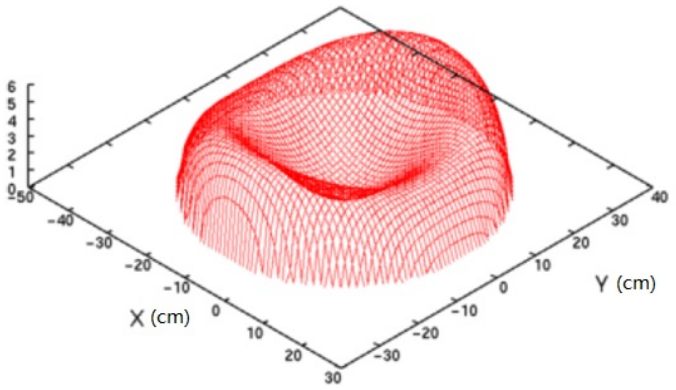


Figure 2.1 Manipulability distribution (Nagatani et al., 2002)

In Fig. 2.1, X-axis and Y-axis show robot base locations. Z-axis shows the manipulability value. By setting a fixed value of manipulability, an area called Manipulability Area (MA) can be generated by slicing the manipulability distribution by an X-Y plane. When the robot base is placed within the MA, the manipulability is secured higher than a fix value. The movement trajectory for implementing the task can be determined as the shortest collision-free trajectory within the MA.

Experiments were executed in an actual robot in real environment. Results show that it is difficult to draw a desired segment on a wall due to an initial positioning error of the robot base. Extra sensors are required to detect the error. Moreover, the task of the robot is to draw a large object on a wall, thus the tip of the robot arm is assumed to be perpendicular to the wall.

Guan et al. (2006) studied the reachable space of a biped humanoid robot in standing postures and proposed a numerical approach using the Monte Carlo method for generation of workspace of the robot. The environments where humanoid robots work are often complex and filled with various obstacles. The robot often stands stably with two feet fixed on the ground and the feet locations are confirmed by obstacles. It must know whether the robot hand can reach the target object. Traditional workspace analysis methods are impractical since more

Degrees of Freedom (DoF) are involved and more constraints must be taken into account. In this approach, a numerical Monte Carlo method is proposed to the generation of reachable boundary of the robot. The boundary is constructed by a series of extreme points that the robot arm could reach. Constraints including the joint limits are also coped with this method. Before applying the Monte Carlo method, constraints of the robot's joint angles must be defined. Suppose a humanoid robot is in a standing pose with two feet fixed on the ground, a world frame,Σ_w is defined and the origin is located at the middle of the two feet on the ground. A body frame Σ_b at the hip is also defined. A reference point, denoted by $P$is defined at the centre of the robot's right hand. When a robot stands with two feet on the ground, the two legs, the waist, and the ground form a closed chain. The transformation T_b^w of the body frame with respect to the world frame can be calculated according to the kinematics of each leg. The kinematics of the two legs, $T_e$and T_r, which are functions of leg joints and foot location must be equal.

$$T_l\left(\Theta_l^l, F_l\right) = T_l(\Theta_r^l, F_r) \tag{2.6}$$

whereF_l and F_r indicate the locations of the left and right feet, respectively. Θ_l^landΘ_r^l indicate the joint angle vectors of the two legs. This imposes a kinematic constraint on the joint angles of the two legs.

Another constraint must be taken into account, which is the balance of the robot. The constraint is that the projection of centre of mass (CoM) of the robot onto the ground must be within the convex hull of the supporting area. Monte Carlo method is applied by sampling hip pose instead of the joint angles of the lower body. Suitable intervals are selected for the random sampling. For an arbitrary hip pose that the legs have no IK solution will be discarded. If a sampled hip pose is valid, the sampling of joint angles of the upper body follows. The proposed method for generating the robot's workspace is straightforward, but with costs of long computation time.

Zacharias et al. (2007) used a directional structure to represent a robot arm's capabilities in its workspace. The directional structure is called capability map.

Using this capability map, a robot is able to deduce places that are easy to reach or position itself to enable optimal manipulation of an object. For grasp planning, it is important to have an abstraction model of a robot arm's capability in its workspace. To calculate the capability map of a robot, the theoretically possible workspace of the robot arm is modelled by a cube with a side-length of arm length centred at the robot base. The cube is then subdivided into equally sized smaller cubes. In each cube, a sphere with a diameter equal to the width of the cube is inscribed. On each sphere, *N* equally distributed points are generated. For each point, a frame is generated. The frame is turned around its Z-axis according to a fixed step-size. An inverse kinematic solution is computed for each resulting frame. The reachability value for a sphere is defined as the percentage of points that have an inverse kinematic solution. By investigating the spheres with high reachability value, the capability map, which is a cone-shaped structure can be found. The capability map of a 7-DoF robot arm is illustrated in Fig. 2.2.

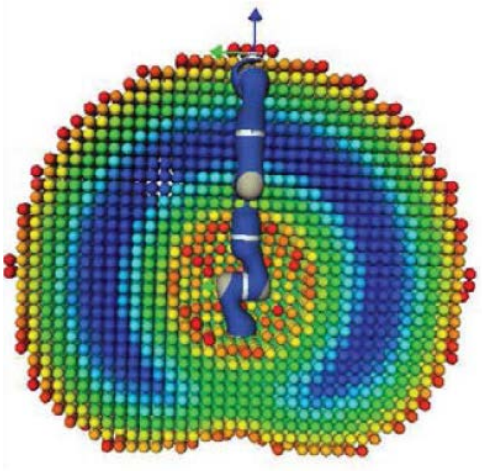


Figure 2.2 Capability map of a 7-DoF robot arm (Zacharias et al., 2007)

Fig. 2.2 shows the reachability sphere across the workspace. The reachability value of each sphere is represented by different colours. Experiments show that the capability map can help a two-handed humanoid robot Justin to decide which arm is best for certain tasks or to find a location where versatile grasping is possible. However, the capability map does not take into account self-collisions

of the arm and the influences of obstacles (e.g. the table where the target object is placed) is also not considered.

Diankov et al. (2008) presented a planning algorithm called BiSpace that finds robust solution to the manipulation and grasping problem by exploring the work and configuration spaces of the robot. Planning algorithms are required to perform quickly in domestic environments so that the resulting solution can be executed before the environment changes. The proposed algorithm reduces planning time by modifying the configuration space sampling distribution through the bias of grasping goal. Given a target grasp g, BiSpace will compute a distribution over the 2D placement of the robot base for which g will be successful. First a kinematic workspace analysis is performed for the HRP2 humanoid robot to generate a reachability volume. Fig 2.3 illustrates the arm reachability volume generated for the HPR2 Humanoid robot.

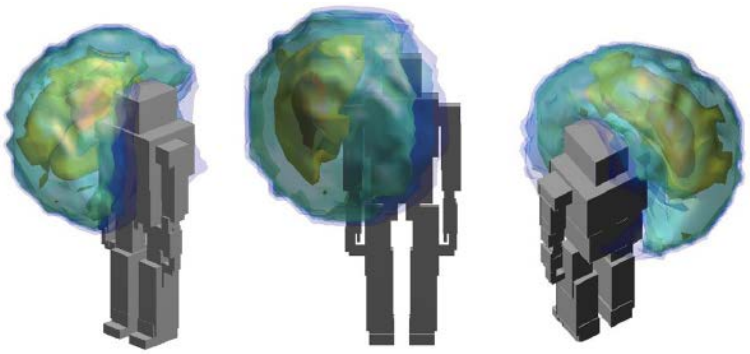


Figure 2.3 Arm reachability volume for HPR2 robot (Diankov et al., 2008)

Fig. 2.3 shows three different views of the reachability volume. The opaque area contains more reachable end-effector poses. This is computed by randomly sampling a 6D end-effector pose around the space of the robot's shoulder and querying for an Inverse Kinematic (IK) solution. The valid 6D end-effector poses will be stored in a grasping space. Then similar grasps to g in the grasping space are selected. Simple counting is performed to extract a probability of existence of

an IK solution. The next step is to compute the inverse reachability volume for the grasps similar to g. Finally, the inverse reachability volume will be converted into a probability distribution. Empirical results show that BiSpace is able to generate a feasible base placement 2.5 times faster than uniform sampling around the grasp. Since the algorithm is goal biased, the grasping pose of the end-effector must be explicitly defined before task execution.

Detry et al. (2009) developed object grasp affordances which is a probabilistic model of object-gripper relative configurations that lead to successful grasp. The grasp affordances store the whole knowledge about the grasping of an object that could be used to facilitate reasoning on grasping solutions. The affordance representation is modelled by a continuous probability density function defined on the 6D gripper pose space within an object-relative reference frame. Grasp affordances are initially learned from imitation or visual cues, leading to grasp hypothesis density under various object poses. The outcomes are used to learn grasps that are confirmed through experience.

Grasp affordances allow a robot to learning initial affordance from various grasp cues, and enrich its grasping knowledge through experience. The affordances are initially constructed from human demonstration, or from a model based method. The grasp data produced by these grasp sources is used to build continuous grasp hypothesis densities. These densities are attached to a 3D visual object model learned beforehand. The robot will execute samples from the grasp hypothesis densities. The successful samples, which is a fraction of the hypothesis densities are used to build the object grasp affordance model. The grasp affordance can lead to many potential applications. By combining with robot capability or external constraints, a robot can select grasp that has the largest chance of success within the subset of achievable grasps. However, the grasp sample execution involves trail of error procedure which could be time consuming when implementing household object manipulation tasks.

Berenson et al. (2009) used Workspace Goal Region (WGRs) to find goal end-effector poses for a mobile robot. The goals are specified intuitively as volumes in the robot's workspace. The previously mentioned approaches often tackle the problem of finding a goal configuration for a robot by sampling some number of end-effector poses and conducting IK to find joint configuration which place the end-effector at the sampled poses. These approaches are often probabilistically incomplete, which means even if some goal pose are reachable, they will be neglected if not sampled. To define WGR, a transformation matrix T_b^a, which specifies the pose of b in the coordinates of frame a is used. T_b^a consists of a 3×3 rotation matrix R_b^a and a 3×1 translation vector t_b^a,

$$T_b^a = \begin{bmatrix} R_b^a & t_b^a \\ 0 & 1 \end{bmatrix}. \tag{2.7}$$

A WGR consists of three parts:

- T_w^0: reference transform of the WGR in world coordinates.
- T_e^w: end-effector offset transform in the coordinates of *w*.
- B^w: 6×2matrix of bounds in the coordinates of *w*.

The *w* frame is usually centred at the origin of an object. An offset from *w* to the origin of the end-effector *e* can be specified by T_e^w. To find the distance from a given configuration q_s to a WGR, forward kinematics is used to calculate the pose of the end-effector at the configuration T_s^0. Then the inverse of the offset T_e^w is applied to get T_s^0, which is the pose of the grasp location. Sampling from a single WGR is done by first sampling a random value between each of the bounds defined by B^w with uniform probability. These values are then complied in a displacement d_{sample}^w and converted into the transformation T_{sample}^w. This sample can then be converted into world coordinates by applying the end-effector transformation. WGRs can be used to plan robot arm trajectories by using IKBiRRT or RRT-JT planning algorithm.

Vahrenkamp et al. (2009) proposed a novel motion planning algorithm for finding a whole-body configuration of a dual-arm robot for grasping robot's pre-computed reachability space with random sampling of free parameters. Finding a

robot configuration that places the end-effector at a given pose is known as the Inverse Kinematics (IK) problem. It is possible to find a configuration for a manipulator with no more than 6 Degree-of-Freedom (DoF) by using an IK solver. In this work, a robot model ARMAR-3 with two 7-DoF arms and a 3-DoF hip is used. To find the whole-body configuration of ARMAR-3, the 3 hip joints are randomly sampled until a configuration of the robot arm is found which brings the end-effector to a grasping pose. The configuration also has to be checked against self-collisions and collisions with obstacles. The search for an arm configuration is stopped after a specific number of tries and it is assumed that there is no valid result. To find a configuration for the robot arms, a reachability space is used. The reachability space is represented by a grid of voxels in 6D pose space. Each voxel holds information about the probability that robot arm configuration can be formed. A 2D view of the reachability space of ARMAR-3 robot is illustrated in Fig 2.4.

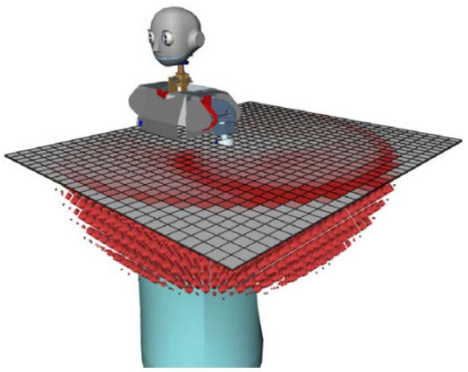


Figure 2.4 Reachability space of ARMAR-3 robot (Vahrenkamp et al., 2009)

In Fig. 2.4, the colour intensity is proportional to the probability that a pose inside voxel is reachable. The reachability spaces can be determined by solving a large number of IK requests and counting the number of successful queries for each voxel. A gradient descent approach can be used to optimise the search for a reachable grasping pose. If the reachability space entry of a grasping pose lies above a threshold, a robot arm configuration is found. The gradient descent

search also checks its neighbouring voxel of the reachability space. If there is a voxel with a higher reachability space entry, the grasp pose is moved towards that voxel. This process also repeats until there are no neighbours with higher entry. This approach to the grounding of robot body configuration has the dependency of a good starting guess. It also relies on fix starting and goal positions which means the environment is precisely known.

2.1.3 Vision based approaches

Zhang et al. (1999) proposed a vision-guided neuro-fuzzy controller for fine positioning a manipulator on to a grasping pose. Traditional methods for vision-guided fine-positioning are based on hand-eye calibration. Such methods depend on fixed hand-eye configuration and robust extraction of geometric features for detecting the grasping pose. Neural network based learning also has been applied in grasping. Geometric features are used as inputs to the pose controller. Since the image processing algorithms are not robust in real environments and computationally expensive, some of the approaches use marking points on the objects for grasping. In their work, principal component analysis is used to reduce the dimension of raw camera image to lower-dimension eigenvectors that can be used as inputs of a neuro-fuzzy controller. Eigenvectors are partitioned by covering them with linguistic terms. The fuzzy controller is constructed according to the B-spline model. The linguistic terms for input variables are defined with B-spline basis functions and for output variables are defined with singletons. The working system implements two phases: off-line training and on-line evaluation. In the off-line phase, a sequence of training images showing the same object in different poses is taken automatically. For each image the pose of the manipulator is recorded. In the on-line phase the camera output is transformed into the eigenspace and is then processed by the fuzzy controller. The controllers output is the end-effector’s pose. The fuzzy controller first produces the pose of the object in the image. These values are then used to move the robot closer to the target object. This process is repeated several times until the deviation of the end-effector in X and Y direction and angular deviation are below a specific threshold.

Experiment results show that this approach is calibration-free, model-free, and robust in real-environment. However, this approach only considers the 3D pose of the end-effector, but not the manipulator's base pose. In the training phase, up to 100 target objects image is needed to be processed which could be time consuming when dealing with more household objects.

In McGuire et al.'s (2002) approach, symbolic and gestural instructions are integrated with visual perceptions and other sensor feedbacks to guide a robot manipulator to finish object manipulation tasks. In order to let the robot to be controlled easily and intuitively, the robot must be capable to understand human's symbolic or gestural instructions and finish the corresponding task autonomously. In this approach, Bayesian Network is used to model semantic categories and spatial relations between objects. Symbolic or gestural instructions will be analysed and segmented into these categories and used as evidence for the robot to decide which object is the human intended one. To illustrate the capability of the approach, an object picking up and deploying task is implemented: First, a number of objects are spread on a table in the workspace of robot arm and camera. A user gives a symbolic instruction referencing one of the objects. The instruction is semantically analysed. The robot may ask for additional pointing information. When a pointing hand is found, the gesture is evaluated and a 3D interest region is generated to help the robot to resolve ambiguities. The robot determines the object to be grasped and a control schema is passed to the arm system. The robot then performs a visually guided approaching movement.

The arm control is implemented as a finite state automation switching between different arm modes and hand states. The transitions are triggered by visual and tactile feedback. In particular, a wrist camera provides visual feedback to approach the grasp offset location and in the grasping phase, the finger tip sensors provide the necessary force feedback.

This approach models uncertainties in the human's symbolic instructions of the target object location with Bayesian Networks. The robot could resolve these uncertainties through requiring more information (e.g. gestural instructions) from the human and from its visual feedbacks. However, the robot still works in a semi-autonomous manner, which means human intervention is required during task implementation. Besides, the robot base pose for grasping the object is not considered.

Roy et al. (2003) introduced a robotic architecture Ripley that provides a basis for grounding spatial symbols such as "above" and "left" to an arbitrary object location and a robot arm movement trajectory for reaching the object. To enable grounding, a set of spatial relations is measured by a vision system between a pair of objects. The first feature is the angle of the line connecting the centres of an object pair. The second feature is the shortest distance between the edges of the objects. The third spatial feature measures the angle of the line which connects the two most proximal points of the objects. By integrating real-time information from the robot's visual system, a mental model of the environment is constructed. The mental model is built upon the Open Dynamic Environment (ODE) rigid body dynamics simulator. As the physical environment changes, perception of these changes drives the creation, updating and destruction of objects in the mental model. Objects can be reached out by interpolating between recorded motion trajectories. A set of sample trajectories are trained by placing objects on the table, placing the robot at a location so that the table is in view and then manually guiding the robot until it touches the objects. A motion trajectory library is collected by indexing each trajectory by the location of the target object. To reach an object in an arbitrary location, a linear interpolation between trajectories is computed.The mental model provides a way to ground target object locations, however uncertainties caused by sensor errors are not considered.

Bowers and Lumia (2003) investigated the problem of grounding robot hand configuration for manipulating randomly placed objects of various sizes and shapes. Fuzzy logic expert system is used for mapping from vision data of object

characteristics to robot hand configuration. The vision data provide information about an object's location, orientation, shape and size, etc. The procedure for grasping is to use the vision data to provide a nonlinear mapping from the visual information to finger spread and pre-grasp hand configuration. Once a grasp attempt is made, finger poses and finger force determine a measure of grasp security. In order to create the fuzzy rule base, a data collection program is used. First, 3 pictures of the target object are taken from different angles. The robot then makes a grasp attempt and the finger spread value is collected and prompted to the user. Finally, the obtained hand data for the grasp quality is measured by the definition of force closure grasp. To map the vision data to hand configurations, the objects are classified as basic shapes and a single-input-single-output (SISO) fuzzy rule base system (FRBS) is designed to map the vision input parameter of one basic object shape to one hand output parameter. The vision data collected by using the data collection program are fed into the Fuzzy Logic Development Kit (FULDEK), which is used to create the membership functions and rules for the fuzzy rule bases from FULDEK's Auto Rule function. The input for each fuzzy is made up of five triangular membership functions and the output membership function is a singleton. Experiments show that due to the simplicity of the SISO mappings, the system is capable of generating grasps very quickly and provides adequate grasp quality.

Popovic et al. (2010) suggested a vision base algorithm for grasping an object in a rather complex environment. The algorithm does not make use of any specific object prior knowledge. It is based on spatial information in terms of co-occurrence of colour properties. A target object and the robot's gripper are modelled as 3D contours by a 3D camera to reflect their spatial information. Contours that share a common plane have the similar colour. Fig 2.5 illustrates 3D contours extracted from the scene shown in the bottom left of the figure.

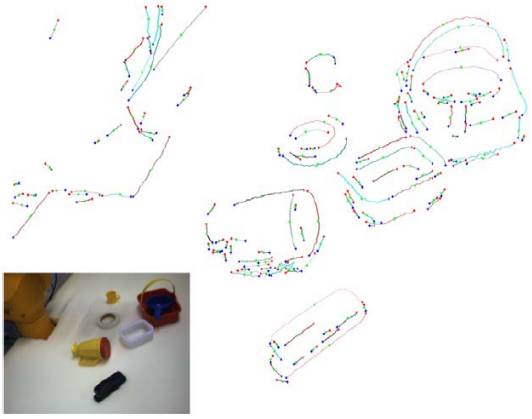


Figure 2.5 3D contours (Popovic et al., 2010)

In Fig. 2.5, contours with the same colour are assumed to be in the same plane. When the gripper contour and the target object contour have the similar colour, a possible grasp pose is obtained. Moving the robot to get the two contours overlapped is the process of trail of error and would involve human intervene in an unstructured environment. This approach fully relies on the visibility of the target objects and that of the robot's gripper. A grasp task will fail if either the target object or the gripper is not visible.

2.1.4 Statistic based approaches

One of the challenges to the aforementioned approaches is the lack of capability for modelling uncertainties in determining the target object location. This is addressed by Mavridis and Roy's (2006) Grounded Situation Model (GSM). GSM is a representational model of a robot's beliefs about its physical environments (i.e. a probabilistic distribution of the location of a target object) as well as pre-coded motor primitives for manipulating the object. With this model, the robot could update the probability distribution of object's location through perception of the environment or through symbolic instructions from the user such as "the ball is on the right of the table". The target object location with the highest probability will be used for generating the appropriate parameterised action schema for implementing object manipulation tasks. The probability

distributions are updated as the weighted sum of the old distribution with a rectangular envelope centred at the new sensor readings. A diffusion process is used to model the decrease of the robot's confidence about the location of target objects over the time. GSM relies on sensor readings to gradually establish the probability distributions and hence can be time consuming in unstructured environments.

Lemaignan et al. (2011) introduced a physical representational model of the environment that can be used as a mediator between the sensor space and symbolic models. This model is called SPAtial Reasoning & Knowledge (SPARK). Compares to the GSM proposed by Marridis and Roy (2006), SPARK offers a richer 3D model that enables the computation of several spatial relationships between objects. In this work, the following relations are computed with respect to the location of the agents and the objects:

- **Location according to a robot:** these spatial locations are computed by dividing the space around the robot (the referent) into n regions based on arbitrary angle values relative to the referent orientation. The number of regions can be chosen depending on the context where the grounding takes place.

- **Location according to an object:** Object locations can also be computed with respect to other objects in the environment. In this work, three main relations are computed based on the bounding box and centre of mass of the object: 1) **isOn** computes if an object O_1 is on another object O_2 by evaluating the centre of mass of O_1 according to the bounding of box of O_2. 2) **isIn** evaluates if an object O_1 is inside another object O_2 based on their bounding boxes. 3) **isNextTo** indicates whether an object O_1 is next to another object O_2. The dimensions of the objects are taken into account.

Each time the current state of the environment changes, these properties will be computed to ensure the model is up-to-date.

2.1.5 Experience based approaches

In Tenorth and Beetz's (2008) Grounded Action-related Model (GrAM), robot base locations for performing household objects manipulation tasks are learning through robot's experiences. First, a robot is controlled to perform different object manipulation tasks in a household environment. Locations where the robot stands when starting to manipulate objects are recorded. These locations are classified into areas with respect to the Euclidean distance of the locations. The areas are called manipulation places. Each manipulation place will be modelled as a probability distribution, e.g. Gaussian distribution. Probability distributions of the robot base locations in manipulation actions are illustrated in Fig. 2.6.

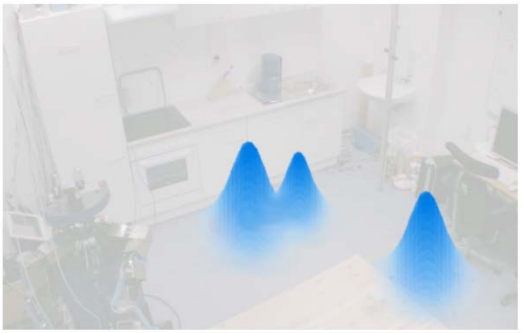


Figure 2.6 Probability distribution of the robot base locations (Tenorth and Beetz, 2008)

The probability distributions as long as the related manipulation tasks are saved in a knowledgebase. Next time when the robot needs to perform a task, robot base location can be estimated using the probability distribution of the corresponding manipulation place.

This approach becomes inefficient when working in unstructured environments since changes in environment will invoke a new learning process and the

approach requires the accumulation of the successful experience to the certain level.

Stulp et al. (2009, 2012) proposed an experience based approach that enables a mobile robot to learn a grasping area from which successful manipulation is possible. The grasping area is modelled as Action-Related Place (ARPLACE). In the approach, the robot acquires experience of ARPLACE through trial-and-error interaction with the environment. Uncertainties in both robot base and target object locations are taken into account which leads to more robust task implementation. To learn the ARPLACE, the robot first gathers experience in simulation by recording successful and failed attempts from different locations. The classification boundaries between successful and failed attempts can be acquired by applying Support Vector Machines for different target object locations. After that, a generalised success model can be calculated over the classifications with a Point Distribution Model (PDM). During task implementation, a Generalised Success Model (GSM) calculates successful boundary for grasping an object from the PDM through a regression process. The ARPLACE, which is a normal distribution of the successful rate of grasping the object, is generated according to the successful area. An example of ARPLACE is illustrated in Fig. 2.7.

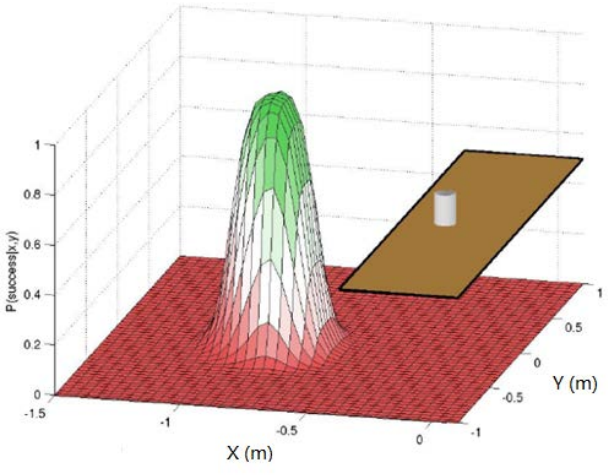


Figure 2.7 ARPLACE (Stulp et al., 2012)

In Fig. 2.7, the horizontal plane shows the locations where the robot stands. The vertical axis shows the probability of successfully grasp the target object. Experiments show that the successful rate of grasping is increased by considering the uncertainty in the object and the robot's location. However in this approach, the influence of the obstacle is not considered. When an obstacle is placed near to the ARPLACE, it cannot be guaranteed the grasping pose is still workable. Furthermore, the altitude of the target object, which is an important issue in determining grasping poses, is neglected.

Based on the ARPLACE approach, Stulp et al. (2009) combined analytic modelling, imitation learning and experienced learning to efficiently learn a suitable grasping pose for mobile manipulation tasks. The core of the approach is to use human motion data to bias the exploration of an analytic model of the robot's workspace. The goal is to improve the efficiency of exploration. The process of finding the suitable grasping pose can be summarised as: First, the workspace of the robot arm is analysed and a reachability sphere map is generated. The reachability sphere map models the successful rates of finding a robot arm configuration to reach the poses inside the workspace. The method used for generating the reachability sphere map is similar to Zacharias et al.'s (2007) approach. The next step is to observe humans performing the similar mobile manipulation tasks in a sensor equipped kitchen environment. The manipulation poses where humans stand will be recorded and classified to places. The boundaries of the places are determined using Support Vector Machine (SVM). The robot could use these places as a starting region to explore the reachability sphere map for finding the suitable grasping pose. The final step is to estimate the target object location. Due to sensor noise and the uncertain environment, the robot has to use a Monte-Carlo simulation to generate a probabilistic advice on the location of the target object. According to the experiments, the exploration efficiency is greatly improved by choosing the starting region based on the successful experience. However, since the learning is based on the human's

successful experience, a new round of learning may be needed when the condition of the environment changes. For example, when the object is placed at different furniture or an obstacle is placed nearby the manipulation place.

2.2 Learning Fuzzy Systems

The approaches for learning fuzzy systems can be classified by the parameters that are subject to adaption:

- **Structure learning:** The terms of each rule and optionally the number of rules are learned. This is normally performed by a Genetic Algorithm (GA).
- **Input membership function learning:** The membership functions are described by functions with unknown parameters. The parameters are optimised by global (GA) or local (back-propagation) nonlinear optimisation.
- **Output membership function learning:** If singletons are used on the output side, the position of singletons can be learned though standard linear Least-Squares (LS) techniques.

Sulzbergar et al. (1993) suggested using neural networks for optimisation of fuzzy rule based systems. A novel neural network model with special neurons is introduced in the proposed method. The performance of the network and the quality of the rule base is improved by training the network using a combination of neural network learning algorithms. The optimised rules and membership functions can be extracted from the net and used in normal fuzzy inference tools.

The network consists of an input, an output and three hidden layers. It is initialised with a fuzzy rule base and the corresponding membership functions. The rule base can be optimised by changing the structure of the net or the data in the neurons. The rules are represented in the net through the connections between the layers. The learning of the rules is implemented as a stochastic search in the rule base. A randomly chosen connection is changed and the network

performance is verified with a cost function. If the performance is worse, the change is discarded. The learning for the membership functions is a combination of gradient descent and a stochastic search. A maximum change in a random direction is initially assigned to all membership functions. The network performance is tested with this membership functions. If the network performance better according to a given cost function, the new value is accepted and another change is tried in the same direction. The method can only be applied to mobile robot navigation problem. The design of the cost function for evaluation the performance of the network is left to the user.

The approaches mentioned above become impractical for higher input dimensions since the number of parameters to be estimated becomes too large. To overcome this problem, Nelles, Fischer and Muller (1996) combined a GA based rule extraction method, called FUREGA, with a linear LS optimisation for learning a fuzzy system.

In the FUREGA, a set of all possible rules is coded as binary strings. The fitness of a fuzzy set is evaluated by a LS optimisation of the singletons. A penalty function that is proportional to the number of rules selected is added as well as another penalty function for singletons that have no physical meanings. The inverse of the penalty function value is the fitness of the corresponding rule set. The penalty for the singletons is calculated by setting a range for every output. Singletons that exceed these bounds lead to an additional penalty term. The rule set will be extracted by using the LS minimization. For example, if the singletons of 3 rules are about the same value, those rules will be approximated by the first one. This reduces the dimension of the rule set. To increase the performance of the extracted rule set, the input membership functions are tuned by a Sequential Quadratic Programming (SQP) algorithm in which an LS optimisation of the output memberships is embedded. The objective function of the optimisation is the normalised mean square error. Experiments were carried out using real-world data. Results show that the FUREGA leads to very small rule sets that are easy to interpret.

With conventional training procedures (e.g. Levenberg-Marqudart technique), disturbance from the environment during a training cycle can lead to instability. This instability may be difficult to alleviate due to the uncertainty concerning the environmental conditions. To overcome this, Efe and Kaynak (2000) proposed an algorithm which combines the conventional Levenberg-Marqudart algorithm with Variable Structure Systems (VSS) approach. Levenberg-Marquardt optimisation method is responsible for minimization of squared error while the VSS based law is responsible for the stability in the parameter change space. Simulation was carried out where the proposed algorithm was used to train a fuzzy controller for a 2-DoF robotic manipulator. Results show that the fluctuations that are most likely to occur during Levenberg-Marquardt training are damped out.

Hagras et al. (2002, 2004) presented a novel Fuzzy-Generic technique for the online learning and adoption of a fuzzy controller which can be applied to an intelligent robotic navigator. Fuzzy-Genetic is a life-long learning technique that enables the robot to navigate in changing environments where it adapts itself to the environment by tuning the controller rules that did not perform well.

The design of the fuzzy system can be formulated as a search problem in high dimensional space where each point represents a rule set, membership functions, and the corresponding system behaviour. Given some performance criteria, the performance of the system forms a hyper-surface in the space. The hyper-surface is nondifferentable since changes in the fuzzy sets are discrete and can have a discontinuous effect on the fuzzy system's performance. GA is guided by the principles of natural evolution and genetics. It is not based on gradient information and has no continuity or convexity requirement on the solution space. Therefore it is a better candidate for searching the hyper-surface than conventional hill climbing search methods. However, GA takes a large number of iterations to develop a good controller. Thus it is not feasible to learn online and adapt in real-time. To tackle this problem, the initial population will be set to the solutions of previously solved problems. In this case the GA does not have to

waste time exploring unpromising subspaces so that the searching efficiency is greatly improved. The technique has been verified in difficult domains such as robot navigating in unstructured environments.

Garcia et al. (2009) proposed an adaptive Fuzzy Inference System (FIS) to solve the problem of mobile robot path planning. The selection of the optimal path relies on the criterion of the FIS, which is adjusted using a Simple Tuning Algorithm (STA). The proposed path planner support static and dynamic obstacle avoidance. To determine the optimal path, the cost of a path is evaluated by the FIS, which considers not only the length of the path but also the difficulty for the navigation. The FIS has two inputs: Effort, and Distance. The first one represents the energy spent by the robot to make turns across the path. Distance is the accumulated Euclidean distance of the path. The output is a weight assigned to the cost of the path. The more weight is given, the path becomes less desirable. The output of the FIS is added to the total Euclidean distance of the path. If there are different route with the same length, the FIS should give preference to the straighter path. The FIS can be tuned for a better performance using the STA. by applying the STA, time and effort are reduced to a single parameter using a tuning factor k. The FIS behaviour can be modified by manipulating the ranges of the membership functions of the input variables. The STA method consists of four steps:

1) **Tuning factor selection:** A number $k \in [0, 1]$ is used to define the tuning adjustment level. $k = 0$ is the biggest settling time and $k = 1$ is the smallest.
2) **Normalisation of the ranges of the input variables:** The range of each input variable is modified in order to have the lower and upper limits equal to $-1$and $+1$.
3) **Tuning factor processing:** Once the range is normalised, the new vector of operation points will be given by:

$$V_{op_{final}} = (V_{op_{initial}})^{r(k)} \tag{2.8}$$

where$V_{op_{initial}}$ is a vector with normalised values of the membership in the X-axis and $r(k)$ is a polynomial of k.

4) **Renormalisation of the ranges of the variables:** Convert the normalised range to the previous range.

The tuning process of the FIS applying the STA falls into making a decision about the tuning factor *k*. The adequate selection of the k value is necessary to obtain the desired settling time for the system.

Desouky and Schwartz (2010) addressed the problem of tuning the input and output parameters of a fuzzy logic controller. Two techniques are proposed. The first technique combines Q-learning with function approximation to tune the parameters of a fuzzy logic controller operating in continuous spaces. The second technique combines Q-learning with genetic algorithm to tune the parameters of fuzzy logic controller in discrete spaces. The proposed techniques are especially useful for tuning the input or output parameters of a fuzzy controller when the system model is partially or completely unknown and it is hard or expensive to get training data or a teacher to learn from. The learning process in the Q-learning Fuzzy Inference System (QLFIS) is performed simultaneously. The FIS is used as a function approximation to estimate the optimal action-value function, $Q^*(s,a)$, in continuous state and action space while the Q-learning is used to tune the input and output parameters of the FIS. In the Q-Learning Based Genetic Fuzzy Controller (QLBGFC), the learning process is performed sequentially. First, in phase 1, the state space are discretized and Q-learning is used to obtain an estimation of the desired training data set, (s,a^*). Then this train data set is used by genetic algorithm in phase 2 to tune the input and the output parameters of the FLC which is used at the same time to generalise the discrete state and action values over the continuous state ad action space. The techniques are tested in a pursuit-evasion game simulation. The authors state that the proposed techniques outperform all the other techniques which include Q-learning only, reward-based GA and neural network based Q-learning in performance and learning time.

2.3 Summery

In this chapter, the state of the art approaches for the grounding of symbolic object manipulation commands to specific robot configurations in terms of robot base poses or arm joint configurations are summarised. The method used includes: optimisation based method, analytic model based method, vision based method, statistic based method and experience based method. Uncertainties in determining the target object location or robot base pose are dealt with statistic based methods and experience based methods. The performance of the approaches is evaluated by computation time, whether obstacles are considered, whether human intervene is need and whether uncertainties are considered. Techniques for learning adaptive fuzzy systems and their applications are also introduced in this chapter.

CHAPTER 3 FUZZY REACHABILITY SPACE

This chapter describes the concept of fuzzy reachability space and the process of constructing such a space for the purpose of symbolic grounding. Symbolic grounding requires to handle uncertainties when converting linguistic commands, also known as task-level commands, to robot poses. The uncertainties are considered as the nature of linguistic commands, which express the elasticity of concepts. They also exist in robots' understanding about environments. Due to the unstructured and the dynamic features of the environments where service robots work, it is often hard for the robots to have the exact and complete information of the environment. Sensor readings, which can have measurement errors, can provide imprecise information about the size, shape and location of objects in the environment. Objects can be introduced in or removed from the environment randomly. More important, the uncertainties exist among robot poses. Due to limited sensing and actuating capabilities, it is often the case where there are differences between the actual target pose and the one that the robots eventually reached. In some cases, it can take the robots a long period of time to reach a target pose even if they are able to do so because of the dynamic feature of the environment where they work.

In the real world, given linguistic commands, human can handle well the uncertainties using human reasoning. Fuzzy logic is one of the methods that mimics human reasoning and, therefore, is the natural candidate to handle the uncertainties in the process of symbolic grounding. The fuzzy reachability space represents a fuzzy relation from robot poses to the extents to which a robot comfortably reaches an object given the inexact and incomplete environmental information. Performing fuzzy reasoning in the fuzzy reachability space can produce the optimal robot pose.

3.1 High Dexterity Zone

3.1.1 Definition of HDZ

The HDZ of a robot is defined as the areas (in the robot coordinate system) that can be reached with the robot arm (Meßmer, 2010). Transferred to the problem described in this research, this means that given a goal pose for the robot arm (location and orientation) an Inverse Kinematic (IK) solver is more likely to find a configuration if the robot base is placed at a pose that makes the goal pose lies within this HDZ. For example, the annular-shaped area illustrated in Fig. 3.1 is one section of the HDZ of a service robot platform Care-O-bot 3 (manufactured by Fraunhofer IPA). Colours show the degrees to which the robot could easily reach an object. A set of such kind annular-shaped areas form the HDZ.

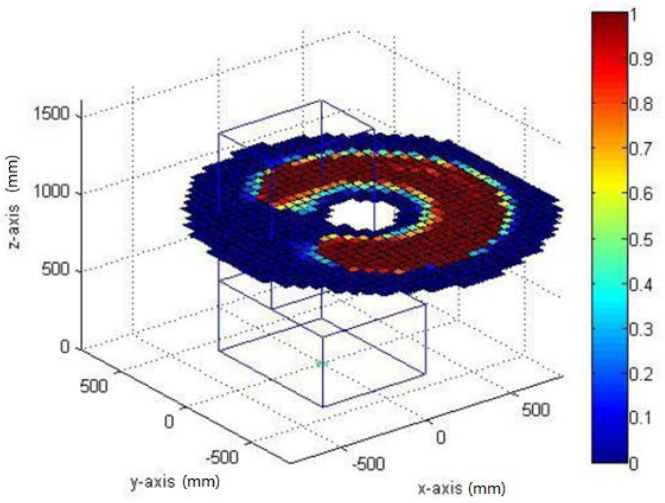


Figure 3.1 HDZ of Care-O-bot 3 robot

- The HDZ of a robot can be computed in the way described below:Set an altitude.

- Partitioning the workspace of the robot into a grid and then using each grid cell as a goal end-effector location. Each end-effector location is amended by six orientations.
- Checking each of the six end-effector poses (location and orientation) using the IK-solver for whether it is reachable or not (i.e. an arm configuration can be calculated). The more end-effector poses are reachable, the higher dexterity of the corresponding grid cell.
- Taking into account of the grid cell in question as a part of a HDZ if it can be reached using all of the six end-effector poses.
- Repeating the last three steps for all altitude.

However, this method does not consider self-collisions. Any two parts of the robot's body may collide into each other when moving the robot arm to a goal pose. In addition, this method is not designed in purpose for object grasping. The HDZ obtained only reflects whether a robot arm can reach an object but does not guarantee the robot arm can grasp the object.

Since this research focuses on symbolic grounding for object grasping, the dexterous workspace of the robot for grasping objects needs to be taken into consideration of determining a HDZ. The workspace depends on the way of a robot grasping an object, known as grasp type (Meßmer, 2010). There are four kinds of grasp types. They are classified according to the location and orientation of the robot's gripper with respect to the location of the target object:

- **Top grasp:** gripper approaching an object from above (see equation 3.1)
- **Front grasp:** gripper approaching an objectfrom front (see equation 3.3)
- **Left grasp:** gripper approaching an object from left (see equation 3.4)
- **Right grasp:** gripper approaching an object from right (see equation 3.5)

The classification of the grasp types is illustrated in Fig. 3.2.

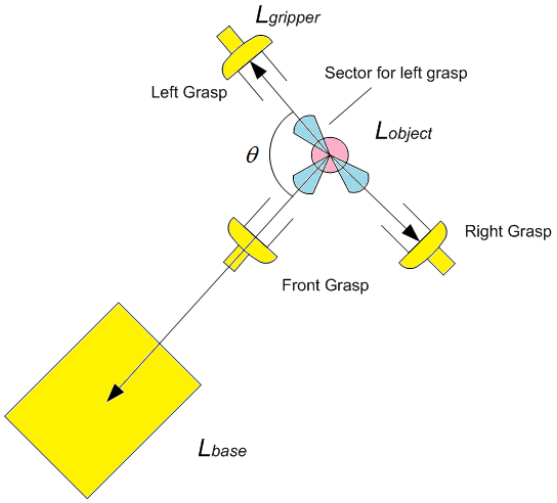


Figure 3.2 Grasp classification

The top grasp type is defined as:

$$gripperPose.Z \geq objectLocation.Z + \alpha \tag{3.1}$$

whereZ is the vertical axis of the 3D workspace of a robot and α represents an adjustable parameter, depending on the type of the robot.

To distinguish among the front grasp type, the left grasp type and the right grasp type, the angle θ between the two vectors:

$$V_{gripper} = L_{gripper} - L_{object}$$

and

$$V_{base} = L_{base} - L_{object}$$

is calculated using the dot product:

$$\theta = \arccos\left(\frac{V_{gripper} \cdot V_{base}}{|V_{gripper}||V_{base}|}\right) \tag{3.2}$$

where $L_{gripper}$, L_{base} and L_{object} stand for the location of a robot gripper, the robot base and a target object, respectively. $V_{gripper}$stands for the vector from the location of the object to the location of the robot gripper. V_{base}stands for the vector from the location of the object to the location of the robot base. If the angle

θ lies within a certain sector (from $-10°$ to $+10°$), the grasp is assigned to the respective basic grasp type:

$$|\theta| \leq 10° \rightarrow graspisclassifiedasafrontgrasp \quad (3.3)$$

$$80° \leq \theta \leq 100° \rightarrow graspisclassifiedasaleftgrasp \quad (3.4)$$

$$260° \leq \theta \leq 280° \rightarrow graspisclassifiedasarightgrasp \quad (3.5)$$

If the robot can reach a target object using all of the four grasp types from a robot base pose, the location of the object corresponding to the robot base pose is considered high dexterity. With respect to each kind of grasp types, the dexterity workspace of the robot arm can be identified. For example, the workspace for the front grasp type is the area where the robot arm can reach by using the front grasp. After the workspaces for the four grasp types are identified, the High Dexterity Zone for grasping objects (HDZ-g) of the robot can be obtained by intersecting the four workspaces. In compare with the HDZ, the HDZ-g of the robot is obtained by taking into account the self-collision of the robot parts, the detecting range of the laser scanner on the robot head and more importantly, the grasping of objects. The target object can be grasped by using four different basic grasp types when the robot base is placed at a pose that makes the target object lies within the HDZ-g. The altitude of target object locations is also considered. When the altitude of an object exceeds a certain range, it will be rejected from further processing.

3.1.2 Identifying HDZ-g for grasping objects

HDZ can be used for the purpose of grasping objects. A simulation scenario was designed where a robot tries to grasp an object that is placed at different locations on a table, as shown in Fig. 3.3. The robot was placed at the origin of a world coordinate, i.e. (0, 0) in X and Y-axis. The orientation of the robot is the same as X axis. A table was placed in front of the robot within the view of the robot. The surface of the table was discretized into a number of grid cells.

The process of milk box grasping simulation can be described as:

- Placing the object at the centre of a grid cell.

- Choosing a grasp type from the four grasp types.
- Calling object detector to extract the location of the object.
- Calling IK-solver to generate a robot arm configuration for reaching the object.
- Moving the robot's gripper to the object.
- Grasping the object.
- Collecting the result of whether the object was successfully grasped.
- Moving the object to the next grid cell.
- Repeating until all grid cells were visited.
- Repeating all steps until all four grasp types was used.
- Calculating intersection.

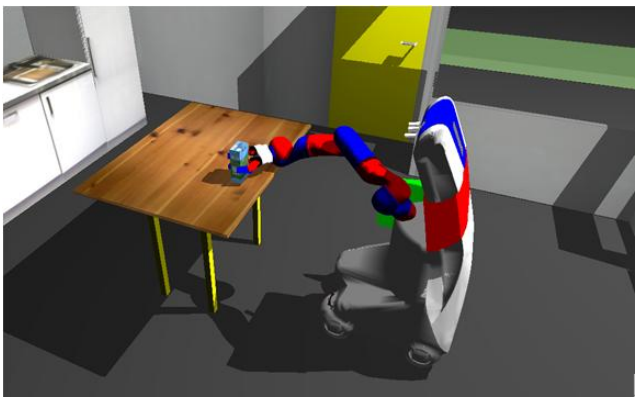


Figure 3.3 Grasping object simulation

Care-O-bot 3, as shown in Fig 3.4, is a service robot platform. It was developed in 2008 and is equipped with the latest state of the art industrial components including omni-directional drives, a 7-DOF redundant manipulator, a three finger gripper and a flexible interaction tray that can be used to safely pass objects between the human and the robot. Its moveable sensor head contains range and image sensors enabling autonomous object learning and detection and 3D supervision of the environment in real time (Graf et al., 2009, Reiser et al. 2009).

Figure 3.4 Care-O-bot 3

Applying the method to Care-O-bot 3 under Robot Operating System (ROS) environment, (Details can be found in Koenig and Howard, 2007, Quigley, et al., 2009 and http://www.ros.org/wiki/ROS/Introduction) yields the following dexterity workspaces.

x\y (m)	-0.2	-0.15	-0.1	-0.05	0	0.05	0.1	0.15
-1								
-0.95								
-0.9								
-0.85								
-0.8								
-0.75								
-0.7								

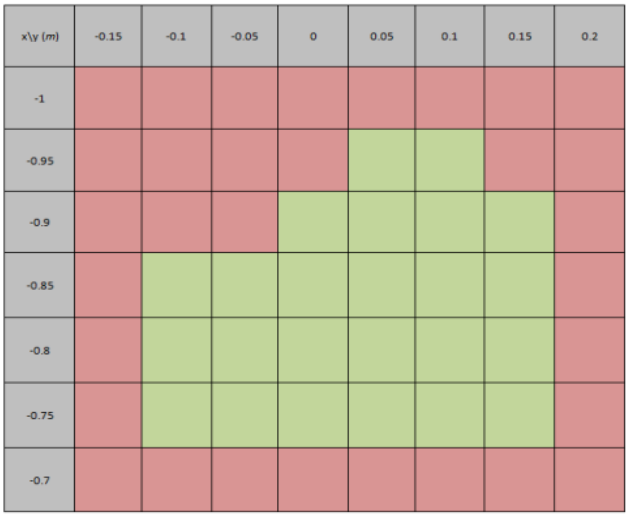
(a) Top grasp

x\y (m)	-0.3	-0.25	-0.2	-0.15	-0.1	0.05	0	0.05
-1								
-0.95								
-0.9								
-0.85								
-0.8								
-0.75								
-0.7								

(b) Right grasp

x\y (m)	-0.2	-0.15	-0.1	-0.05	0	0.05	0.1	0.15
-1								
-0.95								
-0.9								
-0.85								
-0.8								
-0.75								
-0.7								

(c) Front grasp



(d) Left grasp

Figure 3.5 Workspaces of the robot arm for the four basic grasp types

The first columns and the first rows of the diagrams show the *X* and *Y* values of the object locations with respect to the robot base pose. The robot base's location is (0, 0)in the world coordinator and the orientation is the same as X-axis. The grid cells in green colour show the workspace of the robot arm (i.e. the areas that the robot arm can reach). The grid cells in red colour show the area that the robot cannot reach. The reasons for some of the areas cannot be reached include: 1) the areas are out of reach of the robot arm; 2) the areas are two close to the robot base and there is no valid arm configuration can be found; 3) the area is out of the view of the object detector.

The HDZ-g of the robot was obtained by intersecting the four workspaces. When a target object is placed within the intersected area, it is guaranteed that the object can be reached by the robot arm using four basic grasp types. The HDZ-g of the robot which is determined by taking into account the specific characteristics of Care-O-bot 3 (i.e. self-collision of the robot body parts, detecting range of the laser scanner and object grasping) is illustrated in Fig. 3.6.

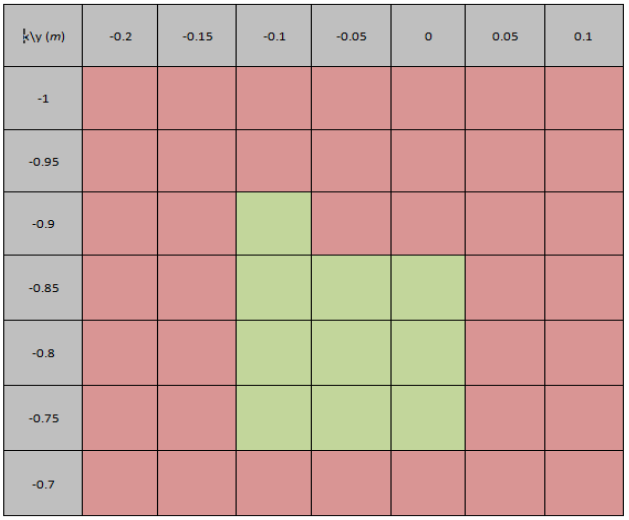


Figure 3.6 HDZ of the robot

The area in green colour shows the HDZ-g of the robot. The first column and the first row show the X and Y value of the target locations. The HDZ-g is a small area in front of the robot base. The location and orientation of the HDZ-g is fixed to the robot base pose. The HDZ-g of the robot can be further used to decide the optimal base pose for grasping a target object. The reason for defining the HDZ-g as the intersected area of the four workspaces is that when the robot is determining the base pose for grasping an object, it does not know which grasp type it will use. After the robot moves to the base pose, it will perform detection for the actual target object position and select a grasp type afterwards. The use of the intersected area of the workspaces will guarantee that the object can be successfully grasped by using the selected grasp type.

3.1.3 Optimal grasping location

Some of grid cells within the HDZ-g are workable, but not the optimal. For example, if a target location is near to the edge of the HDZ-g, there will be fewer trajectories generated by the IK-solver. It can also be seen that if the target location is far away from the robot base, the robot arm will need to move a long

distance to reach the location and thus, it is more likely to face obstacles on its path. In addition, it is more difficult to satisfy a required time constraint. Therefore, given the location of an object, there will be a certain location within an HDZ-g from which the robot arm can most comfortably reach the target object. This location is called the optimal grasping location. The optimal grasping location is defined based on the following criteria:

- It is located within the HDZ-g
- It is not too close to the edge of an HDZ-g
- It is not far away from the robot base.

In the HDZ-g obtained from the simulation, the location marked by a red-cross is defined as the optimal grasping location as shown in Fig. 3.7.

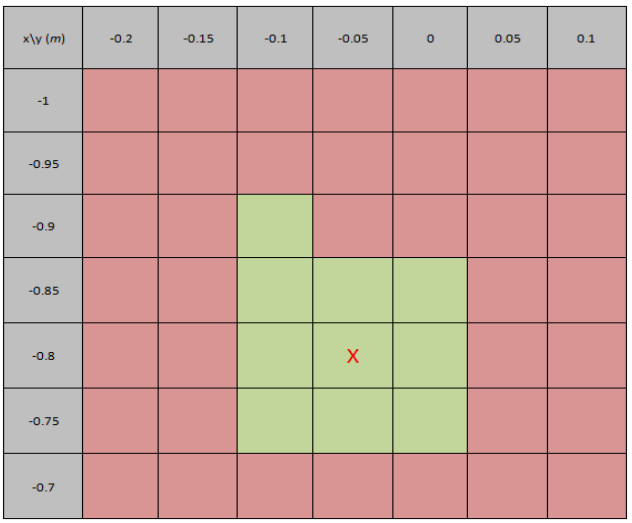


Figure 3.7 The optimal grasping location

The optimal grasping location is also fixed to the robot base pose. That means when the target object location is given, a set of robot base poses can be calculated by placing the optimal grasping location on top of the target object location. These

robot base poses form an annulus around the target object location. An optimal robot base pose for grasping an object can be optimised from these poses.

3.2 Fuzzy Reachability Space

Although an HDZ-g can be defined to represent how easy a robot can grasp an object from given robot base poses and special information can be obtained, as described in the last two sections respectively, the uncertainties mentioned at the beginning of this chapter are not handled by the HDZ and the spatial information. This section introduces the concept of Fuzzy Reachability Space (FRS) that deals with the uncertainty and enables the use of human reasoning in symbolic grounding. This section also describes the way of constructing an FRS.

3.2.1 Spatial information

To be able to construct FRS, perception has to take place by robots and the relevant spatial information of objects in the environment has to be extracted from the sensor data. In order to have this information, laser scanners or 3D cameras are often used to build point maps and geometric shapes of the objects in the environment are extracted from the point maps (Rusu et al., 2009, May et al., 2008, Arbeiter, Hagele and Verl, 2011). A 3D representation of the environment is established based on available point cloud data. This approach can be applied to any robot platform with laser scanners. The process of establishing the 3D presentation of the environment is illustrated in Fig. 3.8.

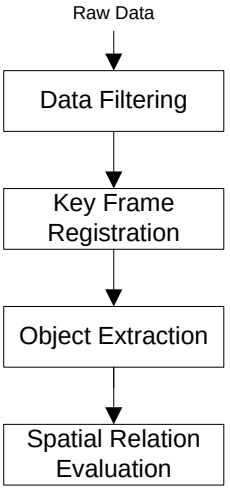


Figure 3.8 Process of establishing 3D representation of environment

First, the point cloud data has to be filtered to reduce noise and redundant sensor information. In the second step, the transformation error between two point clouds is minimised in an iterative way. The registered sensor data is then transformed to the object extraction component that performs planar decomposition in the third step. By evaluating spatial relations of the planes in the last step, the context information about pose and shape of the objects in the environment is extracted. The extracted pose and shape information can be used to construct FRS and model fuzzy constraint functions for performing the fuzzy optimisation, which will be discussed in the following chapters.

3.2.2 Concept of FRS

A fuzzy reachability space describes how easy a robot can grasp an object from different robot base poses. The extent of the easiness depends on the pose of the robot, in terms of location and orientation, relative to the targeted object.

Definition 3.1 (fuzzy reachability space):

A fuzzy reachability space (FRS) is a fuzzy mapping in a 3D space from a set of relative robot base poses to the location of a target object to the extent of the easiness for the robot to grasp the object, that is,

$$FRS \triangleq C \wedge O \rightarrow R \tag{3.6}$$

where C and O are fuzzy variables representing the relative distance and the orientation of the robot base to the target object, respectively, and R is also a fuzzy variable standing for the reachability of the robot to the object. The fuzzy variables C, O and R have fuzzy values that are defined as fuzzy sets.

A FRS is a typically ring-shape surface in a 3D space, as illustrated in Fig. 3.9. Each point of the surface is modelled by a tuple, (x, y, θ, *reachability*), where (x, y, θ) represents a robot base pose and *reachability*, taking values from 0 to 1, indicates how easily a robot can grasp the object from that pose. When the reachability value increases, there will be more valid robot arm configurations for grasping the object. Therefore, it is easier for the robot to find a comfort arm configuration. When the reachability value reaches 1, the corresponding robot base pose is considered to be one of the optimal grasping poses. When the reachability value is 0, the target object cannot be reached by the robot from the corresponding base pose. The reachability value of a robot base pose is deduced from the fuzzy mapping giving in equation (3.6). The fuzzy reachability space is defined according to the specific kinematic characteristics of the robot. Therefore, its shape and size remain constant for different target objects. However, the location of the fuzzy reachability space changes according to the target object location.

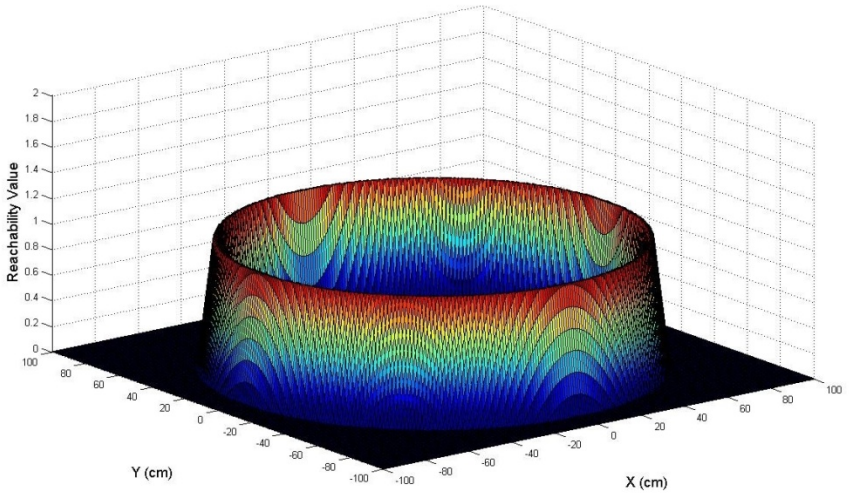


Figure 3.9 An example of FRS

In FRS, relative robot base poses to a target object are represented using two fuzzy sets, namely *C* and *O*. This fuzzy expression of a robot base pose employs regions which do not have a clear-cut boundary but not exact points to represent the robot's location and orientation. The reachability over such a region is the one that a robot can reach when being in the region but not necessarily at a certain point. This means, robots do not have to reach the exact pose and do not have to know the exact location of a target object in order to grasp the object. Therefore, the use of the fuzzy regions compensates against the impreciseness existing in spatial information and robot pose information resulted in the limited sensing and actuating capabilities of robots.

The FRS also enables human reasoning in determining the optimal poses. It is true that humans can comfortably grasp objects not necessarily from a specific location and can grasp the same object that steady stays at a location from different poses. FRS mimics human reasoning by assigning the same reachability over the poses in the same fuzzy region, allowing the poses to have reachability at the same optimal level. It also shows the reachability at different optimal levels for different regions.

3.2.3 Construction

Building up an FRS involves the following steps:

- Defining fuzzy sets to represent a relative robot pose, in terms of the distance and the orientation of a robot base, to a target object.
- Defining a fuzzy set of reachability.
- Performing fuzzy reasoning from the fuzzy sets of robot base to deduce an FRS.

As any target object is placed on an *X*-*Y* plane, the fuzzy set *C*, representing the relative location between a robot and a target object or how close the robot is towards the object, can be defined as:

$$C = \{\frac{membershipdegree}{locationoftheobjectw.r.t.therobotin\ (x,y)}\} \tag{3.7}$$

The location of a target object with respect to the location of robot base is defined in an X-Y plane. The membership degree is decided by the HDZ of a robot. When the robot base is placed at a location that makes the target object location lies at the optimal grasping location, the membership degree is 1. While the target objectmoves away from the optimal grasping location the membership degree decreases accordingly. When the target object location lies outside the HDZ, the membership degree is 0.

Fuzzy set *C* as defined above has a 3D membership function, as illustrated in Fig. 3.10.

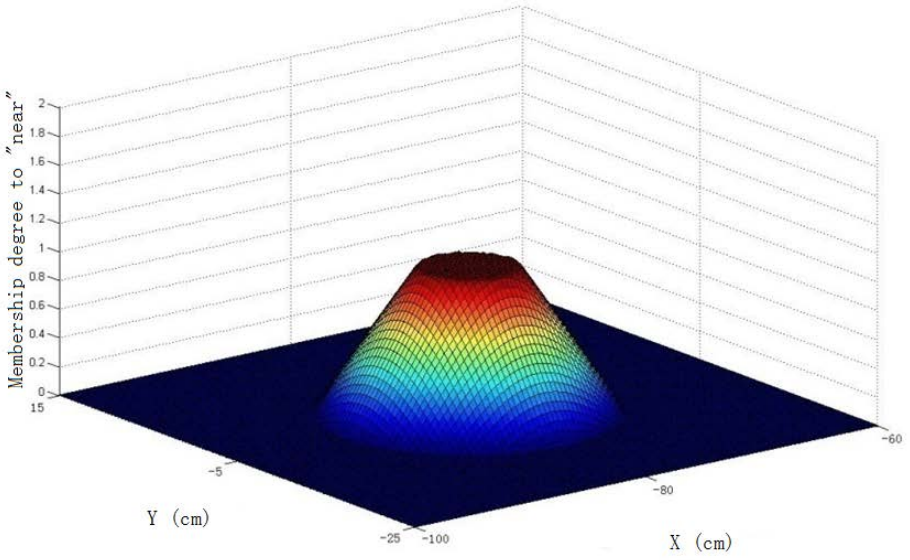


Figure 3.10 Membership function of fuzzy set *C*

Fuzzy set *O* is defined along with the angle, θ, which is the angle between a robot and a target object representing the orientation of the robot relative to the object. This fuzzy set is defined as:

$$O = \{\frac{membershipdegree}{rotation(\theta)}\} \tag{3.8}$$

If the robot is right facing the target object or the orientation θ lies within a certain range, the robot can comfortably reach the object. When θ exceeds this range, it cannot be guaranteed that the robot successfully grasp the object. The membership degree is also decided using HDZ. When θ equals to 0, the membership degree reaches 1. While θ increases, the membership degree decreases accordingly. When θ exceeds the certain range, membership degree becomes 0.

The membership function of the fuzzy set is shown in Fig. 3.11.

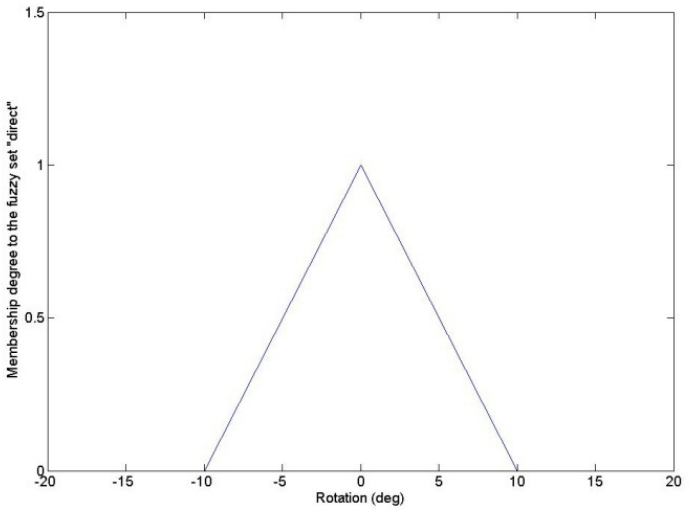


Figure 3.11 Membership function of the fuzzy set *O*

The fuzzy sets, *C* and *O*, give fuzzy reachability with respect to location and orientation, respectively. Given a robot base pose, its reachability is decided by taking into account of both location and orientation at the same time. Fuzzy reasoning serves for this purpose. The fuzzy reasoning, as given in Definition 3.1, is performed over the conjunction of fuzzy set *C* and *O*. The resultant FRS is the one that is shown in Fig. 3.9.

The FRS is constructed by assigning each of the fuzzy regions that represents a collection of robot base poses with a reachability value in the way as described above.

The cross-area of the FRS at the X-Z plane is shown in Fig. 3.12.

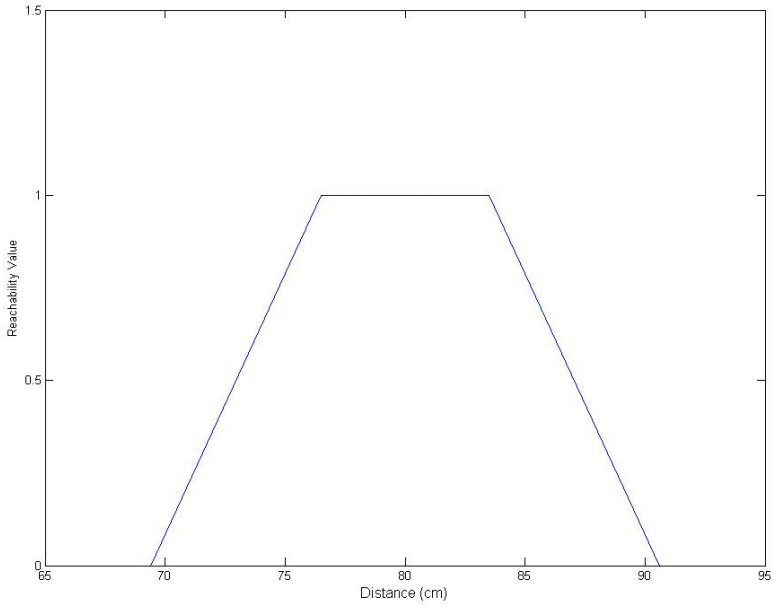


Figure 3.12Cross-area of the FRS at the X-Z plane

In this diagram, X-axis shows the distance between a robot base pose and a target object. The cross-area is a trapezoid. The length of the upper-edge is 7.07*cm*. The length of the bottom-edge is 14.14*cm*. The upper-edge of the trapezoid reflects the fact that when the distance between a robot base pose and a target object is within a certain range, the robot could most easily find an arm configuration to grasp the target object.

The main difference between FRS and HDZ-g is FRS assigns each robot base pose with a reachability value. The reachability value indicates how comfortable the robot can grasp an object from that pose. The optimal robot base pose for grasping an object can be obtained based on the FRS. HDZ-g, on the other hand, is an area located in front of the robot. If the relative target object location w.r.t. the robot base lies within the HDZ-g, the robot can successful grasp the object using four basic grasp types.

3.3 Multi-Layer FRS

As mentioned earlier in this chapter, HDZ-g will change if the altitude of a target object is outside a certain range. Consequently, fuzzy reachability will change, so does the FRS. Combining all FRSs corresponding to all altitude ranges, a multi-layer FRS is formed.

Fig. 3.13 illustrates a 3-layer FRS defined for a robot. On the top layer, FRS is a flat surface with membership degree as 0, because a target object is placed to the altitude that is outside the space the robot can reach. On the middle layer, FRS has the shape similar to that shown in Fig. 3.9. On the bottom layer, FRS shrinks as the object is place too low for the robot to reach.

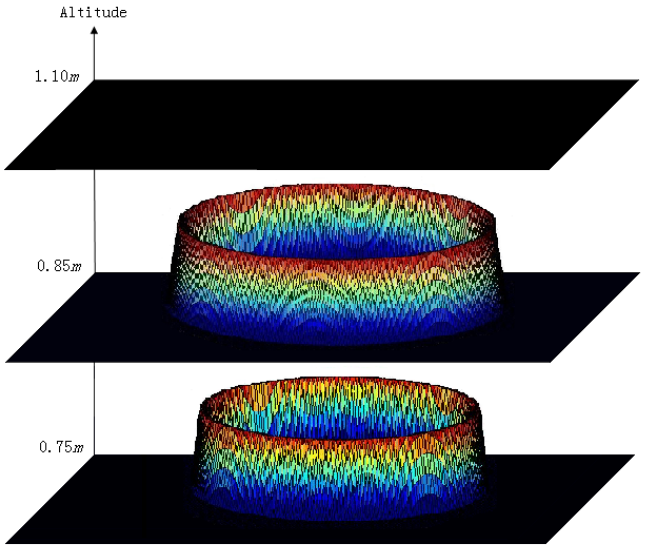


Figure 3.13 A 3-layer FRS

The rest of this section gives details about how to develop a multi-layer FRS for Care-O-bot 3 robot.

The 3D workspace of the robots is divided into four layers, namely, top layer from 1.10*m* and above, higher-middle layer from 0.85*m* to 1.10*m*, lower-middle layer from 0.75*m* to 0.85*m*, and bottom layer from 0.75*m* and below.

As the top and the bottom layers are beyond the height range that the robot can reach, the corresponding FRSs are of plane shape with the height of zero. The FRS for the higher-middle layer is the same as described in Section 3.2. The method of constructing FRS was applied to the lower-middle layer. First, a HDZ-g was identified through simulation where the height of the table where a target object is placed was set to 0.75*m* and the table top was partitioned into grid cells of the size of $25mm \times 25mm$ for the HDZ-g to more accurately match the actual shape of the workspace. The HDZ-g obtained is given in Fig. 3.14.

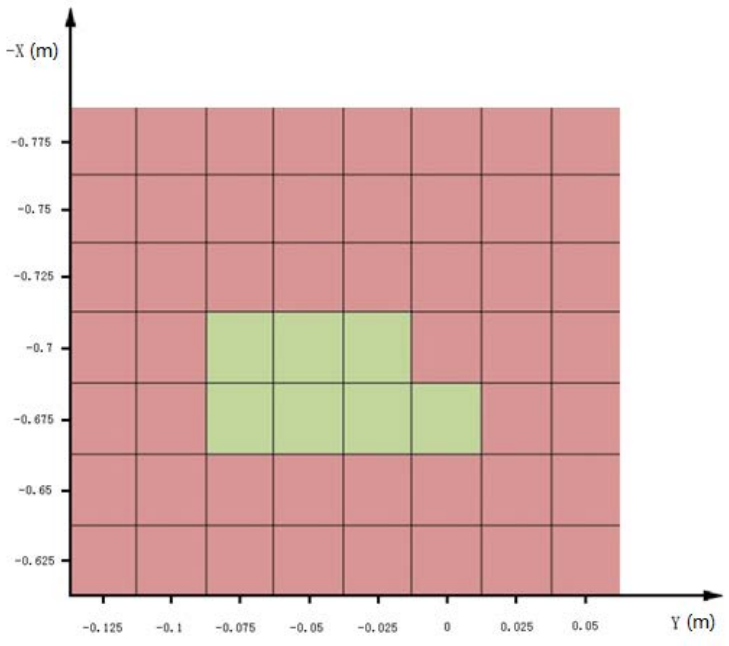


Figure 3.14 HDZ corresponding to the lower-middle layer

Fig. 3.14 shows a world coordinate which is defined by X and Y-axis. The robot was placed at (0, 0) and its orientation is the same as the X-axis. The area in green

colour shows the HDZ-g of the robot. The HDZ-g is located in front of the robot base. The size of the HDZ-g is smaller than the HDZ-g for the higher-middle layer.

After the HDZ-g was identified, the optimal grasping location was defined. When an object is placed at the optimal grasping location, it can be most easily reached by the robot. The same criteria as given in Section 3.1.3 were used for defining the optimal grasping location here. The optimal grasping location is marked with a red cross in Fig. 3.15.

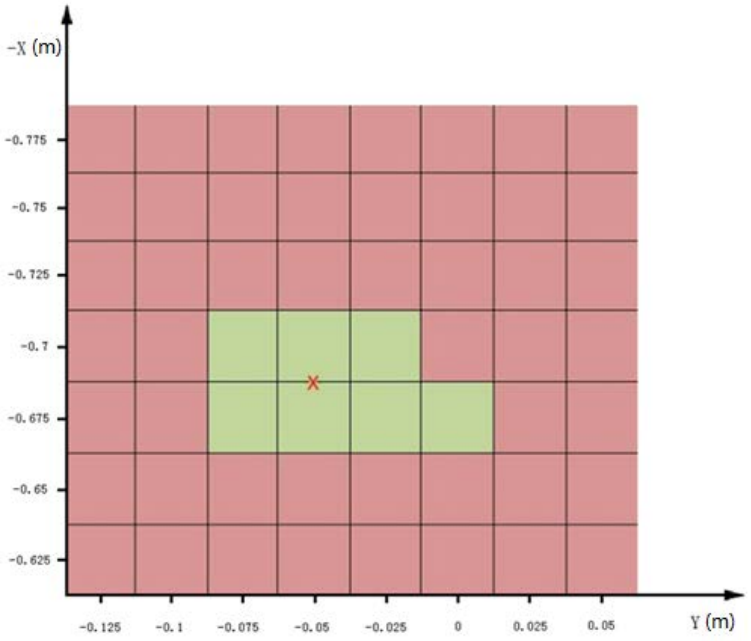


Figure 3.15 Optimal grasping location in the HDZ-g for the lower-middle layer

Fuzzy sets *C* and *O* were defined to model how easily the robot can grasp an object that is placed at the lower-middle layer when the robot is at different poses. The membership function of fuzzy set *C* was defined according to the HDZ-g, as given below and shown in Fig. 3.16.

$$C = \{\frac{membershipdegree}{locationoftheobjectw.r.t.therobotin\ (x,y)}\} \tag{3.9}$$

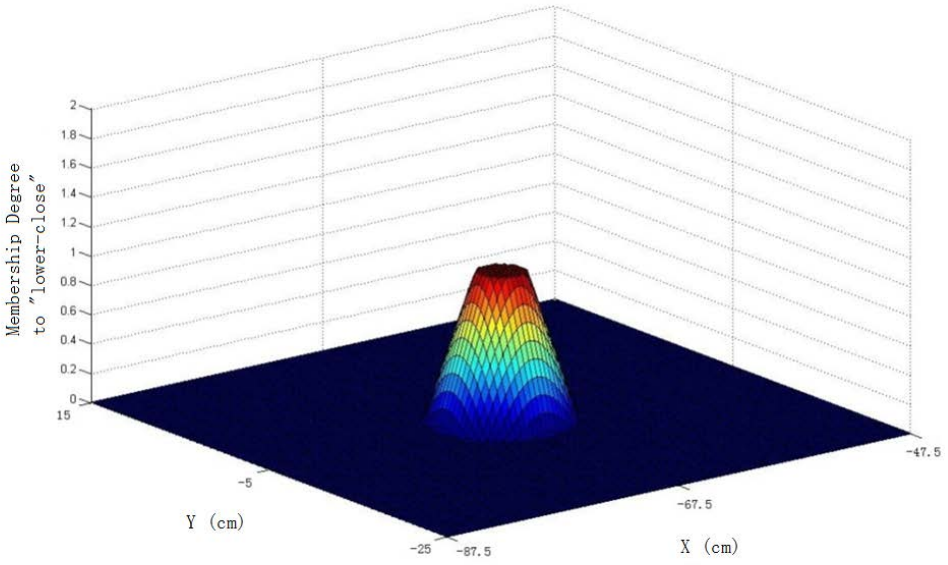


Figure 3.16 Membership function of fuzzy set *C* for the lower-middle layer

In this diagram, the horizontal axes show the location of the target object with respect to the robot base location. The vertical axis shows the membership degree of the fuzzy set.

The membership function of fuzzy set *O* is defined in equation (3.10) and shown in Fig. 3.17.

$$O = \{\frac{membershipdegree}{rotation(\theta)}\} \tag{3.10}$$

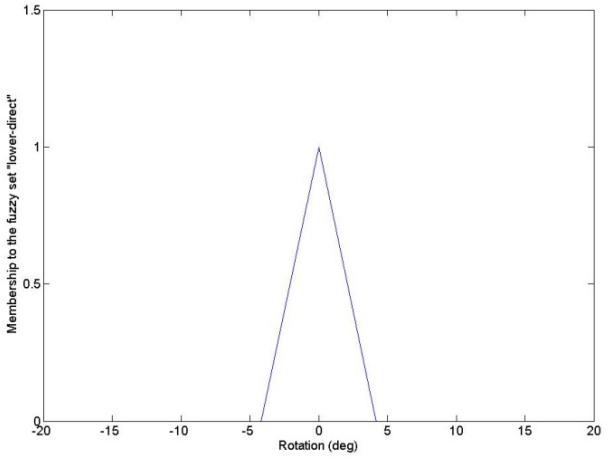


Figure 3.17 Membership function of fuzzy set *O* for the lower-middle layer

The horizontal axis shows the angle between rotation (in *degree*) between the robot's orientation and the orientation of direction from the robot base to the target object. The vertical axis shows the membership degree of the fuzzy set.

The FRS for lower-middle layer was constructed using the fuzzy sets and fuzzy inference, as shown in Fig. 3.18.

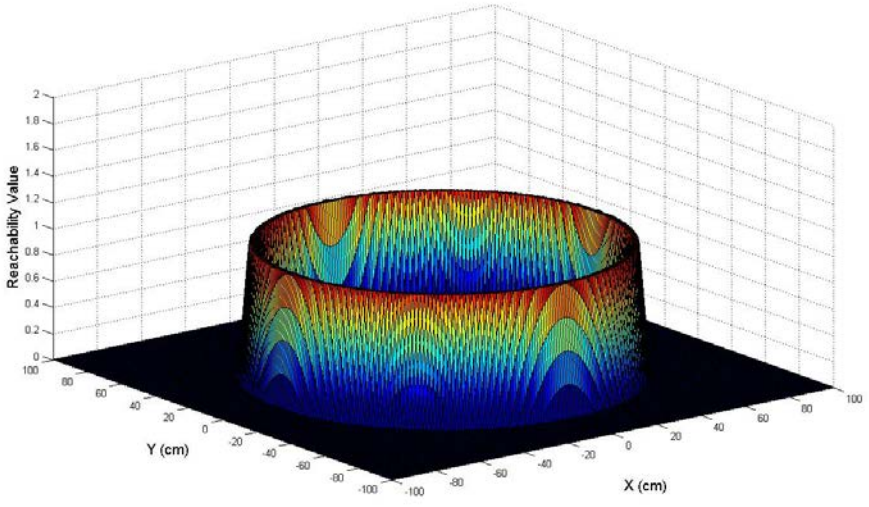


Figure 3.18 Fuzzy reachability space of the lower-middle layer

Given the location of a target object, a suitable HDZ-g needs to be selected according to the altitude of the object. The process of selecting HDZ as well as fuzzy sets *C* and *O* is described in Fig. 3.19.

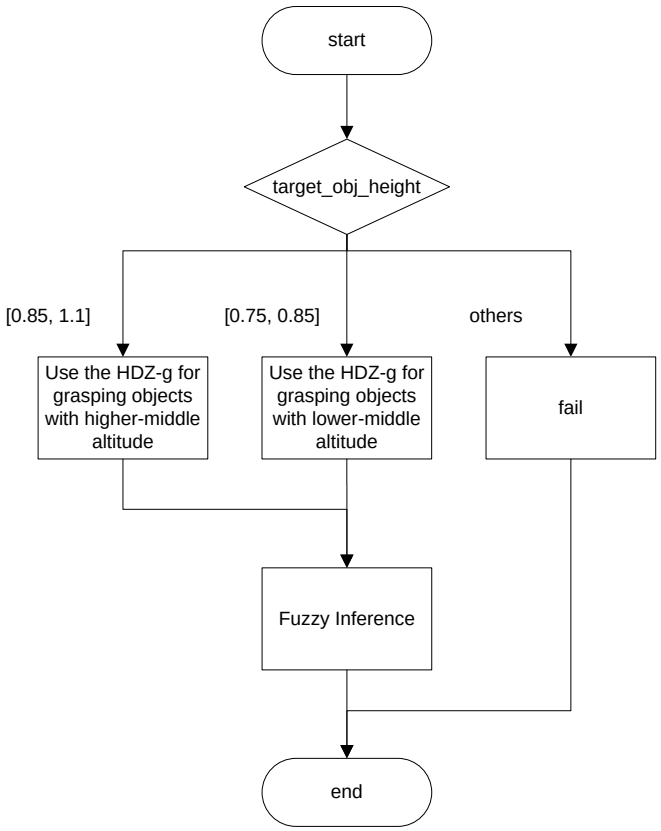


Figure 3.19 HDZ-g and fuzzy sets selecting process

3.4 Summary

In this chapter, the concept and process of constructing FRS are presented. FRS enables uncertainties to be handled and human reasoning be applied in symbolic grounding. Based on FRS, the optimal robot base pose for the robot to grasp a target object can be decided.

CHAPTER 4 SYMBOLIC GROUNDING OVER FUZZY REACHABILITY SPACE

In this chapter, a fuzzy optimisation process is developed for finding the optimal base pose in FRS. Four steps are involved in developing the fuzzy optimisation algorithm: 1) identify design variables; 2) specify constraints; 3) establish an objective function; 4) develop optimisation algorithm. In the first step, design variables should be the desired robot base pose (location and orientation) in the world coordinate. The values of design variables will be evaluated in an objective function to find the optimal solution, that is, a robot base pose from which a robot can most comfortably grasp a target object. In the second step, the constraints are defined according to the influences of obstacles to the reachability value of a desired robot base pose. When the desired robot base pose is close to obstacles, it is more difficult for a robot to navigate to that pose. Some robot arm trajectories may even be blocked. In some cases, the desired robot base pose may be occupied by an obstacle. Therefore, the locations of obstacles related to the desired robot base pose should be used as constraints. Obstacles in the environment cannot be exactly modelled due to uncertainty. Instead, they are modelled as fuzzy constraints which could tolerate the imprecise obstacle spatial information. In the third step, an objective function is defined as a function of the weighted reachability value of the desired robot base pose and the weighted distance between the desired robot base pose and the robot's current pose. The robot base pose that maximise the objective function is the optimal base pose. According to the definition of the objective function, the optimal base pose has the highest reachability value and it is close to the robot's current pose. In the forth step, two optimisation algorithms are developed for finding the optimal base pose in two different cases, namely, unconstrained case and constrained case.

4.1 Unconstrained Fuzzy Optimisation for Symbolic Grounding

When obstacles in the environment are not considered, the problem of finding the optimal base pose based on FRS becomes an unconstrained optimisation problem. An objective function is defined and an optimisation algorithm was developed for determining the optimal base pose. Simulation was carried out to ground and implement an object fetching command for Care-O-bot 3 robot.

4.1.1 Objective function

The objective function was established according to two conditions: 1) the optimal base pose has a high reachability value so that a robot can easily find a valid arm configuration for grasping a target object when it is placed at that pose; 2) the optimal base pose is close to the robot's current pose so that a robot doesn't need to move a long distance to reach that pose. The objective function was defined based on the above two conditions, as a function of the weighted reachability value of a desired robot base pose and the weighted distance between the desired robot base pose and the robot's current pose.

The objective function is defined as:

$$p^* = \underset{p}{\operatorname{argmax}} \left(w_1 * \boldsymbol{R}(p) - w_2 * \sqrt{\left(x_p - x_{current}\right)^2 + \left(y_p - y_{current}\right)^2}/d \right)$$
$$subjectto:\ p \in U. \tag{4.1}$$

where p^* stands for the optimal base pose $(x_{p^*},\ y_{p^*},\ \theta_{p^*})$, argmax stands for the robot base pose for which the given objective function attains its maximum value, $\boldsymbol{R}(p)$ stands for the reachability value of a desired robot base pose p, the reachability value can be obtained from the FRS defined according to the location of the target object, (x_p, y_p) stands for the location of the desired robot base pose in the world coordinate, $(x_{current}, y_{current})$ stands for the location of the robot's current pose, $\sqrt{\left(x_p - x_{current}\right)^2 + \left(y_p - y_{current}\right)^2}$ is the distance between the desired robot base location and the robot's current location, the distance is normalised by dividing it with d, which is the maximum distance between desired

robot base location and the robot's current location, w_1 and w_2 are weight coefficients taking values from 0 to 1, U stands for the universe of discourse, which is the FRS. The weight coefficients indicate the significance of the two conditions, reachability and distance, in determining the optimal base pose. If the reachability value is first considered in determining the optimal base pose, the value of w_1 should be much bigger than the value of w_2. Otherwise, the value of w_2 is set to be much bigger than w_1. The first item of equation (4.1) reflects how a robot base pose meets the first condition. The second reflects how a robot base pose meets the second condition. The objective function was defined as a subtraction of the second item from the first. This is because the robot base pose that maximises the objective function has high reachability value and low distance, which meets the two conditions as mentioned at the beginning of this section.

4.1.2 Unconstrained optimisation algorithm

According to the definition of FRS, a ring-shape area can be found from the FRS. Within this area, all p shares the same highest reachability, as illustrated in Fig 4.1.

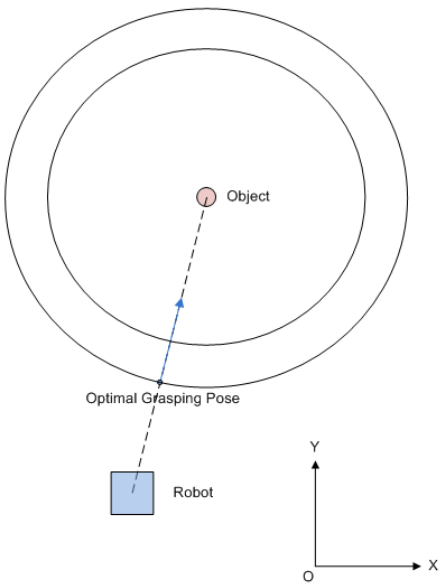


Figure 4.1 Optimal base pose

In the case of unconstrained fuzzy optimisation, the optimisation process becomes a process of searching for the minimal distance from $(x_{current},\ y_{current})$ to a piont within the ring-shape area. This point resides on the line that links $(x_o,\ y_o)$ and $(x_{current},\ y_{current})$. According to Fig. 4.1, this point resides on the out edge of the ring.

The location of the optimal base pose is a point within the ring-shape area that has the minimum distance to the robot's current base location. The orientation of the optimal base pose is the same as the vector from the location of the optimal base pose to the target object's location. Given an FRS and a robot's current base pose, the location of the optimal base pose can be calculated using:

$$x_{p^*} = \begin{cases} x_{current} + (d_{current} - r_{out}) * \cos\theta\,, \\ \qquad d_{current} \geq (r_{in} + r_{out})/2 \\ x_{current} + (d_{current} - r_{in}) * \cos\theta, \\ \qquad d_{current} < (r_{in} + r_{out})/2 \end{cases}$$

$$y_{p^*} = \begin{cases} y_{current} + (d_{current} - r_{out}) * \sin\theta, \\ \qquad d_{current} \geq (r_{in} + r_{out})/2 \\ y_{current} + (d_{current} - r_{in}) * \sin\theta, \\ \qquad d_{current} < (r_{in} + r_{out})/2 \end{cases}$$

$$d_{current} = \sqrt{(x_o - x_{current})^2 + (y_o - y_{current})^2}$$

$$\theta = \text{atan}(\frac{y_o - y_{current}}{x_o - x_{current}}) \tag{4.2}$$

where $(x_{p^*},\ y_{p^*}, \theta_{p^*})$ stands for the optimal base pose in the world coordinate, $d_{current}$ stands for the distance between the robot's current location and the centre of the ring-shape area, r_{in} and r_{out} stand for the inner radius and outer radius of the ring-shape area, $(x_o,\ y_o)$ stands for the centre of the ring-shape area.

The orientation of the optimal base pose can be calculated using:

$$\theta_{p^*} = \text{atan}(\frac{y_o - y_{p^*}}{x_o - x_{p^*}}) \tag{4.3}$$

The process of fuzzy optimisation is shown in Fig 4.2.

In the first step, the target object location is extracted. In the second step, an FRS is established based on the target object location. In the third step, a ring-shape area is identified based on the FRS. Robot base poses within this area have the maximum reachability value. In the forth step, the robot's current location is obtained. In the fifth step, the location of the optimal base pose is calculated using equation (4.2). In the sixth step, the orientation of the optimal base pose is calculated using equation (4.3).

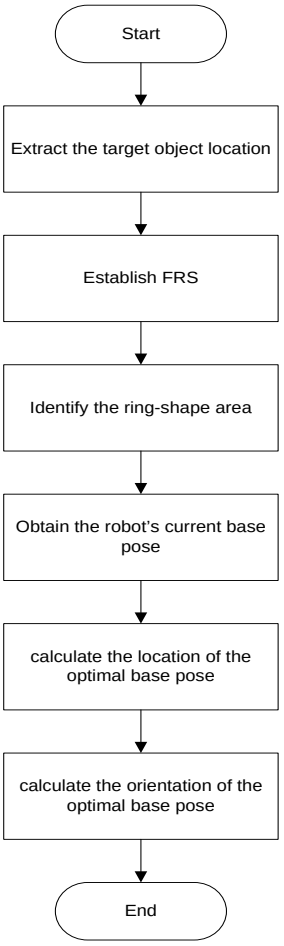


Figure 4.2 Fuzzy optimisation process

4.1.3 Unconstrained symbolic grounding process

When a robot receives an object fetching command *move(base, near, target_object)*, it does not know exactly where the target object is placed. The symbolic term “near” has to be grounded into a preliminary base pose calculated according to an estimated target object location. Due toincomplete prior knowledge of the environment and limited sensor detection range, the estimated object location may not be the actual one. Therefore, after the robot moved to the preliminary base pose, it will have to perform a further detection for having the more accurate location of the object. A reachability value is deduced based on this more accurate location. If the reachability value is above a threshold, the robot will start to grasp the object. If the reachability value is below the threshold, a new optimal base pose, needs to be calculated. After moving to this pose, the robot will start to grasp the object.

The process of symbolic grounding with unconstrained fuzzy optimisation is given as in Fig. 4.3.

In the first three steps, variable values for the symbolic grounding are initialised. These variables include:

- **tar_obj_loc (target object location):** This variable is initially set to an estimated target object location which is provided by a decision making component (an introduction of the decision making component and how the symbolic grounding is integrated with the decision making component will be presented in Chapter 6). Due to the limited prior knowledge of the environment and the sensor detection range, the estimated target object location may be different from the actual target object location. This value needs to be updated through perception of the environment in step 9.
- **weight_coeffs (weight coefficient):** This value is the weight coefficient in the objective function.

- **thres (threshold):** This is a threshold value defined according to the localisation error of the robot's base pose and the object location extraction error. This value is used to decide whether a robot base pose is suitable for grasping the target object.

```
process   UnconstrainedSymbolicGrounding
    1.    tar_obj_loc = est_tar_obj_loc;
    2.    weight_coeffs = predef_weight_coeffs;
    3.    thres = predef_thres;
    4.    current_rbp = getRBP ( );
    5.    frs = getFRS (tar_obj_loc);
    6.    obj_func = getOF (frs, current_rbp, weight_coeffs);
    7.    obp = getOBP (obj_func);
    8.    moveBase (obp);
    9.    tar_obj_loc = getTOL ( );
    10.   current_rbp = getRBP ( );
    11.   frs = getFRS (tar_obj_loc);
    12.   reach = getReach (frs, current_rbp);
    13.   if (reach > thres):
    14.         ret current_rbp;
    15.   else:
    16.         obj_func = getOF (frs, current_rbp, weight_coeffs);
    17.         obp = getOBP (obj_func);
    18.         ret obp;
End
```

(**Meaning ofAbbreviations:** tar_obj_loc/TOL: target object location, est_tar_obj_loc: estimated target object location, predef: predefined, weight_coeffs: weight coefficients, thres: threshold, rbp/RBP: robot base pose, frs/FRS: fuzzy reachability space, obj_func/OF: objective function, obp/OBP: optimal base pose, reach/Reach: reachability)

Figure 4.3 Pseudo-code of unconstrained symbolic grounding process

In step 4, a function called “getRBP ( )” is executed to obtain the robot’s current base pose in the world coordinate. The robot’s base pose is provided by a robot localisation component. In ROS simulation environment, the localisation error for Care-O-bot 3 is up to 5cm.

In step 5, a function called “getFRS ( )” is executed to update the location of the FRS. The FRS was constructed using a fuzzy inference system described in Chapter 3. The centre of the FRS is the same as the target object location and its shape remains constant for different objects.

In step 6, the objective function for performing optimisation is established.

In step 7, a preliminary base pose is calculated based on the estimated target object location. It may need to be updated in the following steps.

In step 8, an action command “moveBase (obp)” is executed to navigate the robot to the preliminary base pose calculated in step 7.

As mentioned in the beginning of this section, the preliminary base pose is calculated according to an estimated target object location. This location may be different from the actual one, which means the preliminary base pose may not be suitable for grasping the target object. Therefore, after the robot moved to the preliminary base pose, detection is taken place to extract the actual object location. In step 9, an action command “getTOL ( )” is executed to extract the actual target object location.

In step 10, the robot’s current base pose is updated through executing the “getRBP ( )” function.

In step 11, the location of fuzzy reachability space is updated according to the extracted target object location.

In step 12, the reachability value of the robot's current pose is obtained from the FRS. If the reachability value is above a threshold, the grounding process is finished and the robot will start to grasp the object from its current base pose. If the reachability value is below the threshold, a new optimal base pose will need to be calculated according to the actual target object location extracted in step 9.

In step 13, the reachability value of the robot's current base pose will be compared with the threshold. A threshold value is defined according to the maximum robot base localisation error and the maximum target object location extraction error. In the situation where the robot's current base pose can be exactly located and the actual target object location can be exactly extracted, the threshold value can be set to zero. This is because according to the definition of FRS, when the reachability value of a robot base pose is above zero, it is guaranteed that a valid robot arm configuration can be found for grasping the target object. However, in real-world task implementation, even if the reachability value of the robot base pose is above zero, the target object may still lies outside the HDZ-g due to the robot base localisation error and the object location extraction error. Therefore the threshold needs to be set according to the following equation:

$$\text{threshold} = \frac{\text{localisation error} + \text{object location extraction error}}{\text{bottom side radius of the membership function of "close"}} \qquad (4.4)$$

If the reachability value of a robot base pose is above the threshold, it can be guaranteed that the target object lies within the HDZ-g and the robot can successfully grasp the object from that pose.

In step 14, if the reachability value of the robot's current base pose is above the threshold, the robot will start to grasp the target object and the grounding process is finished.

If the reachability value is below the threshold, it cannot be guaranteed that the object can be successfully grasped from the robot's current base pose. Therefore, in step 16 and 17, a new optimal base pose is calculated according to the actual

target object location extracted in step 9. After that, the robot will move to the optimal base pose and the grounding process is finished.

4.1.4 Simulations

The aim of the simulations is to test the unconstrained fuzzy optimisation algorithm. Two object fetching simulations were carried out where a Care-O-bot 3 robot tried to fetch an object placed at different locations in ROS simulation environment. In the first simulation, the target object was placed at (-2.2*m*, 0.2*m*) in the world coordinator. The estimated target object location was (-2.2*m*, 0.25*m*). The robot was placed at (0.0*m*, 0.0*m*). An FRS was established and a preliminary base pose was calculated according to the estimated target object location. The preliminary base pose was (-1.41*m*, 0.16*m*, -6.48°). After the robot reached the preliminary base pose, detection was carried out to extract the actual target object location. A reachability value of the robot's current pose and a threshold were calculated (in the simulation, *reachability* = 0.53, *threshold* = 0.47). The reachability value was above the threshold. The robot started to grasp the target object from its current base pose. The FRS and the movement trajectory of the robot base are shown in Fig 4.4.

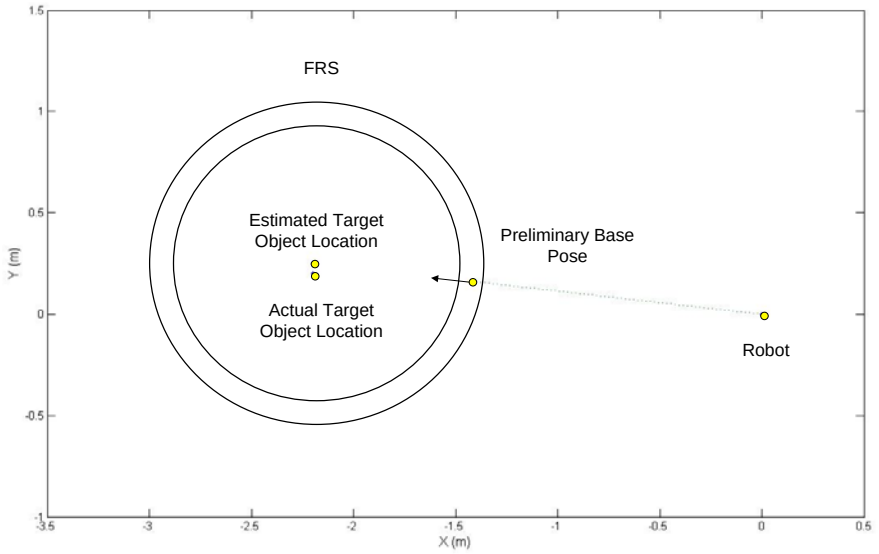


Figure 4.4 The FRS and therobot base movement trajectory to the preliminarybasepose

In this simulation, the estimated target object location was close to the actual target object location. The preliminary base pose calculated according to the estimated target object location has a high reachability value. Therefore, the robot doesn't need to calculate a new optimal base pose. Instead, it started to grasp the object from the preliminary base pose. The object was successfully grasped.

In the second simulation, the target object was placed at (-2.2*m*, -0.3*m*). The estimated target object location was still (-2.2*m*, 0.25*m*). An FRS was established and a preliminary base pose was calculated according to the estimated target object location. After the robot reached the preliminary base pose, detection was carried out to extract the actual target object location. A reachability value of the robot's current pose and a threshold were calculated (in the simulation, *reachability* = 0, *threshold* = 0.47). The reachability value was below the threshold. An optimal base pose was calculated according to the actual target object location. The optimal base pose was (-1.51m, -0.46m, 30.21°). After the robot reached the optimal base pose, it started to grasp the object. The movement trajectory of the robot base to the optimal base pose is shown in Fig. 4.5.

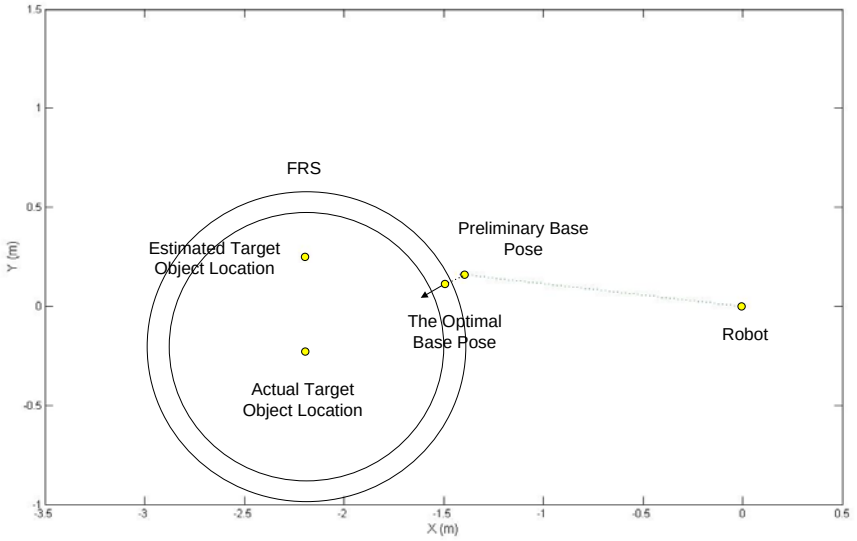


Figure 4.5 The FRS and therobot base movement trajectory to the optimalbasepose

In this simulation, the estimated target object location was far away from the actual target object location. The reachability value of the preliminary base pose was 0. Therefore, a new optimal base pose was calculated according to the actual target object location. After the robot reached that pose, it successfully grasped the object.

4.2 Constrained Fuzzy Optimisation for Symbolic Grounding

Environments where service robots work are often cluttered with obstacles. When obstacles are considered, the optimal base pose grounded using unconstrained optimisation algorithm is no longer suitable. For example, when a robot is placed near to a piece of furniture, it is more difficult for the robot to find a valid arm configuration since its arm movement trajectory is more likely to be blocked. Therefore, robot base poses that are near to obstacles should have lower reachability value. As mentioned in the beginning of Chapter 4, Obstacles in the environment cannot be exactly modelled due to uncertainty. Instead, obstacles' location, size and shape are modelled using fuzzy constraint functions. A fuzzy

constraint is different from traditional constraints. It is a fuzzy set defined by a fuzzy constraint function. Traditional optimisation algorithms are not suitable for finding the optimal base pose under fuzzy constraints. Therefore, a constrained fuzzy optimisation algorithm is developed.

4.2.1 Fuzzy constraints

A fuzzy constraint is normally defined by an r-type function.

Definition 4.1 (fuzzy constraint) (Wang, 1983)
A fuzzy constraint $\tilde{A}$ is a fuzzy subset of U, $\forall\, u \in U$, a value $\mu_{\tilde{A}}(u) \in [0,1]$ is defined. $\mu_{\tilde{A}}(u)$ is the membership degree of u to $\tilde{A}$. The mapping:

$$\mu_{\tilde{A}}: U \to [0,1],$$
$$u \mapsto \mu_{\tilde{A}}(u) \tag{4.5}$$

is the membership function of $\tilde{A}$. U is the domain of discourse.

Fig 4.6 illustrates a fuzzy constraint.

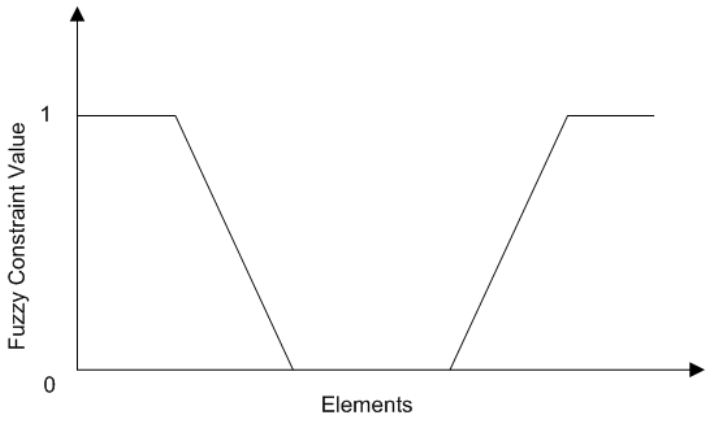


Figure 4.6 Fuzzy constraint

In this diagram, the horizontal axis represents elements of a given domain. The vertical axis represents fuzzy constraint value. The fuzzy constraint value of an

element indicates the degree of influence from the corresponding constraint to the selection of this element. When the fuzzy constraint value increases, the influence decreases accordingly. When the fuzzy constraint value of an element reaches 1, this element is outside the constrained domain (i.e. there will be no influence from the corresponding constraint to the selection of this element). When the fuzzy constraint value of an element is 0, this element is inside the constrained domain (i.e. this element cannot be selected for optimisation).

In order to model obstacles, spatial information of obstacles needs to be extracted. In Section 3.2.1, the method for extracting spatial information of objects in the environment is presented. The information is stored in a knowledgebase in the following format:

---

objects: [‘Table0’, ‘Stove0’, ‘Fridge0’, ‘Sofa0’, ‘Dishwasher0’, Sink0]
objectInfo:
-
l: 0.899999976158
w: 0.899999976158
h: 0.740000009537
pose:
location:
x: 0.649999976158
y: 1.21000003815
z: 0.10000000149
orientation:
x: 0.0
y: 0.0
z: 0.0
w: 1.0

…

The first row is a list of object names. The following is size (length, wide and height) and 6D pose (location and orientation) of the first object in the name list. The unit is *m*. The orientation of an object is presented in the quaternion format. The object spatial information can be retrieved using services provided by a knowledge base component. Through perception of the environment, the information can be updated while task implementation. In this research, all of the furniture in the environment will be considered as obstacles.

By using the object spatial information, a fuzzy constraint function can be defined for each obstacle in an environment. For example, the area surrounding a cylinder-shape obstacle can be modelled as a 3D fuzzy constraint function illustrated in Fig. 4.7.

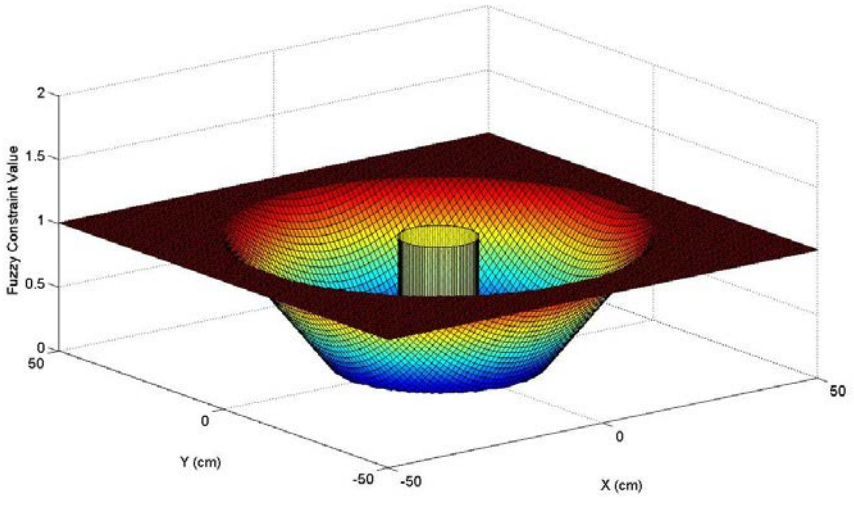


Figure 4.7 Fuzzy constraint function for a cylinder-shaped obstacle

In this diagram, the cylinder in yellow colour is an obstacle. The radius of the cylinder is 7.5cm. The obstacle is placed at (0cm, 0cm) in the world coordinate. The horizontal axes show the robot base locations (along X-axis and Y-axis). The vertical axis shows the fuzzy constraint value. The fuzzy constraint value is 0 within the area occupied by the obstacle. When a robot base location moves away from the obstacle the fuzzy constraint value increases accordingly. This reflects the fact that the obstacle will have less influence to the reachability value of the robot base location. The increasing rate of the fuzzy constraint function is calculated using:

$$\frac{1}{d_{max}} \tag{4.6}$$

where d_{max} is the distance from the centre of the robot base to the maximum reach of the robot. In this research, d_{max} is the same as the inner radius of the ring-shape area, r_{in}, defined in Section 4.1.2. The value of the increasing rate indicates if an obstacle is out of reach of the robot, it will have no influence to the movement trajectory of the robot arm. Therefore, the obstacle has no influence to the reachability value of the robot base pose. When the distance between a robot location and the obstacle exceeds d_{max}, the fuzzy constraint value is 1.

The area surrounding a cubic-shape obstacle can be modelled as a 3D fuzzy constraint function illustrated in Fig. 4.8.

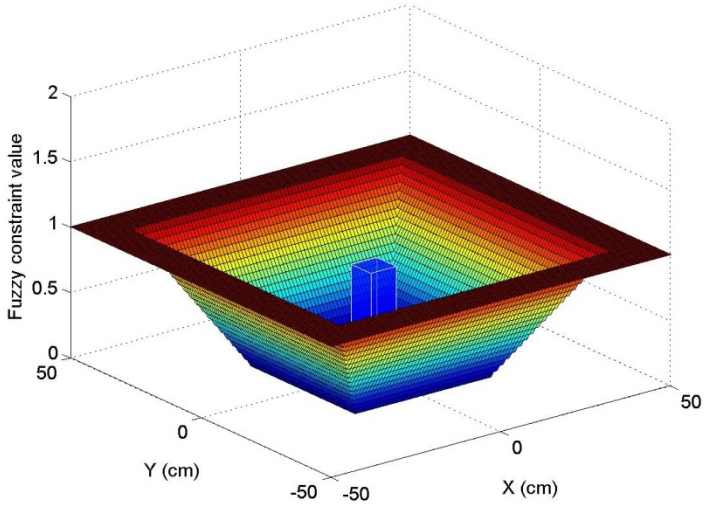


Figure 4.8 Fuzzy constraint function for a cubic-shaped obstacle

In this diagram, the cubic in blue colour is an obstacle. The length is 7.5cm and the width is 7.5cm. The obstacle is placed at (0cm, 0cm). The horizontal axes show the robot base locations (along X-axis and Y-axis). The vertical axis shows the fuzzy constraint value. Similar to the fuzzy constraint functions for cylinder-shape obstacles, the fuzzy constraint value is 0 within the area occupied by the obstacle. When the robot base location moves away from the obstacle the fuzzy constraint value increases accordingly. The increasing rate is calculated using equation (4.6). When the distance between the robot base location and the obstacle exceeds d_{max}, the fuzzy constraint value is 1.

Fuzzy constraint functions model the influences of obstacles to the reachability value of robot base poses. By applying fuzzy constraints, the optimal base pose will be determined within the constrained area (i.e. the area far from obstacles).

4.2.2 Constrained fuzzy optimisation

A constrained optimisation algorithm is developed for determining the optimal base pose under fuzzy constraints. The following definitions are used in this algorithm:

Definition 4.2 (3D fuzzy constraint set)

A 3D fuzzy constraint set $\widetilde{\boldsymbol{C}}$ is a fuzzy subset defined on $\boldsymbol{U}$ by a 3D fuzzy constraint function. $\boldsymbol{U}$ is defined by robot base locations within FRS.

Definition 4.3 (2D cut set)

A 2D cut set $\boldsymbol{C_\lambda}$ is defined as:

$$\boldsymbol{C_\lambda} = \{l \in \boldsymbol{U} | \widetilde{\boldsymbol{C}}(l) \geq \lambda\} \tag{4.7}$$

where λ is a threshold, taking values from 0 to 1. $\widetilde{\boldsymbol{C}}(l)$ is the membership degree of a robot base location,l, to the 3D fuzzy constraint set $\widetilde{\boldsymbol{C}}$.

Definition 4.4 (2D maximising set)

For all $\lambda \in [0, 1]$, a 2D cut set $\boldsymbol{C_\lambda}$ can be defined for a 3D fuzzy constraint set $\widetilde{\boldsymbol{C}}$. The 2D maximising set $\boldsymbol{M_\lambda}$ is the subset of $\boldsymbol{C_\lambda}$. Robot base locations within $\boldsymbol{M_\lambda}$ have the maximum reachability value, $\boldsymbol{R}(l \in \boldsymbol{M_\lambda}) = 1$. Another two sets are defined:

$$\boldsymbol{M} \triangleq \cup_{0<\lambda\leq 1} \boldsymbol{M_\lambda} \tag{4.8}$$

$$\overline{\boldsymbol{M}} \triangleq \cup_{0\leq\lambda\leq 1} \boldsymbol{M_\lambda} = \boldsymbol{M} \cup \boldsymbol{M_0}. \tag{4.9}$$

Definition 4.5 (3D fuzzy maximising set)

A 3D fuzzy maximising set $\widetilde{\boldsymbol{C}}_{\boldsymbol{R}}$ is defined as:

$$\widetilde{\boldsymbol{C}}_{\boldsymbol{R}} \triangleq \cup_{0<\lambda\leq 1} \lambda \boldsymbol{M_\lambda} \ (= \cup_{0\leq\lambda\leq 1} \lambda \boldsymbol{M_\lambda}) \tag{4.10}$$

where $\lambda \boldsymbol{M_\lambda}$ is a fuzzy subset defined on $\boldsymbol{U}$, its membership function is:

$$(\lambda \boldsymbol{M_\lambda})(l) \triangleq \lambda \wedge \boldsymbol{M_\lambda}(l) = \begin{cases} \lambda, & l \in \boldsymbol{M_\lambda} \\ 0, & l \notin \boldsymbol{M_\lambda} \end{cases}. \tag{4.11}$$

$\widetilde{\boldsymbol{C}}_{\boldsymbol{R}}$ is a fuzzy subset of the fuzzy constraint set $\widetilde{\boldsymbol{C}}$.

Definition 4.6 (2D fuzzy maximising value)

A 2D fuzzy maximising value $\boldsymbol{R}_{\tilde{c}}$ is a fuzzy subset defined on the domain of reachability value, taking values from 0 to 1. Its membership degree to the fuzzy constraint set is:

$$\boldsymbol{R}_{\tilde{c}}(r) = \bigvee_{R(l)=r} \widetilde{\boldsymbol{C}}_{\boldsymbol{R}}(l). \tag{4.12}$$

In order to determine the fuzzy maximising set $\widetilde{\boldsymbol{C}}_{\boldsymbol{R}}$, the following theoremsare used:

Theorem 4.1

$$\widetilde{\boldsymbol{C}}_{\boldsymbol{R}} = \widetilde{\boldsymbol{C}} \cap \boldsymbol{M} = \widetilde{\boldsymbol{C}} \cap \bar{\boldsymbol{M}} \tag{4.13}$$

Proof:

From Definition 4.3,

$$\boldsymbol{M} \triangleq \bigcup_{0<\lambda\leq 1} \boldsymbol{M}_{\boldsymbol{\lambda}} = \text{supp}\widetilde{\boldsymbol{C}}_{\boldsymbol{R}}. \tag{4.14}$$

From equation (4.14),

$$\widetilde{\boldsymbol{C}}_{\boldsymbol{R}}(l) = 0,\ l \in \bar{\boldsymbol{M}} - \boldsymbol{M} \tag{4.15}$$

where

$$\bar{\boldsymbol{M}} - \boldsymbol{M} = (\boldsymbol{M}_{\boldsymbol{0}} \cup \boldsymbol{M}) - \boldsymbol{M} = \boldsymbol{M}_{\boldsymbol{0}} - \boldsymbol{M}. \tag{4.16}$$

When $l \in \bar{\boldsymbol{M}} - \boldsymbol{M}$, we have $l \in \boldsymbol{M}_{\boldsymbol{0}} - \boldsymbol{M}$, and $\widetilde{\boldsymbol{C}}(l) = 0$. In fact, when $\widetilde{\boldsymbol{C}}(l) = \lambda > 0$, we have $l \in \boldsymbol{C}_{\boldsymbol{\lambda}}$, and when $l \in \boldsymbol{M}_{\boldsymbol{0}}$, we have

$$\boldsymbol{R}(l) = \max_{l_1 \in \boldsymbol{C}_{\boldsymbol{0}}} \boldsymbol{R}(l_1) \geq \max_{l_1 \in \boldsymbol{C}_{\boldsymbol{\lambda}}} \boldsymbol{R}(l_1) \tag{4.17}$$

and

$$\boldsymbol{R}(l) = \max_{l_1 \in \boldsymbol{C}_{\boldsymbol{\lambda}}} \boldsymbol{R}(l_1). \tag{4.18}$$

Therefore, $l \in \boldsymbol{M}_{\boldsymbol{\lambda}} \subseteqq \boldsymbol{M}$. This is conflict with $l \in \boldsymbol{M}_{\boldsymbol{0}} - \boldsymbol{M}$. As a result, we have

$$\widetilde{\boldsymbol{C}}(l) = 0,\ l \in \bar{\boldsymbol{M}} - \boldsymbol{M}. \tag{4.19}$$

From equation (4.19), we have

$$\widetilde{\boldsymbol{C}} \cap \bar{\boldsymbol{M}} = \widetilde{\boldsymbol{C}} \cap \boldsymbol{M}. \tag{4.20}$$

From equation (4.14), if we want to proof theorem4.1, we need to proof the following equation:

$$\widetilde{\boldsymbol{C}}_{\boldsymbol{R}}(l) = \widetilde{\boldsymbol{C}}(l),\ \ l \in \boldsymbol{M}. \tag{4.21}$$

$\forall\, l_0 \in \boldsymbol{M}$, there is at least one $\lambda > 0$, that makes $l_0 \in \boldsymbol{M}_{\boldsymbol{\lambda}}$. When $\mu > \lambda$ and $l_0 \in \boldsymbol{C}_{\boldsymbol{\mu}}$, we have $\boldsymbol{M}_{\boldsymbol{\mu}} = \boldsymbol{C}_{\boldsymbol{\mu}} \cap \boldsymbol{M}_{\boldsymbol{\lambda}}$, therefore $l_0 \in \boldsymbol{M}_{\boldsymbol{\mu}}$, which means, when $\mu > \lambda$,

$$l_0 \in \boldsymbol{M}_{\boldsymbol{\mu}} \Leftrightarrow\ l_0 \in \boldsymbol{C}_{\boldsymbol{\mu}}. \tag{4.22}$$

Therefore,

$$\widetilde{\boldsymbol{C}}_{\boldsymbol{R}}(l_0) = \bigvee_{0\le\lambda\le1}\big(\lambda \wedge \boldsymbol{M}_{\boldsymbol{\lambda}}(l_0)\big) = \bigvee_{0\le\lambda\le1}(\lambda \wedge \boldsymbol{C}_{\boldsymbol{\lambda}}(l_0)) = \widetilde{\boldsymbol{C}}(l_0). \quad (4.23)$$

The maximising set $\bar{\boldsymbol{M}}$ can be determined using theorem4.2.

Theorem 4.2

Suppose $\boldsymbol{R}(l)$ and $\widetilde{\boldsymbol{C}}(l)$ are two 3D r-type functions defined on $U \triangleq FRS$. $\boldsymbol{R}(l)$ has the peak-field of $\boldsymbol{P}_{\boldsymbol{R}}$ and $\widetilde{\boldsymbol{C}}(l)$ has the peak-field of $\boldsymbol{P}_{\tilde{\boldsymbol{c}}}$,

$$\bar{\boldsymbol{M}} = \begin{cases} \boldsymbol{P}_{\boldsymbol{R}}, & \boldsymbol{P}_{\boldsymbol{R}} \cap \boldsymbol{P}_{\tilde{\boldsymbol{c}}} \neq \emptyset \\ \boldsymbol{L}, & \boldsymbol{P}_{\boldsymbol{R}} \cap \boldsymbol{P}_{\tilde{\boldsymbol{c}}} = \emptyset \end{cases} \quad (4.24)$$

where $\boldsymbol{L}$ is a set of robot base locations within the area between the outer edge of $\boldsymbol{P}_{\boldsymbol{R}}$and inner edge of $\boldsymbol{P}_{\tilde{\boldsymbol{c}}}$:

Proof:

It is known that $\boldsymbol{M}_{\boldsymbol{0}} = \boldsymbol{P}_{\boldsymbol{R}}$, $\boldsymbol{C}_{\boldsymbol{1}} = \boldsymbol{P}_{\tilde{\boldsymbol{c}}}$. When $\boldsymbol{P}_{\boldsymbol{R}} \cap \boldsymbol{P}_{\tilde{\boldsymbol{c}}} \neq \emptyset$,

$$\boldsymbol{M}_{\boldsymbol{0}} \cap \boldsymbol{C}_{\boldsymbol{1}} \neq \emptyset. \quad (4.26)$$

Therefore, $\forall\ 0 \le \lambda \le 1$, $\boldsymbol{M}_{\boldsymbol{0}} \cap \boldsymbol{C}_{\boldsymbol{\lambda}} \neq \emptyset$, and $\boldsymbol{M}_{\boldsymbol{\lambda}} = \boldsymbol{M}_{\boldsymbol{0}} \cap \boldsymbol{C}_{\boldsymbol{\lambda}} = \boldsymbol{P}_{\boldsymbol{R}}$, As a result,

$$\bar{\boldsymbol{M}} = \boldsymbol{P}_{\boldsymbol{R}}. \quad (4.27)$$

When $\boldsymbol{P}_{\boldsymbol{R}} \cap \boldsymbol{P}_{\tilde{\boldsymbol{c}}} = \emptyset$, we have $\boldsymbol{M}_{\boldsymbol{0}} \cap \boldsymbol{C}_{\boldsymbol{1}} = \emptyset$, and $\boldsymbol{M}_{\boldsymbol{1}} \cap \boldsymbol{M}_{\boldsymbol{0}} = \emptyset$, therefore,

$$\boldsymbol{M}_{\boldsymbol{1}} = \{l | l = \max_{l\in \boldsymbol{P}_{\tilde{\boldsymbol{c}}}} \boldsymbol{R}(l)\} \quad (4.28)$$

$\forall l \in \boldsymbol{L}$, we have $\boldsymbol{M}_{\widetilde{\boldsymbol{C}}(l)} = \{l\}$, therefore,

$$\bar{\boldsymbol{M}} = \boldsymbol{L} \quad (4.29)$$

Up to now, the optimal base location l^* can be determined by the following equation:

$$l^* = \max_{l\epsilon \bar{\boldsymbol{M}}} \widetilde{\boldsymbol{C}}(l) \quad (4.30)$$

where $\widetilde{\boldsymbol{C}}(l)$ is a fuzzy constraint function. The maximising set $\bar{\boldsymbol{M}}$ can be determined by theorem4.2.

The optimal base orientation can be determined equation (4.3).

4.2.3 Constrained symbolic grounding process

The constrained symbolic grounding process is shown in Fig. 4.9.

In step 7 of the constrained symbolic grounding process, spatial information of the obstacles in the environment are retrieved using services provided by the knowledge base component. The format of the spatial information is described in Section 4.2.1.

In step 8, fuzzy constraint functions are defined according to the spatial information of obstacles retrieved in step 7. The method for defining fuzzy constraint function is described in section 4.2.1.

In step 9, a novel fuzzy optimisation algorithm is applied for determining the optimal base pose under fuzzy constraints. The detailed fuzzy optimisation algorithm is presented in Section 4.2.2.

In step 18-21, a new optimal base pose is determined according to the updated environmental information and the robot's base pose.

The other steps in the constrained symbolic grounding process are the same as the unconstrained symbolic grounding process descried in Section 4.1.3.

process ConstrainedSymbolicGrounding

```
1.  tar_obj_loc = est_tar_obj_loc;
2.  weight_coeffs = predef_weight_coeffs;
3.  thres = predef_thres;
4.  current_rbp = getRBP ( );
5.  frs = getFRS (tar_obj_loc);
6.  obj_func = getOF (frs, current_rbp, weight_coeffs);
7.  obstacle_spatial_info = getOSI ( );
8.  fuzzy_constraint_funcs = getFCFs (obstacle_spatial_info);
9.  obp = getOBP (obj_func, fuzzy_constraint_funcs);
10. moveBase (obp);
11. tar_obj_loc = getTOL ( );
12. current_rbp = getRBP ( );
13. frs = getFRS (tar_obj_loc);
14. reach = getReach (frs, current_rbp);
15. if (reach > thres):
16.     ret current_rbp;
17. else:
18.     obj_func = getOF (frs, current_rbp, weight_coeffs);
19.     obstacle_spatial_info = getOSI ( );
20.     fuzzy_constraint_funcs = getFCFs (obstacle_spatial_info);
21.     obp = getOBP (obj_func, fuzzy_constraint_funcs);
22.     ret obp;
```

End

(**Meaning ofAbbreviations:** tar_obj_loc/TOL: target object location, est_tar_obj_loc: estimated target object location, predef: predefined, weight_coeffs: weight coefficients, thres: threshold, rbp/RBP: robot base pose, frs/FRS: fuzzy reachability space, obj_func/OF: objective function, OSI: obstacle spatial information, FCFs: fuzzy constraint functions, obp/OBP: optimal base pose, reach/Reach: reachability)

Figure 4.9 Pseudo-code of constrained symbolic grounding process

4.2.4 Simulations

The aim of the simulation is to test the constrained fuzzy optimisation algorithm. An object fetching simulation was carried out where a Care-O-bot 3 robot tried to fetch an object in ROS simulation environment. In the simulation, a table was placed at (-2.65*m*, 0.25*m*, 0°) in the world coordinate. The length of the table was 1.5*m* and the width was 1.0*m*. Thetarget object was placed on the table. The location of the object was (-2.2*m*, 0.2*m*). The estimated object location was (-2.2*m*, 0.25*m*). A cylinder-shape obstacle was placed at (-1.75*m*, 0.0*m*). The radius of the obstacle was 0.1*m*. The robot was placed at (0.0*m*, 0.0*m*). An FRS was established according to the estimated target object location. Two fuzzy constraints were defined according to the spatial information of the table and the obstacle. According to the FRS and the fuzzy constraints, a preliminary base pose was calculated using constrained fuzzy optimisation algorithm. The preliminary base pose was (-1.51*m*, 0.65*m*, 30.01°). After the robot reached the preliminary base pose, detection was carried out to extract the actual target object location. A reachability value of the robot's current pose and a threshold were calculated (in the simulation, *reachability* = 0.53, *threshold* = 0.47). The reachability value was above the threshold. The robot started to grasp the target object from its current base pose. The FRS,the maximizing set $\bar{\boldsymbol{M}}$,and the movement trajectory of the robot base are shown in Fig 4.10.

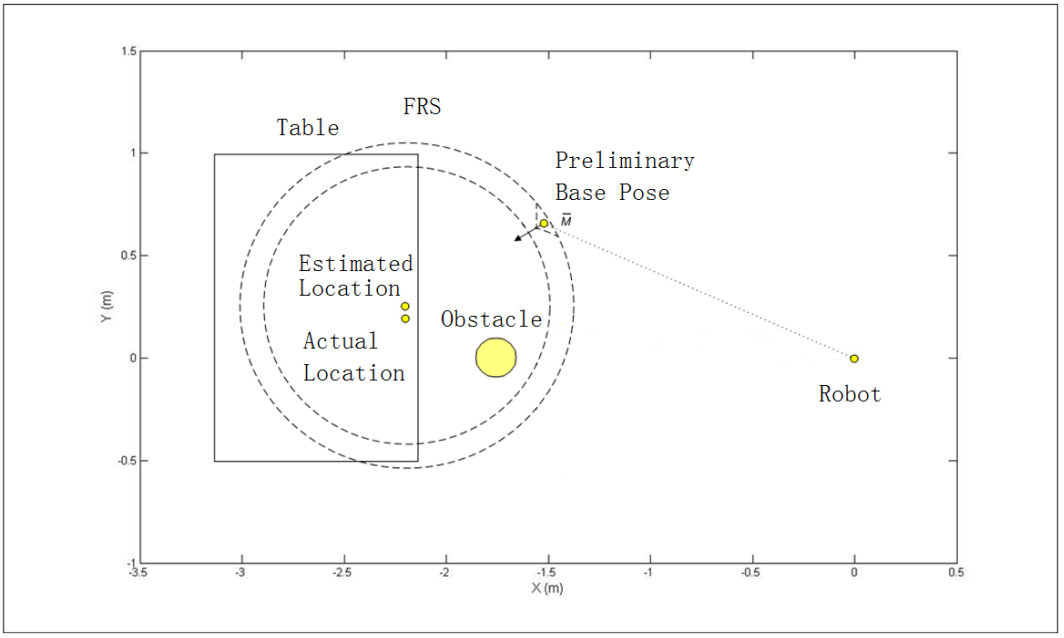


Figure 4.10 The FRSand the robot base movement trajectory to the preliminary base pose

From the simulation, the preliminary base pose calculated using constrained fuzzy optimisation algorithm has high reachability value and it is also far away from obstacles. The estimated target object location was close to the actual target object location. Therefore, the robot started to grasp the object from the preliminary base pose and the object was successfully grasped.

4.3 Summery

In this chapter, an unconstrained and a constrained symbolic grounding process are presented. An objective function is defined as a function of the reachability value of a desired robot base pose and the distance between a robot's current base pose and the desired robot base pose. Obstacles and their influence to the reachability value of robot base poses are modelled as fuzzy constraints. By applying novel fuzzy optimisation algorithms, the optimal base pose for grasping an object can be determined.

CHAPTER 5 SYMBOLIC GROUNDING WITH FUZZY OBJECTIVE FUNCTION

The previous chapters presented the algorithms for grounding symbolic task commands as specific robot base poses. In unstructured environments, it is very time-consuming for a robot to navigate to an exact pose, especially when this pose is close to an obstacle (Qiu et al., 2012). When a robot is navigating to a target pose, the robot will first calculate its current base pose based on sensory information from an odometry. After that, it will calculate a collision-free trajectory and a velocity command based on sensory information from laser scanners. When the robot is very close to the target pose, this process will repeat for several times to make sure the robot base is located at the exact pose. It is inspired by human reasoning that when human approach to an object they do not turn to calculate an exact position for themselves to pick up the object (Rosenbaum, 1980). To improve the efficiency of task implementation, symbolic task commands can be grounded as regions. When a robot is navigating to a target region, it only needs to decide whether its base is located within the region. This will save the time for exact placing the robot base. It is unnecessary to have exact robot base regions due to the use of human reasoning. Therefore, a fuzzy objective function is defined based on FRS and the fuzzy constraints of the optimal base pose calculated using the fuzzy optimisation algorithm discussed in Chapter 4. The fuzzy objective function models the uncertainty of whether the reachability value of a robot base pose is high enough for implementing a task. The optimal base region can be then determined using the fuzzy objective function and the optimisation algorithm both of which will be given in this chapter.

5.1 Fuzzy Objective Function

In this section, the concept of fuzzy objective function and the process of establishing a fuzzy objective function are presented. A fuzzy objective function

will be used to determine the optimal base region. The reachability function $r = \boldsymbol{R}(l)$ described in Chapter 4 and a fuzzy set obtained in the domain of reachability values of robot base locations are used to establish the fuzzy objective function. The membership function of the fuzzy set is defined according to the fuzzy constraints of the optimal base pose calculated using the constrained fuzzy optimisation algorithm discussed in Chapter 4. The fuzzy set indicates if a robot base pose is close to an obstacle, its reachability value will have lower membership degree. Therefore, it will have less opportunity to be determined as a part of the optimal base region.

5.1.1 Concept of fuzzy objective function

A fuzzy objective function is defined as a fuzzy set based on a traditional objective function along with a parameter α that defines the shape of the fuzzy set. A 3D fuzzy objective function is defined over a traditional objective function.. The reachability function $r = \boldsymbol{R}(l)$ is used as the traditional objective function because it can give the optimal robot base pose (refer to Chapter 4). The X-axis and Y-axis of the fuzzy objective function indicate the plane where a robot residents. The Z-axis of the fuzzy objective function indicates the membership degree of the reachability values to the fuzzy set.

Definition 5.1 (fuzzy maximum value)

A fuzzy maximum value of an objective function $\boldsymbol{R}(l)$ is defined by fuzzy set $\widetilde{M}$. Suppose $r = \boldsymbol{R}(l)$ is a 3D objective function defined on $\boldsymbol{U} \triangleq FRS$.

$$m = \min_{l \in \boldsymbol{U}} \boldsymbol{R}(l), \tag{5.1}$$

$$M = \max_{l \in \boldsymbol{U}} \boldsymbol{R}(l). \tag{5.2}$$

A fuzzy set $\widetilde{M}$ is defined on $[m, M]$. The membership function of the fuzzy set, $\widetilde{M}(r)$, is a monotoncally increasing function, and

$$\widetilde{M}(M) = 1 \tag{5.3}$$

Definition 5.2 (3D fuzzy objective function)

A 3D fuzzy objective function, $\widetilde{\boldsymbol{A}}$, is defined as a fuzzy set:

$$\widetilde{\boldsymbol{A}}(l) = \widetilde{M}(\boldsymbol{R}(l)) \tag{5.4}$$

In this research, an objective function is defined based a 3D reachability function, such as:

$$r = \boldsymbol{R}(l),\ l \in \boldsymbol{U},\ r \in [0,1] \tag{5.5}$$

where r is the reachability value of a robot base location l. Let

$$m = 0,\ M = 1. \tag{5.6}$$

According to Definition 5.1, a fuzzy set, $\widetilde{M}(r)$ can be defined as:

$$\widetilde{M}(r) = r^{\alpha},\ \alpha > 0. \tag{5.7}$$

r^{α} is a monotoncally increasing function and $\widetilde{M}(M) = 1^{\alpha} = 1$.

This can be illustrated in Fig. 5.1.

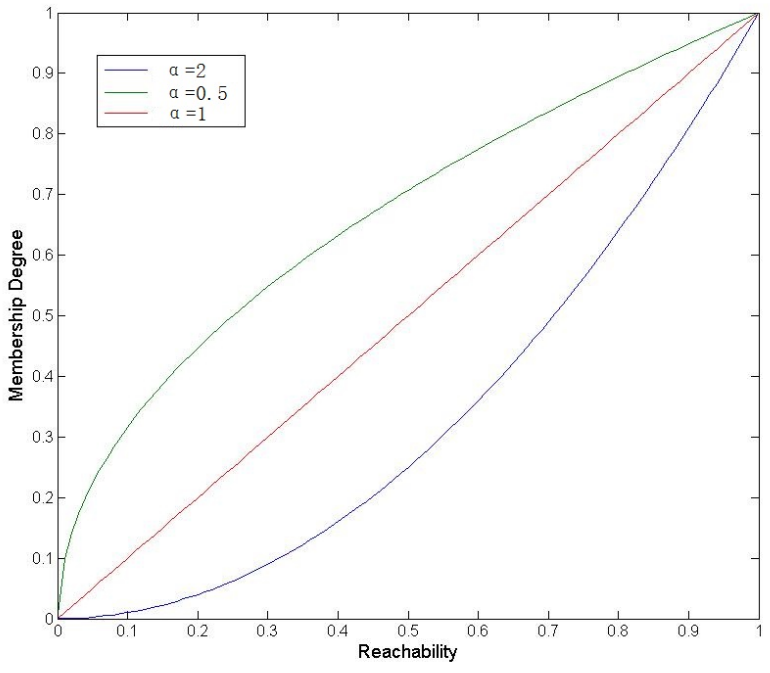


Figure 5.1 An example of the fuzzy set $\widetilde{M}$

In this diagram, α is set to three different values ($\alpha = 1$, $\alpha = 2$, and $\alpha = 0.5$). The red curve shows the membership function of the fuzzy set $\widetilde{M}$ when α is set to 1. The blue curve shows the membership function of the fuzzy set $\widetilde{M}$ when α is set to 2. The green curve shows the membership function of the fuzzy set $\widetilde{M}$ when α is set to 0.5. The horizontal axis represents the reachability value of robot base locations. The vertical axis represents the membership degree of the reachability values to the fuzzy set $\widetilde{M}$. The fuzzy set $\widetilde{M}$ models the uncertainty of whether the reachability value of a robot base location is high enough for implementing a task. A reachability value can have different membership degree to the fuzzy set when the value of α changes.

According to Definition 5.2, a 3D fuzzy objective function can be established as:

$$\widetilde{\boldsymbol{A}}(l) = \widetilde{M}(\boldsymbol{R}(l))=(\boldsymbol{R}(l))^{\alpha} \tag{5.8}$$

When $\alpha = 1$, the 3D fuzzy objective function becomes $\widetilde{\boldsymbol{A}}(l) = \boldsymbol{R}(l)$. The 3D fuzzy objective function is illustrated in Fig. 5.2 and the cross-area of the 3D fuzzy objective function at the X-Z plane is illustrated in Fig. 5.3.

In Fig. 5.2, the horizontal axes represent the robot base locations in $\boldsymbol{U}$. The vertical axis represents the membership degree of the robot base locations to the fuzzy set $\widetilde{\boldsymbol{A}}$. The membership degree indicates whether a robot base location can be determined as a part of the optimal base region.

In Fig. 5.3, The cross-area shows the shape and size of the 3D fuzzy objective function. When the value of α changes the fuzzy objective function will change accordingly.

When $\alpha = 2$, the 3D fuzzy objective function becomes $\widetilde{\boldsymbol{A}}(l) = (\boldsymbol{R}(l))^2$. The 3D fuzzy objective function is illustrated in Fig. 5.4. The cross-area of the 3D fuzzy objective function at the X-Z plane is illustrated in Fig. 5.5.

In Fig. 5.4, the horizontal axes represent the robot base locations within $\boldsymbol{U}$. The

vertical axis represents the membership degree of the robot base locations to the fuzzy set $\widetilde{\boldsymbol{A}}$. The membership degree of some robot base locations decreases when the value of α increases. This indicates a smaller optimal base region will be determined using this fuzzy objective function.

In Fig. 5.5, The cross-area shows the size and shape of the 3D fuzzy objective function. The cross-area becomes smaller when the value of α increases.

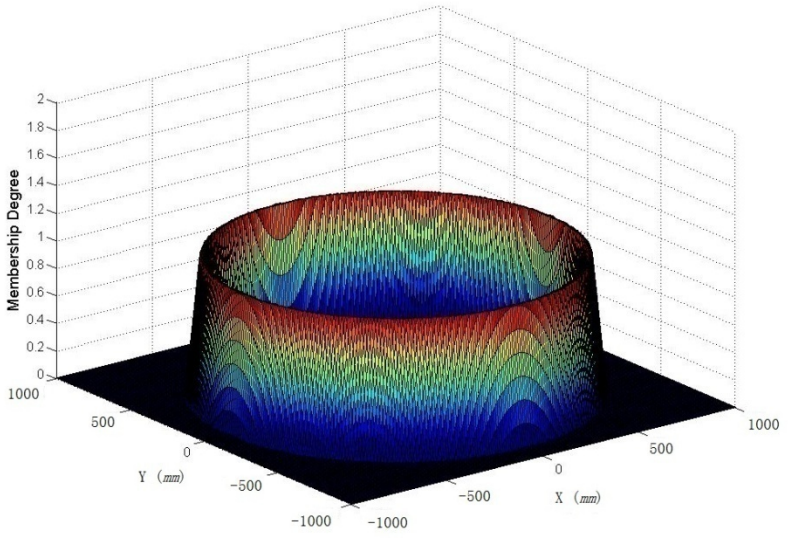


Figure 5.2 3D Fuzzy objective function ($\alpha = 1$)

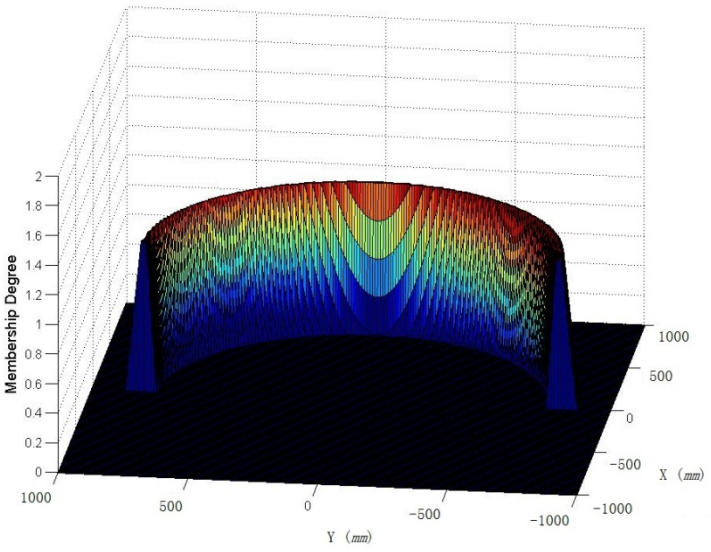


Figure 5.3 Cross-area of the 3D fuzzy objective function at the X-Z plane ($\alpha = 1$)

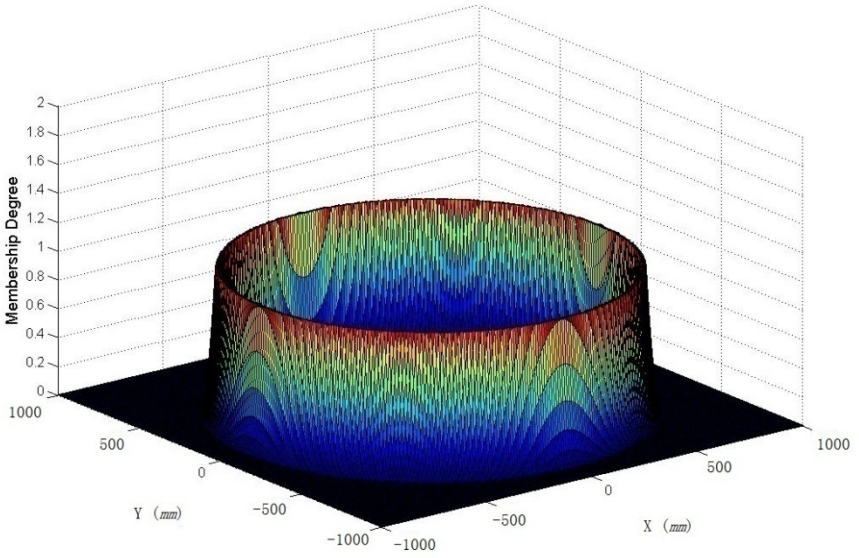


Figure 5.4 3D Fuzzy objective function ($\alpha = 2$)

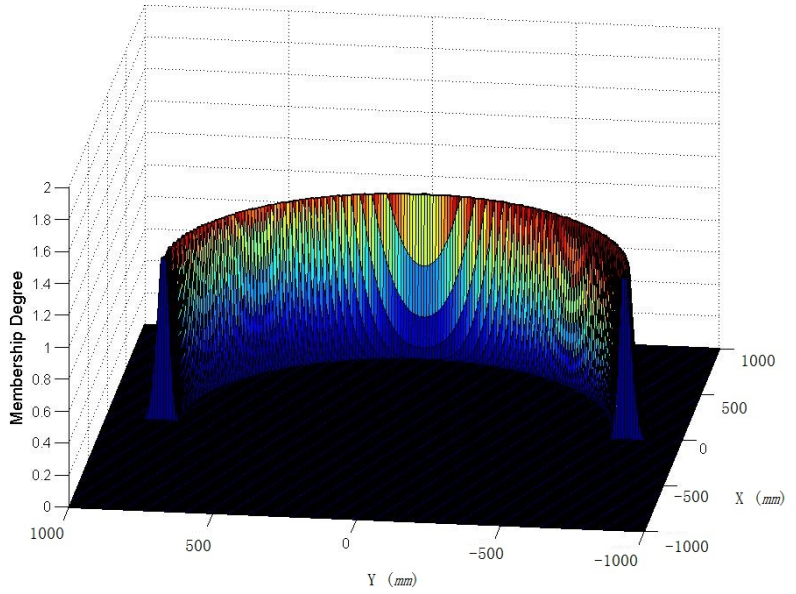


Figure 5.5 Cross-area of the 3D fuzzy objective function at the X-Z plane ($\alpha = 2$)

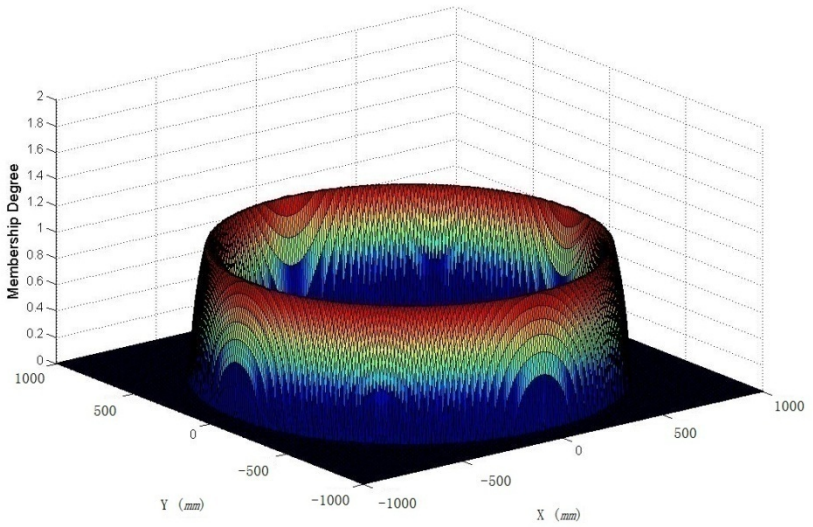


Figure 5.6 Fuzzy objective function ($\alpha = 0.5$)

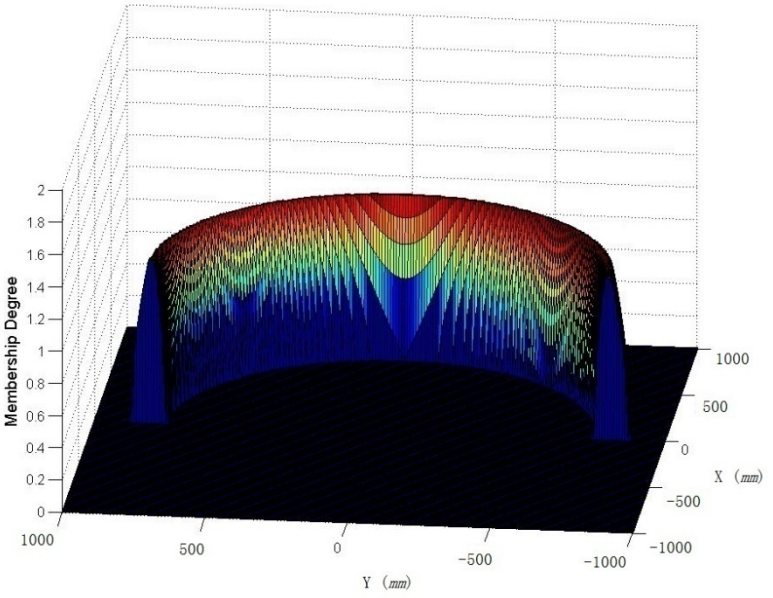


Figure 5.7 Cross-area of the 3D fuzzy objective function at the X-Z plane ($\alpha = 0.5$)

When $\alpha = 0.5$, the 3D fuzzy objective function becomes $\widetilde{\boldsymbol{A}}(l) = \sqrt{\boldsymbol{R}(l)}$. The 3D fuzzy objective function is illustrated in Fig. 5.6. The cross-area of the 3D fuzzy objective function at the X-Z plane is illustrated in Fig. 5.7.

In Fig. 5.6, the horizontal axes represent the robot base locations within $\boldsymbol{U}$. The vertical axis represents the membership degree of the robot base locations to the fuzzy set $\widetilde{\boldsymbol{A}}$. The membership degree of some robot base locations increases when the value of α decreases. This indicates a bigger optimal base region will be determined using this fuzzy objective function.

In Fig. 5.7, The cross-area shows the size and shape of the 3D fuzzy objective function. The cross-area becomes bigger when the value of α decreases.

5.1.2 Establishment of fuzzy objective function

To establish a fuzzy objective function, the following criterion is used:

- The robot base locations with high fuzzy constraint value and high reachability value should havehigh membership degree to the fuzzy set $\widetilde{\boldsymbol{A}}$.

The robot base locations with high fuzzy constraint value are far away from obstacles. Therefore, these locations should have high membership degree to the fuzzy set $\widetilde{\boldsymbol{A}}$**,** which meanstheselocationsare more likely to be determined as a part of the optimal base region. On the other hand, when the robot base navigation error and the object location extraction error are considered, a robot could still find a valid arm configuration when the robot is placed at the locations with high reachability value. Therefore, robot base locations with high reachability value should have high membership degree to the fuzzy set $\widetilde{\boldsymbol{A}}$**,** which means these locationsare more likely to be determined as a part of the optimal base region.

According to Definition 5.2, when the value of α is determined, a fuzzy objective function can be established. Define L as a set of robot base locations:

$$L = \{l | d(l, E_{\boldsymbol{R}}) = e_{nav} + e_{obj}\} \tag{5.9}$$

where $d(l, E_{\boldsymbol{R}})$ is the distance between a robot base location l and the bottom edge $E_{\boldsymbol{R}}$ of the 3D reachability function, e_{nav} is the maximum robot base navigation error, e_{obj} is the maximum object location extraction error. The value of α can be calculated using:

$$(\boldsymbol{R}(L))^{\alpha} = c \tag{5.9}$$

$$\alpha = \log_{\boldsymbol{R}(L)} c \tag{5.10}$$

where c is the constraint value of the optimal base pose calculated using the constrained fuzzy optimisation algorithm presented in Chapter 4.

The cross-areas of two 3D fuzzy objective functionsat the X-Z plane are illustrated in Fig. 5.8.

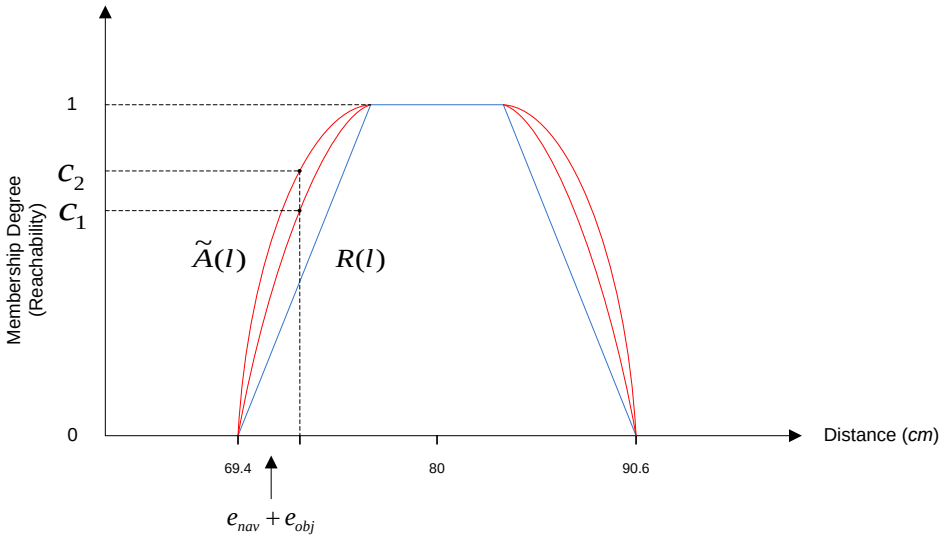


Figure 5.8 Cross-areas of two 3D fuzzy objective functionsat the X-Z plane

In this diagram, two fuzzy objective functions are established according to two constraint values of the optimal base pose. c_1 and c_2 are two constraint values. The curves in red colour show the size of the fuzzy objective functions. The trapezoid in blue colour shows the cross-area of the reachability function at the X-Z plane. When the constraint value of the optimal base pose is high, the optimal base pose is far from obstacles. Therefore, there will be a bigger optimal base region for a robot to implement a task. From the diagram, when the optimal base pose has a higher constraint value, robot base locations with the same reachability value will have a higher membership degree to the fuzzy set $\widetilde{\boldsymbol{A}}$. Therefore, they are more likely to be determined as a part of the optimal base region. According to Definition 4.1, the membership function of the fuzzy set defined on the domain of reachability value is a monotoncally increasing function, which means robot base poses with higher reachability value have higher membership degree to the fuzzy set $\widetilde{\boldsymbol{A}}$.

5.2 Fuzzy Optimisation with Fuzzy Objective Function and Fuzzy Constraints for Symbolic Grounding

A fuzzy optimisation algorithm was developed for determining the optimal base region. The optimal base region should meet the following criteria:

- When the robot navigation error and the object location extraction error are taken into account, a robot could still find a valid arm configuration when the robot is place at a location within the optimal base region.
- The constraint value of the robot base locations within the optimal base region is not lower than the constraint value of the optimal base pose calculated using the constrained fuzzy optimisation algorithm (refer to Section 4.2 of Chapter 4). This is to make sure that the optimal base region is far away from obstacles.

The optimal base region, defined as R^*, can be determined using the following theorem:

Theorem 5.1

$$R^* = \begin{cases} \boldsymbol{A}_{c_1} \cap \boldsymbol{C}_c, & c \geq \boldsymbol{R}(L) \\ \boldsymbol{A}_{\boldsymbol{R}(L)} \cap \boldsymbol{C}_c, & c < \boldsymbol{R}(L) \end{cases} \tag{5.11}$$

where $\boldsymbol{R}(l)$ is the reachability function described in Chapter 4, c is the constraint value of the optimal base pose calculated using the constrained fuzzy optimisation algorithm, $\boldsymbol{C}_c$ is a fuzzy cut set of the fuzzy constraint set $\widetilde{\boldsymbol{C}}$,

$$\boldsymbol{C}_c = \{l | \widetilde{\boldsymbol{C}}(l) \geq c\} \tag{5.12}$$

(refer to Chapter 4), $\boldsymbol{A}_{\boldsymbol{R}(L)}$ is a fuzzy cut set of the fuzzy set $\widetilde{\boldsymbol{A}}$,

$$\boldsymbol{A}_{\boldsymbol{R}(L)} = \{l | \widetilde{\boldsymbol{A}}(l) > \boldsymbol{R}(L)\}, \tag{5.13}$$

$\boldsymbol{A}_{c_1}$ is a proper subset of a fuzzy cut set $\boldsymbol{A}_c$, $\boldsymbol{A}_{c_1} \subsetneq \boldsymbol{A}_c$, and

$$\boldsymbol{A}_{c_1} = \{l | \widetilde{\boldsymbol{A}}(l) > c\}, \tag{5.14}$$

L is a set of robot base locations defined by equation (5.9).

Proof:

When c $\geq \boldsymbol{R}(L)$,

$$(\boldsymbol{R}(L))^{\alpha} \geq \boldsymbol{R}(L). \tag{5.15}$$

Because $\boldsymbol{R}(L) \in [0, 1]$, $\alpha \leq 1$. From equation (5.14),

$$\boldsymbol{A}_{c_1} = \{l | (\boldsymbol{R}(l))^{\alpha} > c\}, \tag{5.16}$$

$$\boldsymbol{A}_{c_1} = \{l|(\boldsymbol{R}(l))^{\alpha} > (\boldsymbol{R}(L))^{\alpha}\}, \tag{5.17}$$

$$\boldsymbol{A}_{c_1} = \{l|\boldsymbol{R}(l) > \boldsymbol{R}(L)\}. \tag{5.18}$$

When $\boldsymbol{R}(l) > \boldsymbol{R}(L)$,

$$L_1 = \{l_1|d(l_1,\ l) < e_{nav} + e_{obj}\} \subsetneq U, \tag{5.19}$$

where $d(l_1,\ l)$ is the distance between l_1 and l. Then

$$\boldsymbol{R}(L_1) > 0. \tag{5.20}$$

According to the definition of reachability function, the optimal base region R^*meets the first criteria.

Because $R^* = \boldsymbol{A}_{c_1} \cap \boldsymbol{C}_c$, $R^* \subset \boldsymbol{C}_c$, then

$$\widetilde{\boldsymbol{C}}(R^*) \geq c \tag{5.21}$$

Therefore, the optimal base region R^*alsomeets the second criteria.

When c $< \boldsymbol{R}(L)$,

$$(\boldsymbol{R}(L))^{\alpha} < \boldsymbol{R}(L). \tag{5.22}$$

Because $\boldsymbol{R}(L) \in [0, 1]$, $\alpha > 1$. From equation (5.13),

$$\boldsymbol{A}_{\boldsymbol{R}(L)} = \{l|(\boldsymbol{R}(l))^{\alpha} \geq \boldsymbol{R}(L)\}. \tag{5.23}$$

Because c $< \boldsymbol{R}(L)$,

$$\boldsymbol{A}_{\boldsymbol{R}(L)} \subset \{l|(\boldsymbol{R}(l))^{\alpha} > c\}, \tag{5.24}$$

$$\boldsymbol{A}_{\boldsymbol{R}(L)} \subset \{l|(\boldsymbol{R}(l))^{\alpha} > (\boldsymbol{R}(L))^{\alpha}\}, \tag{5.25}$$

$$\boldsymbol{A}_{\boldsymbol{R}(L)} \subset \{l|\boldsymbol{R}(l) > \boldsymbol{R}(L)\}. \tag{5.26}$$

According to equation (5.19) and equation (5.20), the optimal base region R^*meets the first criteria.

Because $R^* = \boldsymbol{A}_{\boldsymbol{R}(L)} \cap \boldsymbol{C}_c$, $R^* \subset \boldsymbol{C}_c$, then

$$\widetilde{\boldsymbol{C}}(R^*) \geq c \tag{5.27}$$

Therefore, the optimal base region R^*alsomeets the second criteria.

The cross-areas of two fuzzy objective functions ($\widetilde{\boldsymbol{A_1}}$ and $\widetilde{\boldsymbol{A_2}}$) and the reachability function ($\boldsymbol{R}(l)$) at the X-Y plane are illustrated in Fig. 5.9.

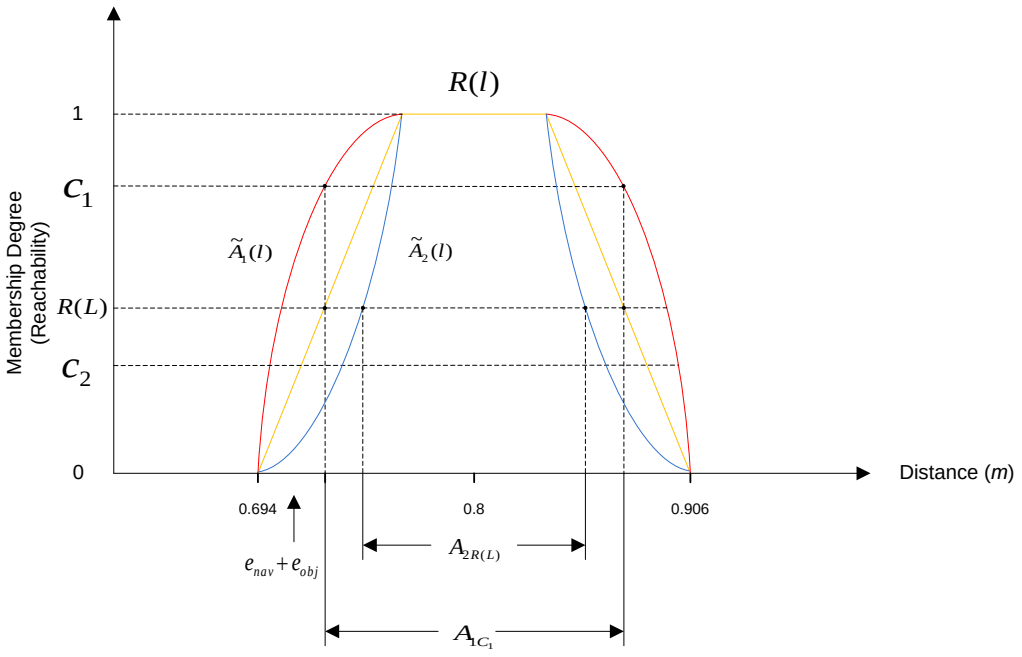


Figure 5.9 The cross-areas of two fuzzy objective functions and the reachability function at the X-Y plane

In this diagram, the curve in blue colour shows the shape of the fuzzy objective function $\widetilde{\boldsymbol{A_1}}$. The curve in red colour shows the shape of the fuzzy objective function $\widetilde{\boldsymbol{A_2}}$. The trapezoid in yellow colour is the cross-area of the reachability function $\boldsymbol{R}(l)$ at the X-Y plane. $\widetilde{\boldsymbol{A_1}}$ is established according to a constraint value c_1. $\widetilde{\boldsymbol{A_2}}$ is established according to a constraint value c_2. $c_1 > R(L)$, $\alpha < 1$, a fuzzy cut set $\boldsymbol{A}_{1c_1}$ of $\widetilde{\boldsymbol{A_1}}$ is calculated using the threshold value of c_1. $c_2 < R(L)$, $\alpha > 1$, a fuzzy cut set $\boldsymbol{A}_{2R(L)}$ of $\widetilde{\boldsymbol{A_2}}$ is calculated using the threshold value of $R(L)$. The size of $\boldsymbol{A}_{1c_1}$ is bigger than the size of $\boldsymbol{A}_{2R(L)}$, which indicates the optimal base region will become smaller when it is close to an obstacle.

The process of fuzzy optimisation over a fuzzy objective function is shown in Fig. 5.10.

In the first step of the fuzzy optimisation process, the target object location is

extracted. In the second step, a reachability function is established according to the target object location. In the third step, spatial information of obstacles in the environment are retrieved from a knowledge base component (refer to Chapter 4). In the forth step, fuzzy constraints are established based on the spatial information of obstacles. In the fifth step, the optimal base pose and its constraint value are calculated using the constrained fuzzy optimisation algorithm described in Chapter 4. In the sixth step, a fuzzy objective function is established using the method described in Section 5.1.2. If the constraint value (c) of the optimal base pose is equal or higher than the reachability value ($R(L)$) of the robot base location set L defined by equation (5.9), in the seventh step, the constraint value c is used as a threshold to calculate a fuzzy cut set ($\boldsymbol{A}_{c_1}$) of the fuzzy objective function. Otherwise, the reachability value $R(L)$ is used as a threshold to calculate a fuzzy cut set ($\boldsymbol{A}_{R(L)}$) of the fuzzy objective function. In the eighth step, a fuzzy cut set $\boldsymbol{C}_c$ of the fuzzy constraint set is calculated using the threshold value of c. In the ninth step, the intersection set of the fuzzy cut set calculated in the seventh step ($\boldsymbol{A}_{c_1}$ or $\boldsymbol{A}_{R(L)}$) and the fuzzy cut set calculated in the eighth step ($\boldsymbol{C}_c$) is calculated. The intersection set is the optimal base region according to Theorem (5.1).

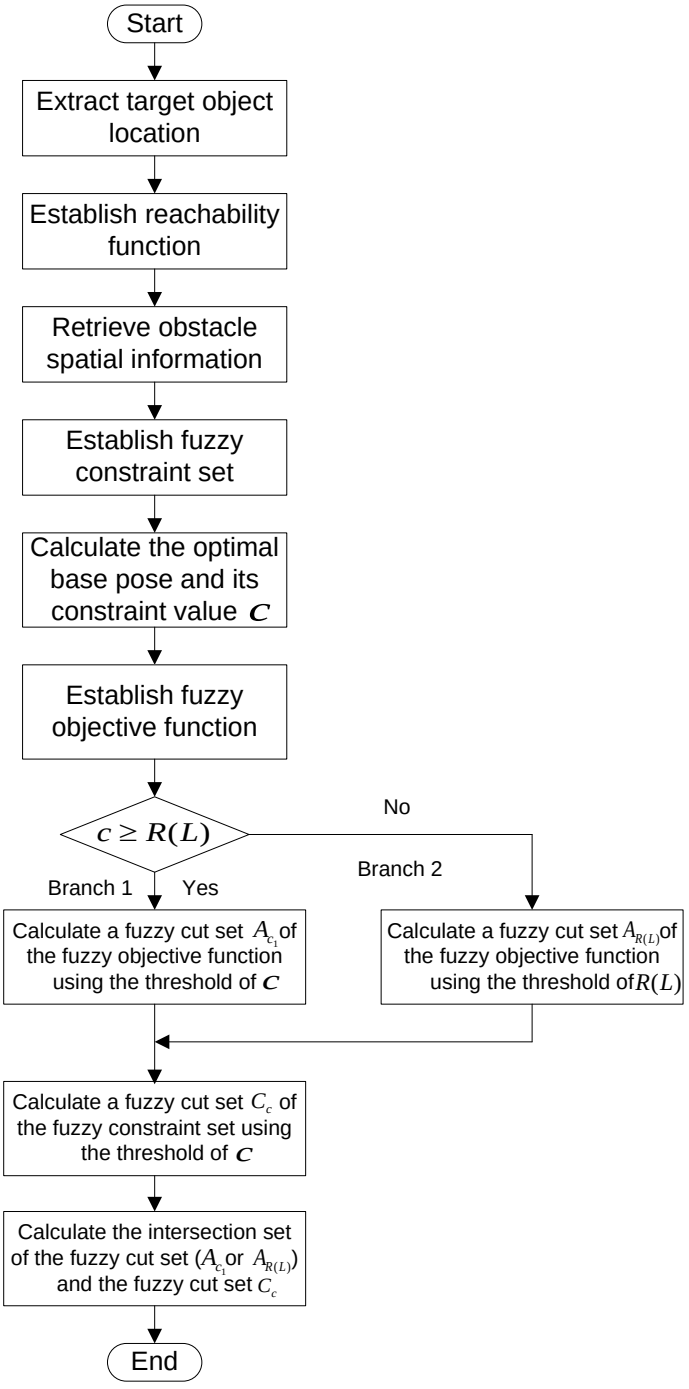


Figure 5.10 Fuzzy optimisation process

For Branch 1 of the fuzzy optimisation process, the constraint value c is used to calculate a fuzzy cut set $\boldsymbol{A}_{c_1}$ of the fuzzy objective function. The optimal base region is determined based on $\boldsymbol{A}_{c_1}$. For Branch 2 of the fuzzy optimisation

process, the reachability value $R(L)$ is used to calculate a fuzzy cut set $\boldsymbol{A}_{R(L)}$ of the fuzzy objective function. The optimal base region is determined based on $\boldsymbol{A}_{R(L)}$. The two optimal base regions are illustrated in Fig. 5.10 and Fig 5.11. In the case where the constraint value of the optimal base pose is higher than the reachability value of the robot base location set L ($c \geq R(L)$),the constraint value c is used as a threshold to calculate a fuzzy cut set, $\boldsymbol{A}_{c_1}$, of the fuzzy objective function $\widetilde{\boldsymbol{A}}(l)$. The intersection of the fuzzy cut set $\boldsymbol{A}_{c_1}$ and a fuzzy cut set, $\boldsymbol{C}_c$, of the fuzzy constraint set $\widetilde{\boldsymbol{C}}(l)$ is the optimal base region.

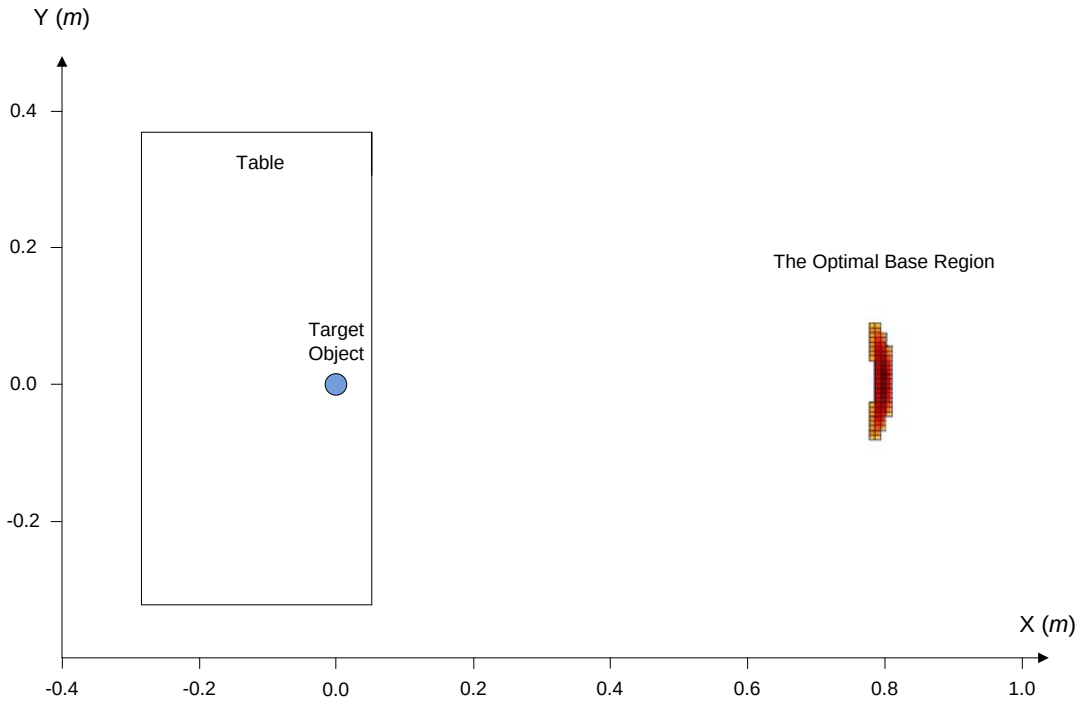


Figure 5.11 The optimal base region ($c \geq R(L)$)

In Fig. 5.11, the X-axis and Y-axis show robot base locations. A table is placed at (-0.15m, 0.0m, 0°). The length of the table is 0.7m and the width is 0.35m. An object is placed at (0.0m, 0.0m). The table is considered as an obstacle. The area in red colour is the optimal base region.

In the case where the constraint value of the optimal base pose c is lower than the reachability value of the robot base location set L ($c < R(L)$),the reachabilty value $\boldsymbol{R}(L)$ is used as a threshold to calculate a fuzzy cut set, $\boldsymbol{A}_{\boldsymbol{R}(L)}$, of the fuzzy objective function $\widetilde{\boldsymbol{A}}(l)$. The intersection of the fuzzy cut set $\boldsymbol{A}_{\boldsymbol{R}(L)}$ and a fuzzy cut set, $\boldsymbol{C}_c$, of the fuzzy constraint set $\widetilde{\boldsymbol{C}}(l)$ is the optimal base region.

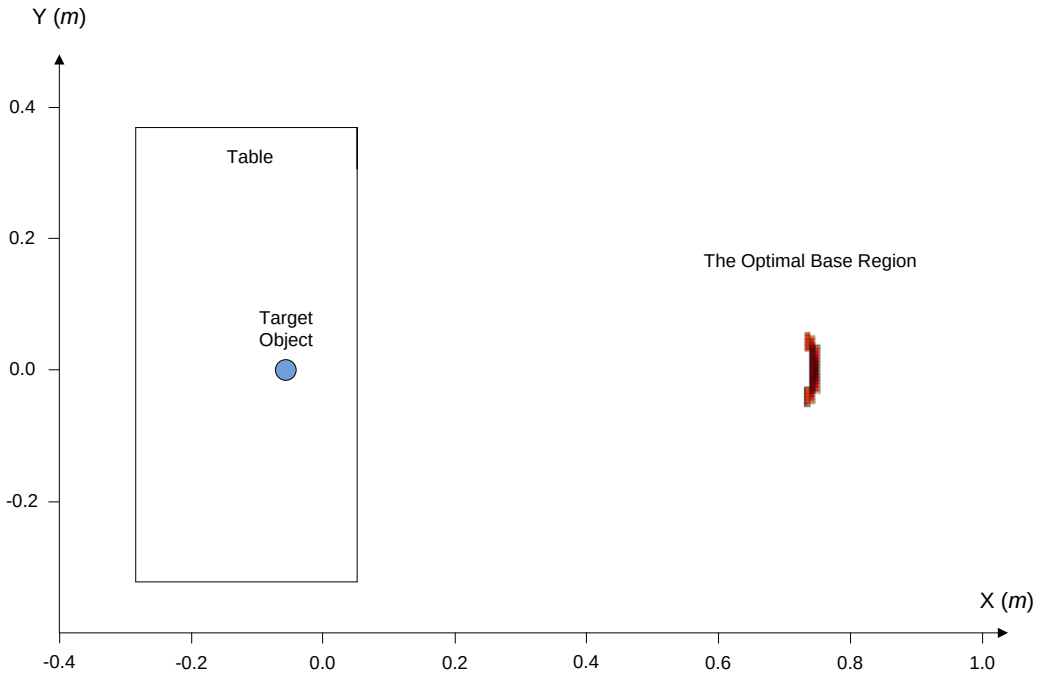


Figure 5.12 The optimal base region ($c < R(L)$)

In Fig. 5.12, the X-axis and Y-axis show robot base locations. A table is placed at (-0.15m, 0.0m, 0°). The length of the table is 0.7m and the width is 0.35m. An object is placed at (-0.05m, 0.0m). The table is considered as an obstacle. The area in red colour is the optimal base region. From this diagram, the optimal base region is closer to the edge of the table (obstacle). The optimal base region becomes smaller since it is more difficult for a robot to find a valid arm configuration when the robot is placed near to an obstacle.

5.3 Simulations and Discussion

The aim of the simulation is to test the fuzzy optimisation algorithm. Two object fetching simulations were carried out where a Care-O-bot 3 robot tried to fetch an object in ROS simulation environment. In the first simulation, a table was placed at (-2.65*m*, 0.25*m*, 0°) in the world coordinate. The length of the table was 1.5*m* and the width was 1.0*m*. Thetarget object was placed on the table. The location of the object was (-2.2*m*, 0.2*m*). A cylinder-shape obstacle was placed at (-1.75*m*, 0.0*m*). The radius of the obstacle was 0.1*m*. The robot was placed at (0.0*m*, 0.0*m*). Two fuzzy constraints were defined according to the spatial information of the table and the obstacle. An FRS was established according to an estimated target object location, which was (-2.2*m*, 0.25*m*). A preliminary base pose and its constraint value were calculated using the constrained fuzzy optimisation algorithm described in Chapter 4. In this simulation, the constraint value was 1. Based on the FRS and the fuzzy constraint value of the preliminary base pose, a fuzzy objective function was established. A preliminary base region was determined using the fuzzy optimisation algorithm. After the robot reached the preliminary base region, detection was carried out to extract the actual target object location. A reachability value of the robot's current pose and a threshold were calculated (in the simulation, *reachability* = 0.51, *threshold* = 0.47). The reachability value was above the threshold. The robot started to grasp the target object from its current base pose. The preliminary base region(the area in red colour)and the movement trajectory of the robot base are shown in Fig 5.13.

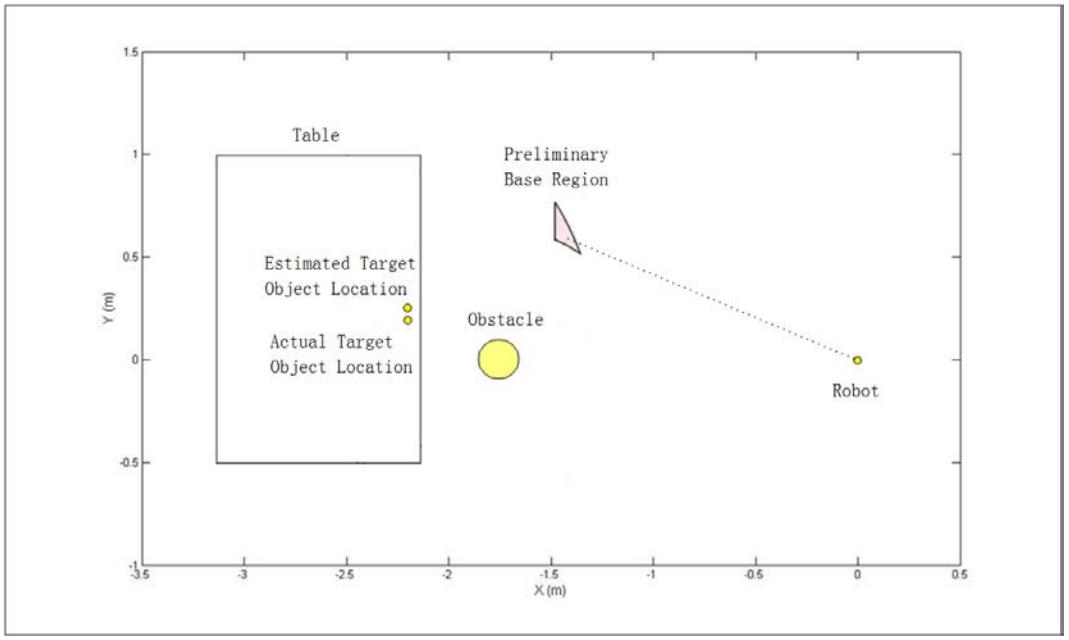


Figure 5.13 The preliminary base region and the robot base movement trajectory (constraint value = 1)

In this simulation, the constraint value of the robot base locations within the preliminary base region was 1. The preliminary base region was far away from the obstacles. The robot started to grasp the object when its base location was within the preliminary base region. The object was successfully grasped.

In the second simulation, a table was placed at (-2.65*m*, 0.25*m*, 0°) in the world coordinate. The length of the table was 1.5*m* and the width was 1.0*m*. Thetarget object was placed on the table. The location of the object was (-2.25*m*, 0.2*m*). A cylinder-shape obstacle was placed at (-1.75*m*, 0.05*m*). The radius of the obstacle was 0.1*m*. The robot was placed at (0.0*m*, 0.0*m*). Two fuzzy constraints were defined according to the spatial information of the table and the obstacle. An FRS was established according to an estimated target object location, which was (-2.25*m*, 0.25*m*). A preliminary base pose and its constraint value were calculated using the constrained fuzzy optimisation algorithm described in Chapter 4. In this simulation, the constraint value was 0.87. Based on the FRS and the fuzzy constraint value of the preliminary base pose, a fuzzy objective function was

established. A preliminary base region was determined using the fuzzy optimisation algorithm. After the robot reached the preliminary base region, detection was carried out to extract the actual target object location. A reachability value of the robot's current pose and a threshold were calculated (in the simulation, *reachability* = 0.49, *threshold* = 0.47). The reachability value was above the threshold. The robot started to grasp the target object from its current base pose. The preliminary base region(the area in red colour)and the movement trajectory of the robot base are shown in Fig 5.14.

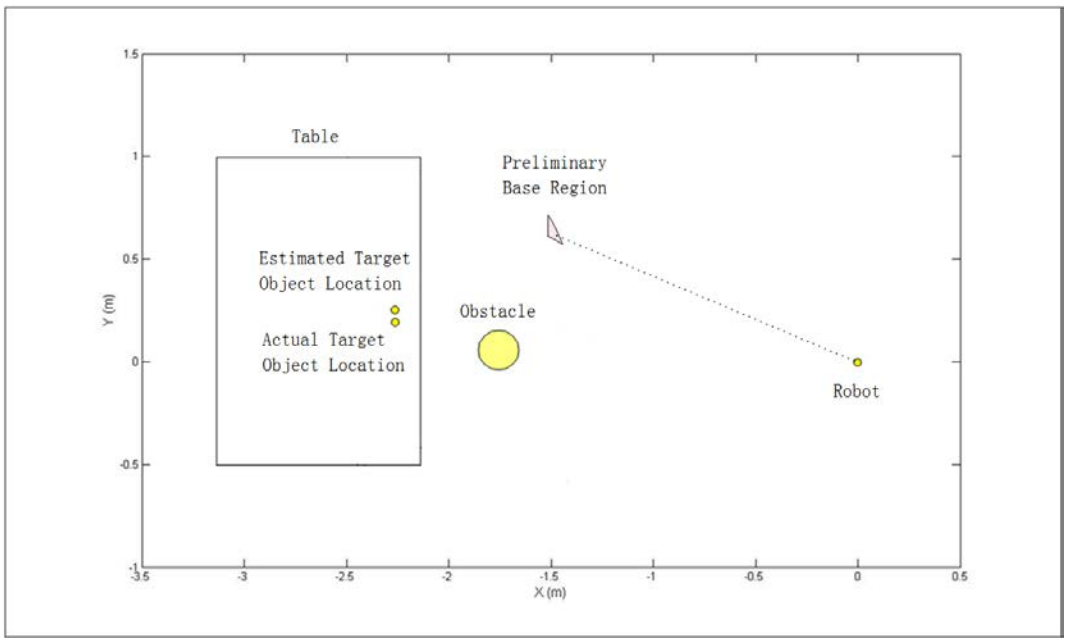


Figure 5.14 The preliminary base region and the robot base movement trajectory (constraint value = 0.87)

In this simulation, the constraint value of the robot base locations within the preliminary base region was 0.87. The preliminary base region was close the obstacles. Therefore, the preliminary base region was smaller than the first simulation. The robot started to grasp the object when its base location was within the preliminary base region. The object was successfully grasped.

5.4 Summery

In this chapter, a fuzzy objective function is defined based on FRS and fuzzy constraints. The fuzzy objective function models the uncertainty of whether the reachability value of a robot base location is high enough for implementing a task. A fuzzy optimisation algorithm is developed for determining the optimal base region over the fuzzy objective function. By grounding symbolic task commands to optimal base regions, a robot does not need to position itself to a specific pose for implementing the task. This can improve the efficiency of task implementation.

CHAPTER 6 INTEGRATING SYMBOLIC GROUNDING SERVICES WITH SHADOW ROBOTIC SYSTEM AND EXPERIMENTS

In this chapter, the fuzzy optimisation based symbolic grounding algorithm presented in Chapter 5 will be realised as Robotic Operating System (ROS) service blocks and integrated into Shadow Robotic System (SRS) autonomous control framework. The autonomous control framework and the symbolic grounding service blocks enable a Care-O-bot 3 robot to carry out fetch and carry tasks in unstructured home environments. In order to test the integrated system, experiments were designed and carried out in two different domestic environments. The analysis of results is also provided in this chapter.

6.1 Integrating Symbolic Grounding Services with Shadow Robotic System

Shadow Robotic System (SRS) is an EU FP7 ICT research project. The aim of the project is to develop robust personal assistive robots using Robot Operating System (ROS) and Care-O-bot 3 as the initial demonstration platform. The core of SRS is an autonomous control framework. In this section, the architecture of the autonomous control framework and how the symbolic grounding service blocks are integrated with the framework are presented.

6.1.1 Autonomous control framework

The autonomous control framework can be divided into two parts (Qiu et al., 2012 and Ji et al., 2012). First, it has an automatic task planner, which initialises actions on the symbolic level. The planner produces proactive robotic behaviours based

on updated semantic knowledge. Second, it has an action executive for coordination actions at the level of sensing and actuation. The two parts are integrated by the symbolic grounding service blocks. The control flow of the framework can be summarised as follows:

1) The task planner first evaluates the application domain, and derives a generic world model for the domain.
2) Based on partially perceived information from the environment and pre-knowledge from a semantic Knowledge Base (KB), an action sequence can be derived on the symbolic level to transfer the current state to the goal state.
3) In this step, three processes run in parallel:
    a) A task planner monitors the feedback from task coordination. It compares the actual feedback with the expected feedback from the generic world model. If unexpected behaviour is identified, the coordination is terminated and the control goes back to step 2.
    b) Symbolic terms in the action sequence derived in step 2 will be grounded into the optimal robot configurations by some mental actions. The mental actions are implemented by the symbolic grounding service blocks. To take most advantage of the updated semantic information, symbolic grounding is carried out in every step of the action sequence.
    c) A pre-developed reactive task coordination schema is loaded based on the action sequence and the grounded optimal robot configuration. The environment is treated as structured at this level. The coordination is ready to be interrupted at any time.

The architecture of the autonomous control framework is shown in Fig. 6.1.

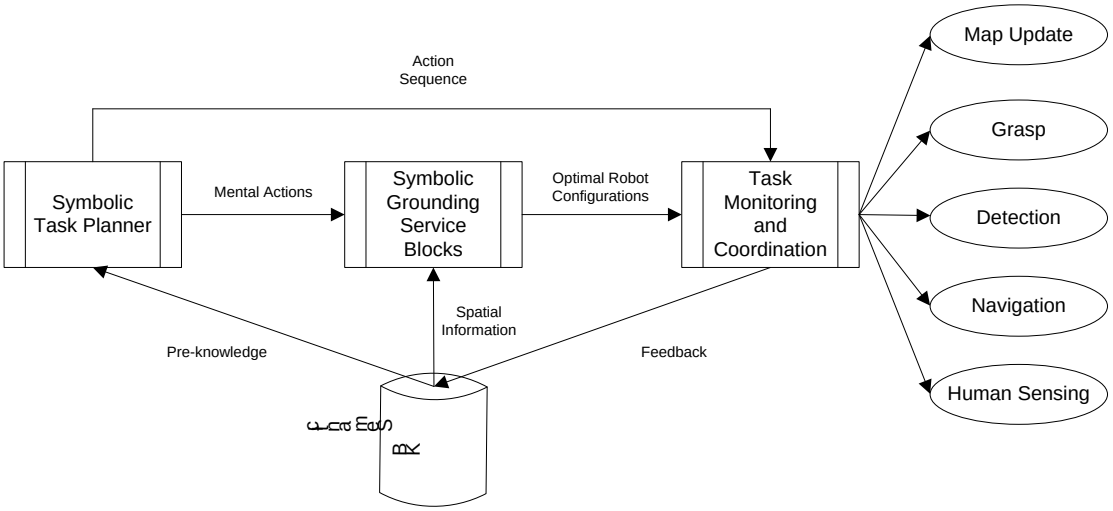


Figure 6.1 Architecture of the autonomous control framework

Proactive behaviours (action sequence) are generated on the left side by the task planner based on pre-knowledge of the environment from the Semantic KB and feedbacks from the task coordination. Requests of mental actions for grounding symbolic terms in the action sequence are sent to the symbolic grounding service blocks on the middle side. The symbolic grounding service blocks ground the symbolic terms into specific robot configurations based on the spatial information of objects in the environment. The spatial information is retrieved from the semantic KB. The robot configurations are sent to the task coordination. Reactive behaviours are realised on the right side by the task coordination. Feedbacks from the task coordination are sent back to the semantics KB.

The symbolic task planner uses a symbolic AI planning algorithm, named recursive back-trace searching, to plan a solution for a task. A task can be interpreted as a goal, described by a set of states of the world and the robot. For example, a task “get a milk box” implies the final state of “robot with a milk box on its tray”. The eventual objective is to satisfy a condition that the current states of the robot match the final goal state. This is achieved by individual action units into a valid sequence, which can be generated recursively by searching for actions

that match the corresponding conditions. To achieve the above-mentioned goal, a causal model of the actions is built based on the Stanford Research Institute Problem Solver (STRIPS). The model describes the affordances of an action and the effect of an action on the environment. For example, an action is defined as *move(base, near, MilkBox0)*. To implement actions at the level of sensing and actuation, symbolic terms in the actions such as “near” need to be grounded as specific robot configurations. The symbolic grounding process is referred to as a special type of action, named mental action here. The action sequence and mental actions generated for a getting milk box task is illustrated in Fig. 6.2.

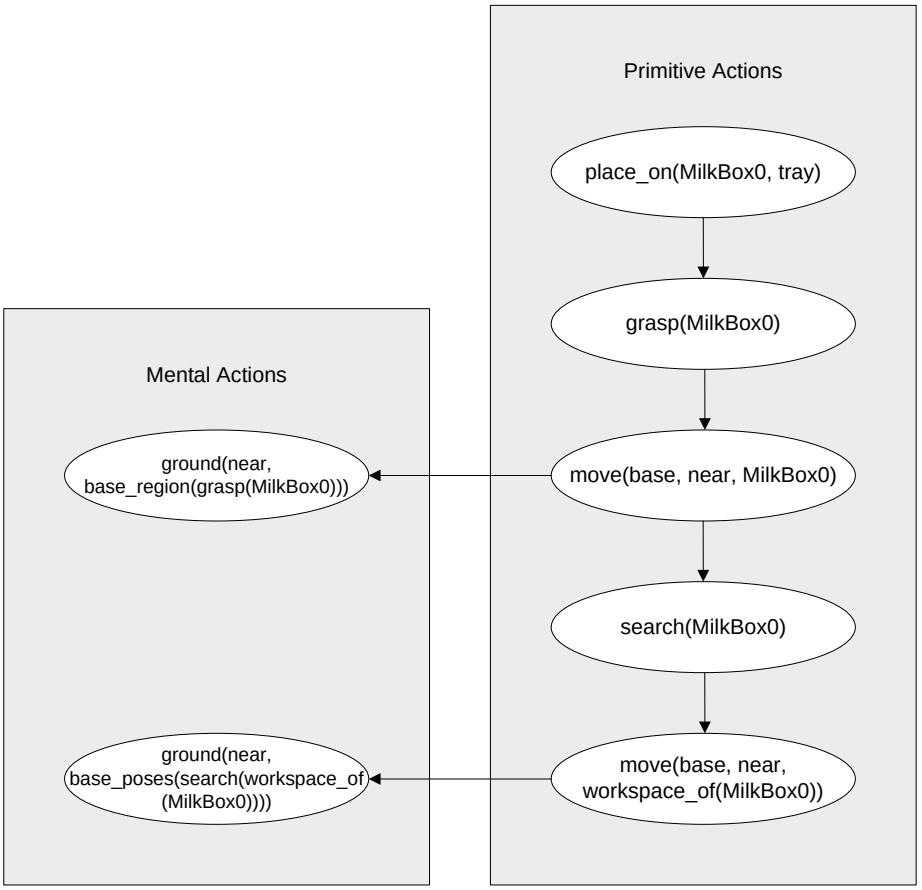


Figure 6.2 Action sequence and mental actions for a getting milk box task

The mental action *ground(near, base_region(grasp(MilkBox0)))* is used to calculate the optimal base region for the robot to grasp *MilkBox0*. The mental

action *ground(near, base_poses(search(workspace_of(MilkBox0))))* is used to calculate a list of robot base poses for the robot to search *MilkBox0*.

The semantic KB is developed based on the Ontology Web Language (OWL). For example, the knowledge of a home environment is saved in the semantic KB as semantic maps. There are two functional areas in the environment, a kitchen and a living room. In the semantic map (in the OWL format) of the kitchen, there is a fridge (labelled as *Fridge0*), a dishwasher (*Dishwasher0*), a stovetop (*Stove0*), a sink (*Sink0*), and an oven (*Oven0*). The living room instance contains a sofa (*Sofa0*) and a table (*Table0*). There is also an instance of milk box, named *MilkBox0* in the database. It has a property aboveOf in relation to an instance of dishwasher, named *Diswasher0*, as *object_on(MilkBox0, Dishwasher0)*. The property aboveOf is a sub-property of spatiallyRelated. The spatial information including position, shape and size of the objects in the environment is also saved in the semantic KB. The information can be updated during task implementation.

The task coordination has a four-layer structure. The concept is prototyped using ROS SMACH. The structure of the task coordination is shown in Fig. 6.3.

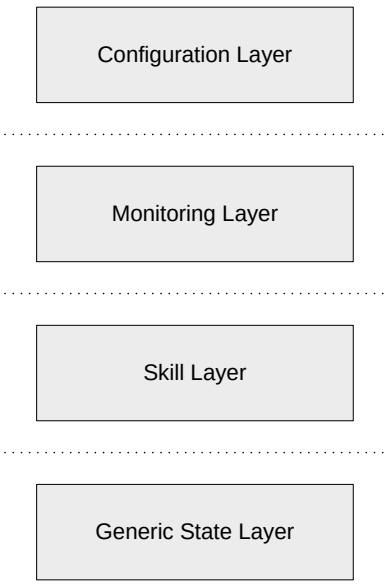


Figure 6.3 Structure of the task coordination

The top layer is called the “configuration layer”. It provides a unified interface for switching between different control logics. The logical pattern of the layer is implemented using a state machine.

The second layer is called the “monitoring layer”. It checks interventions from the higher level, and pre-empt the task coordination based on the defined logic.

The third layer is called the “skill layer”. This layer focuses on reusable skills, e.g. “pickup”, “environment update”, and “detection”. These skills and their application contexts in terms of pre and post-conditions are stored in the semantic KB as primitive actions. Some primitive actions may have their own action hierarchy. This is realised as nested state machines in the layer. For example, a primitive action DETECT_OBJECT enables a robot moving around a table for searching table-top objects. The coordination at this layer also collects information from sensors and sends it back to the semantic KB through outcomes. The grounded robot configurations will be sent to this layer for the robot to implement primitive actions.

The bottom layer is called the “generic state layer”. The generic states normally contain some clients which send requests. Lower level robot solutions such as navigation, manipulation, detection etc. can respond to the requests as services. The implementation is realised with the ROS actionlib.

6.1.2 Symbolic grounding service blocks

Two ROS service blocks for symbolic grounding are developed. A ROS service is a computer program that receives requests from a caller and sends out responses to a receiver. The requests to the service and the responds from the service can be defined as any Python type message. The first service block implements the mental action of *ground(near, base_poses(search(workspace_of(TargetObject))))*.

This mental action is used to ground the symbolic term “near” in the move robot base action *move(base, near, workspace_of(TargetObject))* into a list of robot base poses for searching the workspace of a target object. The workspace of a target object is the surface of the furniture where the target object is placed. This piece of furniture is called the parent object of the target object. Through searching the workspace (by implementing the action of *search(TargetObject)*), the exact location of the target object can be extracted and send to the Semantic KB.

A method for calculating robot base poses for searching a workspace is developed. The robot base poses are calculated based on the following criteria:

- The whole workspace is covered by the laser scanner equipped on the robot.
- The robot base poses are not occupied by obstacles.

To meet the first criterion, the scan area that can be covered by the laser scanner when the robot is placed at a specific pose is calculated using:

$$w_{scanner} = d_{scanner} * \tan(0.5 * \theta_{scanner}) \quad (6.1)$$

$$l_{scanner} = \sqrt{(d_{max})^2 - (h_{robot} - h_{parent})^2} - d_{robot} \quad (6.2)$$

$$d_{scanner} = \sqrt{(d_{robot})^2 + (h_{robot} - h_{parent})^2} \quad (6.3)$$

where $w_{scanner}$ is the width of the scan area, $d_{scanner}$ is the distance between the laser scanner and the edge of the parent object, $\theta_{scanner}$ is the detection angle of the laser scanner, $l_{scanner}$ is the length of the scan area, d_{max} is the maximum detection range of the laser scanner, h_{robot} is the height of the robot, h_{parent} is the height of the parent object, d_{robot} is the distance between the edge of the parent object and the centre of the robot base. $\theta_{scanner}$, d_{max}, h_{robot}, h_{parent} can be retrieved from the semantic KB. $d_{scanner}$ can be calculated using equation (6.3). The scan area calculated using equation (6.1) to equation (6.3) is illustrated in Fig. 6.4.

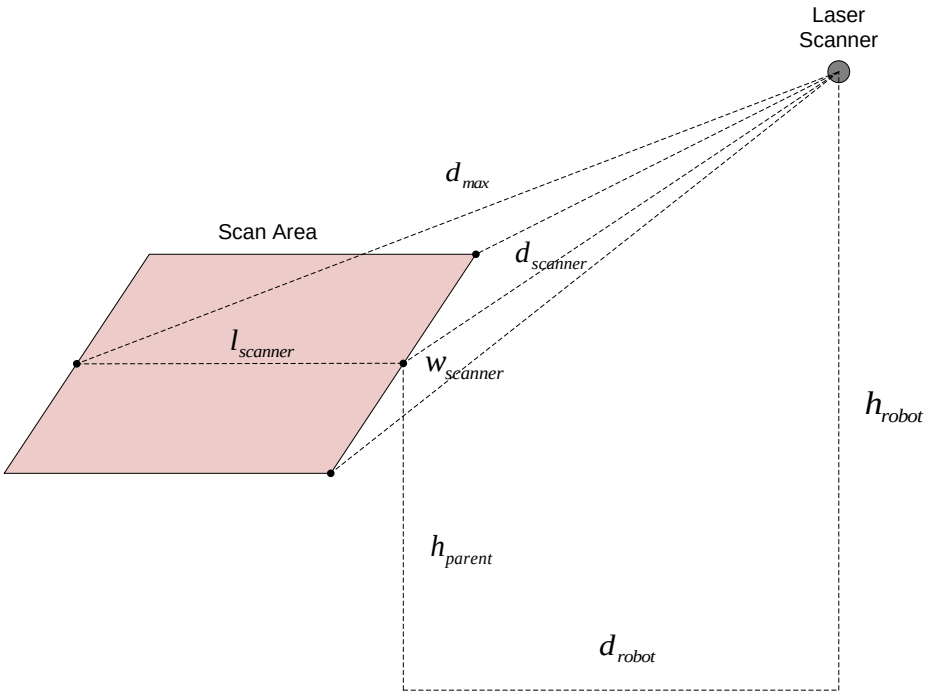


Figure 6.4 Scan area of the laser scanner

From the diagram, the scan area is the rectangular area (in red colour) located in front of the robot. The pose of the scan area is fixed to the pose of the robot base. By placing the scan area on the surface of a parent object, a list of robot base poses around the parent object can be calculated. The scan areas should cover the whole surface of the workspace. A list of robot base poses for scanning a rectangular workspace is illustrated in Fig. 6.5.

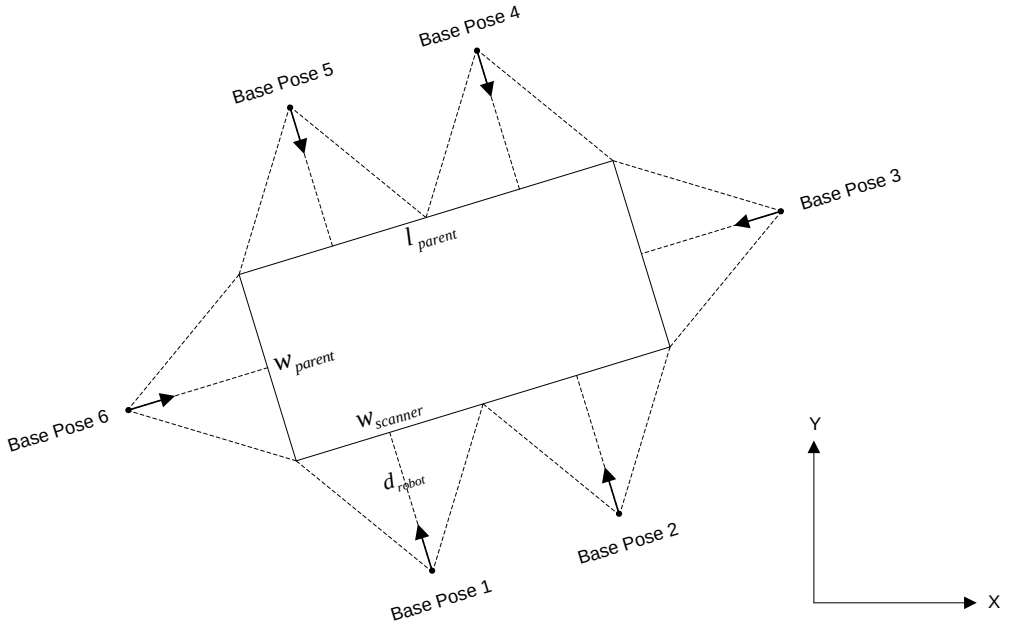


Figure 6.5 Robot base poses for scanning a rectangular workspace

In this diagram, $w_{scanner}$ can be calculated using equation (6.1). l_{parent} is the length of the parent object and w_{parent} is the width of the parent object. l_{parent} and w_{parent} can be retrieved from the semantic KB.

To meet the second criterion, the robot base poses occupied by obstacles will be removed from the list.

This method is realised as the symbolic grounding service block described in the first paragraph of this sub-section. The inputs to the service block include: 1) the mental action command *ground(near, base_poses(search(workspace_of(TargetObject))))* from the task planner; 2) the spatial information of the parent object and other furniture in the environment from the semantic KB. The outputs from the service block include: 1) the request to the semantic KB for retrieving the spatial information; 2) the grounded robot base poses to the skill layer of the task coordination.

The second service block implements the mental action of *ground(near, base_region(grasp(TargetObject)))*. This mental action is used to ground the symbolic term “near” in the move robot base action *move(base, near, TargetObject)* into the optimal base region for grasping the target object. The fuzzy optimisation algorithm described in Chapter 5 is realised as this service block. The inputs to the service block include: 1) the mental action command *ground(near, base_region(grasp(TargetObject)))* from the task planner; 2) the location of target object and the spatial information of furniture in the environment from the semantic KB. The outputs from the service block include: 1) the request to the semantic KB for retrieving the spatial information; 2) the grounded robot base poses to the skill layer of the task coordination.

6.1.3 Integrating symbolic grounding service blocks with autonomous control framework

The autonomous control framework is realised as ROS service blocks. A symbolic grounding service block for searching workspaces and a symbolic grounding service block for grasping objects were developed in the reported research in the previous three chapters by the author. Each service block processes requests from other service blocks and sends out response to its receiver. A service block can also send requests to other services for responses. The software architecture of the autonomous control framework is illustrated in Fig. 6.6.

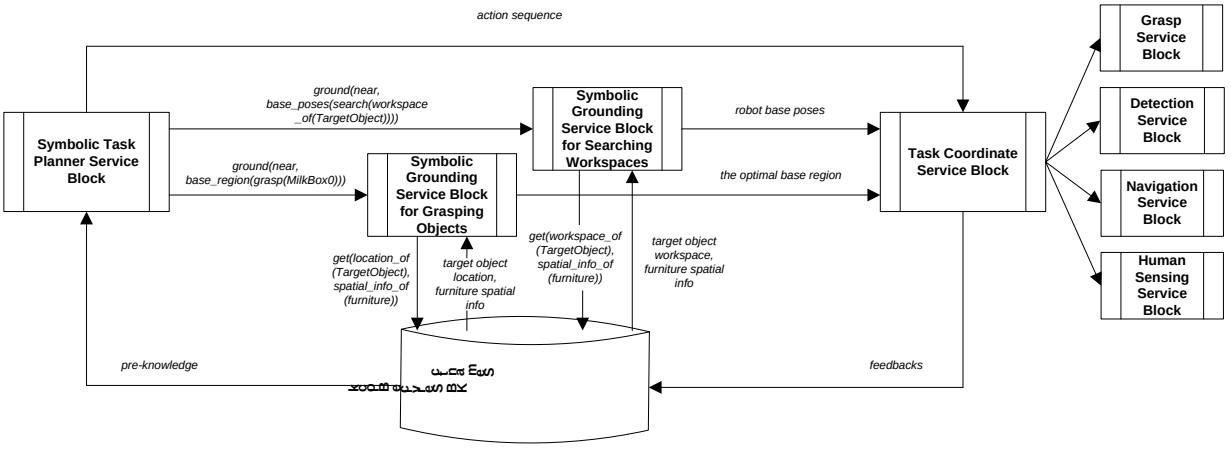


Figure 6.6 Software architecture of the autonomous control framework

From the diagram, the autonomous control framework consists of 9 service blocks:

Symbolic task planner service block: When receiving a task command from a human user, the symbolic task planner service block generates an action sequence for implementing the task based on the pre-knowledge of the environment from the semantic KB service block. Two mental actions for grounding the symbolic terms in the action sequence are also generate by this service block. The action sequence will be sent to the task coordination service block. The first mental action, which is in the format of *ground(near, base_poses(search(workspace_of(TargetObject))))*, will be sent to the symbolic grounding service block for searching workspaces. This mental action is used to ground the symbolic term "near" in the move robot base action *move(base, near, workspace_of(TargetObject))* into robot base poses for searching the workspace of the target object. The second mental action, which is in the format of *ground(near, base_region(grasp(TargetObject)))*, will be sent to the symbolic grounding service block for grasping objects. This mental action is used to ground the symbolic term "near" in the move robot base action *move(base, near, TargetObject)* into the optimal base region for grasping the target object.

Semantic KB service block: This service block receives requests from other service blocks for information of the environment. For example, when it receives a request *get(location_of(TargetObject), spatial_info_of(furniture))* from the symbolic grounding service block for grasping objects, it will send a respond, which includes the requested information to the symbolic grounding service block. The format of the respond is a 2D location of the target object and a list of spatial information of furniture in the environment. The spatial information includes length, width, height and a 3D pose of each piece of furniture in the environment. The semantic KB block also receives feedbacks from the task coordination service block. These feedbacks can be used as pre-knowledge of the environment.

Task coordination service block: This service block receives action sequence from the symbolic task planner service block and grounded robot configurations in terms of robot base poses and the optimal base region from the symbolic grounding service blocks. Based on the action sequence and the grounded robot configurations, the task coordination service block sends an action command for each action in the action sequence along with the grounded robot configuration to a corresponding service block for implementing the action. For example, an action command for implementing the *move(base, near, TargetObject)* action along with the optimal base region for grasping the target object will be sent to the navigation service block. Results of action implementation will be sent back to the semantic KB service block. For example, the result of the *Search(TargetObjetct)* action, which is the extracted target object location will be sent back to the semantic KB. The extracted target object location can be used as spatial information of the environment.

Service blocks for implementing specific actions: There are four service blocks for implementing specific actions:

1) grasp service block: it implements the action of *grasp(TargetObject)* when the robot base is placed within the optimal base region.
2) detection service block: it implements the action of *search(TargetObject)* from the robot base poses grounded by the symbolic grounding service

block. A surface scan will be conducted from each grounded base pose until the target object is extracted.

3) navigation service block: it implements the action of *move(base, near, TargetObject)* or *move(base, near, workspace_of(TargetObject))* when a robot base pose or the optimal base region is given.

4) human sensing service block: it detects the location of a human in the environment.

Symbolic grounding service block for searching workspaces: This service block implements the mental action of *ground(near, base_poses(search (workspace_of(TargetObject))))* using the method described in Section 6.1.2. When receiving the mental action command from the symbolic task planner service block, it sends a request *get(workspace_of(TargetObject), spatial_info_of(furniture))* to the semantic KB service block for retrieving the spatial information of the workspace and furniture in the environment. The spatial information of the workspace (in the format of *(length, width, height, 3D_pose)*) is used to calculate the robot base poses for searching the workspace. The spatial information of furniture (in the format of (*(length_1, width_1, 3D_pose_1), (length_2, width_2, 3D_pose_2),…)*) is used to determine which robot base poses are occupied by obstacles. The grounded robot base poses (in the format of *(3D_pose_1, 3D_pose_2,…)*) will be sent to the task coordination service block.

Symbolic grounding service block for grasping objects: This service block implements the mental action of *ground(near, base_region(grasp(TargetObject)))* using the fuzzy optimisation algorithm described in Chapter 5. When receiving the mental action command from the symbolic task planner service block, it sends a request *get(location_of(TargetObject), spatial_info_of(furniture))* to the semantic KB service block for retrieving the target object location and the spatial information of furniture in the environment. The target object location (in the format of *3D_pose*) is used to establish the fuzzy objective function. The spatial information of furniture (in the format of (*(length_1, width_1, 3D_pose_1), (length_2, width_2, 3D_pose_2),…)*) is used to establish fuzzy constraints for

performing fuzzy optimisation. The grounded optimal base region will be sent to the task coordination service block.

6.2 Experiments and Results

Three object fetching experiments were carried out where a homecare robot, Care-O-bot 3, tried to implement symbolic action sequences of fetching objects placed at different locations in two different domestic environments. Symbolic terms in the action sequences were grounded into specific robot configurations by the symbolic grounding service blocks. The robot implemented the action sequences according to the configurations. The aim, settings and results of the experiments are presented in this section.

6.2.1 Experiments

The aim of the experiments is to test whether the symbolic grounding service blocks can ground symbolic action sequences generated by the task planner into specific robot configurations in terms of robot base poses and the optimal base regions and whether the grounded robot configurations can support a homecare robot platform, Care-O-bot 3, to successfully implement the corresponding action sequences in domestic environments.

The first two experiments were carried out in a kitchen environment at Fraunhofer IPA in Stuttgart, Germany. A picture of the kitchen environment and the robot, Care-O-bot 3, is shown in Fig. 6.7.

Figure 6.7 Kitchen environment in Stuttgart and Care-O-bot 3

In the kitchen, there was a fridge (labelled as *Fridge0*), a dishwasher (*Diswasher0*), a stove (*Stove0*), a sink (*Sink0*), an oven (*Oven0*), a table (*Table0*), and a sofa (*Sofa0*). The spatial information, which includes length, width, height and pose (in the format of (x, y, θ) in the world coordinate) of the furniture in the kitchen was save in the semantic KB as pre-knowledge of the environment. The spatial information can be updated during task implementation.

The layout of the kitchen environment is illustrated in Fig. 6.8.

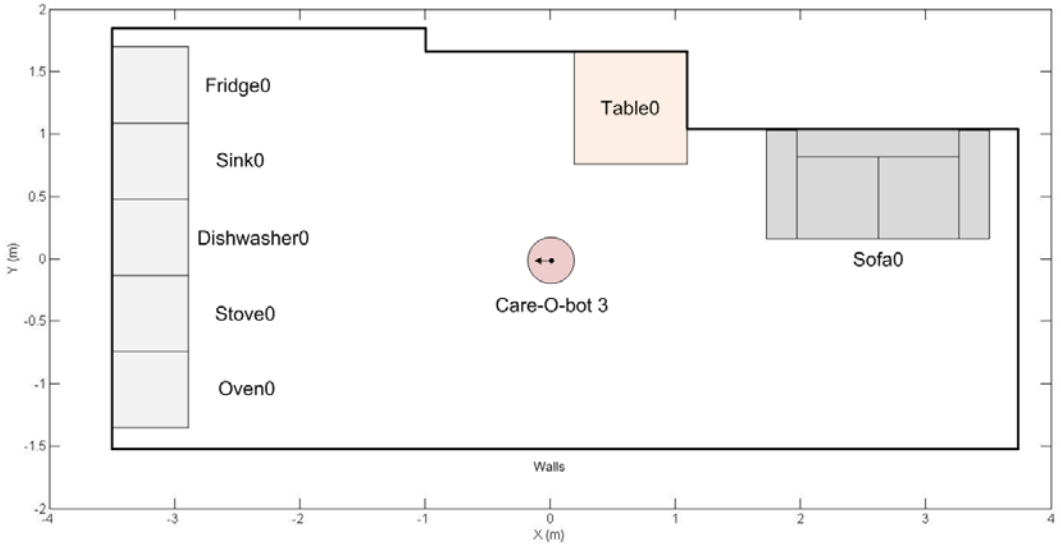


Figure 6.8 Layout of the kitchen environment

The spatial information of the furniture in the kitchen is shown in Table 6.1.

Spatial Info / Furniture	Location		Orientation	Length	Width	Height
	X (*m*)	Y (*m*)	(*Deg*)	(*m*)	(*m*)	(*m*)
Table0	0.65	1.21	0.0	0.9	0.9	0.74
Oven0	-3.2	-1.04	0.0	0.6	0.6	1.4
Stove0	-3.2	-0.54	0.0	0.6	0.6	0.85
Fridge0	-3.2	1.36	0.0	0.6	0.6	1.4
Sofa0	2.625	0.595	0.0	0.89	1.75	0.45
Sink0	-3.2	0.76	0.0	0.6	0.6	1.4
Dishwasher0	-3.2	0.159	0.0	0.6	0.6	1.85

Table 6.1 Spatial information of the furniture in the kitchen environment

The third experiment was carried out in a home environment at Don Carlo Gnocchi hospital in Milan, Italy. A picture of the home environment and Care-O-bot 3 is shown in Fig. 6.9.

Figure 6.9 Home environment in Milan and Care-O-bot 3

In the home environment, there were two tables (labelled as *KitchenTableLeft* and *KitchenTableRight*), a shelf (*IkeaShelfMilan*), an oven (*Oven0*), a TV console-table (*TVConsolleMilan*), a sofa (*Sofa0*), a bedside-table (*BedsideMilan*), and a bed (*SRSBedMilan*).

The layout of the home environment is illustrated in Fig. 6.10.

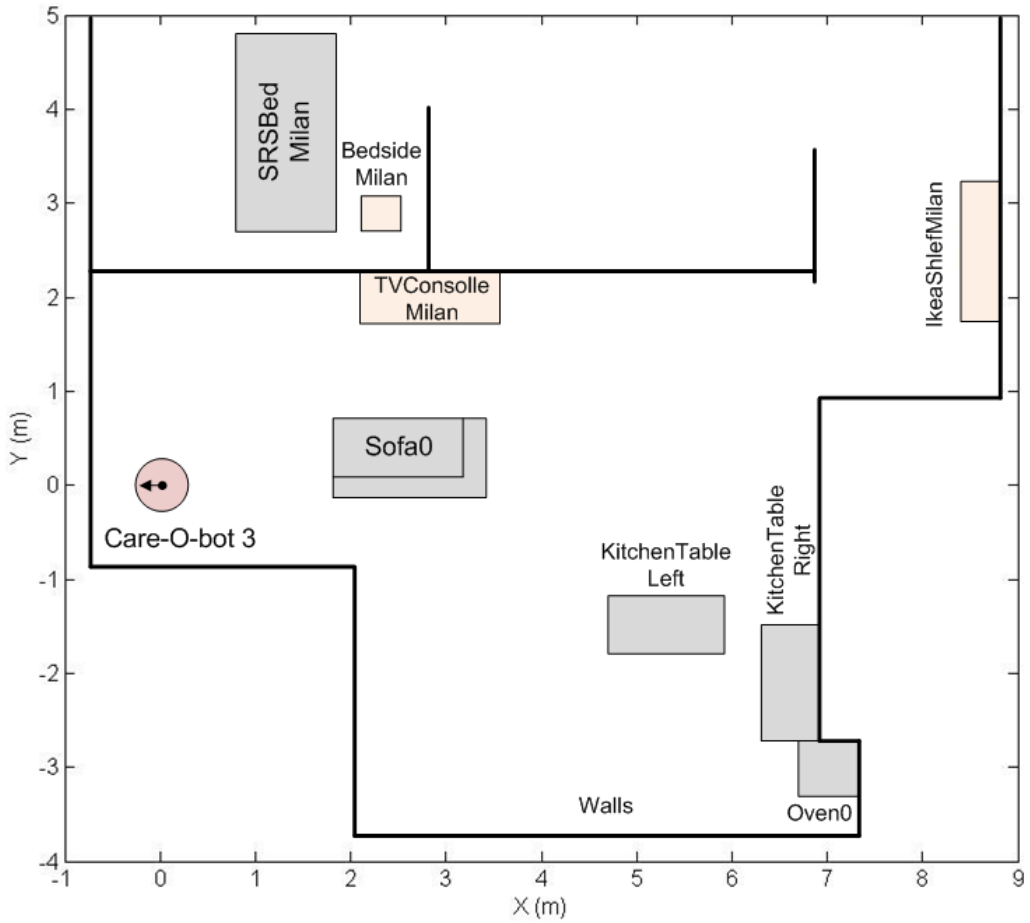


Figure 6.10 Layout of the home environment

The spatial information of the furniture in the home environment is shown in Table 6.2.

Spatial Info / Furniture	Location		Orientation	Length	Width	Height
	X (*m*)	Y (*m*)	(*Deg*)	(*m*)	(*m*)	(*m*)
KitchenTableLeft	5.3	-1.5	90.0	1.2	0.6	0.8
IkeaShelfMilan	8.6	2.5	0.0	1.5	0.4	0.8
KitchenTableRight	6.6	-2.1	0.0	1.2	0.6	0.8
Oven0	-7.0	-3.0	0.0	0.65	0.6	1.45
TVConsolleMilan	2.8	2.0	90.0	1.49	0.55	0.35
Sofa0	2.6	0.3	-90.0	1.6	0.87	0.73
BedsideMilan	2.3	2.9	-90.0	0.43	0.45	0.87
SRSBedMilan	1.3	3.9	0.0	2.1	1.03	0.81

Table 6.2 Spatial information of the furniture in the home environment

In the first experiment, a milk box (labelled as *MilkBox0*) was placed on the dishwasher (*Dishwasher0*) at (-2.95*m*, 0.2*m*) in the world coordinate. In the second experiment, *MilkBox0* was placed on the table (*Table0*) at (0.65*m*, 0.79*m*). In the third experiment, a medicine box (labelled as *Madicine0*) was placed on the shelf at (8.47*m*, 2.5*m*). The robot was initially placed at (0.0*m*, 0.0*m*, 0.0°) in the three experiments.

6.2.2 Results

In the first experiment, a “get milk” task was given to the robot by a human user. The symbolic task planner service block generated an action sequence for implementing the task and two mental actions for grounding the action sequence. The action sequence and the mental actions are illustrated in Fig. 6.11.

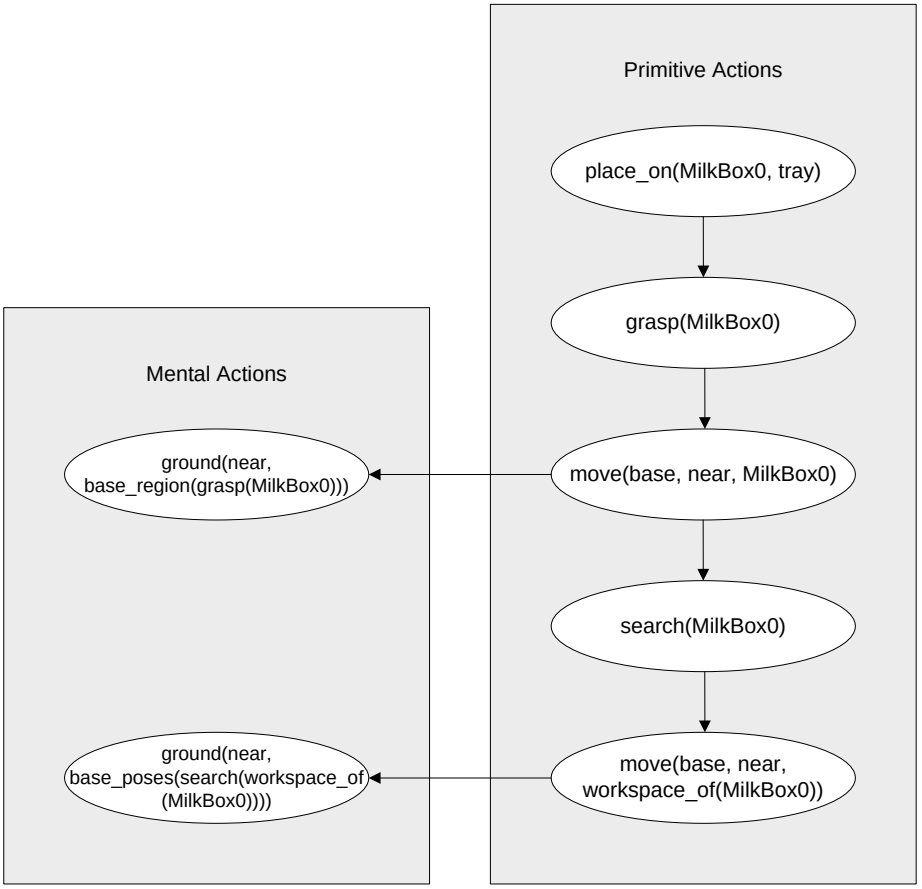


Figure 6.11 Action sequence and mental actions for implementing “get milk” task

Two mental actions were generated by the symbolic task coordination service block for grounding the action sequence. The first mental action, *ground(near, base_poses(search(workspace_of(MilkBox0))))*, was sent to the symbolic grounding service block for searching workspaces. To implement the mental action, the symbolic grounding service block sent a request, *get(workspace_of(MilkBox0), spatial_info_of(furniture))*, to the semantic KB service block for retrieving spatial information. A response was sent back to the symbolic grounding service block. The response consisted of the spatial information of the workspace and furniture in the environment:

[workspace: (0.6*m*, 0.6*m*, 1.85*m*, (-3.2*m*, 0.159*m*, 0.0°)),
furniture_spatial_info:

(0.9*m*, 0.9*m*, 0.74*m*, (0.65*m*, 1.21*m*, 0.0°)),
(0.6*m*, 0.6*m*, 1.4*m*, (-3.2*m*, -1.04*m*, 0.0°)),
(0.6*m*, 0.6*m*, 0.85*m*, (-3.2*m*, -0.54*m*, 0.0°)),
(0.6*m*, 0.6*m*, 1.4*m*, (-3.2*m*, 1.36*m*, 0.0°)),
(0.89*m*, 1.75*m*, 0.45*m*, (2.625*m*, 0.595*m*, 0.0°)),
(0.6*m*, 0.6*m*, 1.4*m*, (-3.2*m*, 0.76*m*, 0.0°))].

According to the spatial information of the workspace, 4 robot base poses for searching the workspace were calculated using the method described in Section 6.1.2:

[robot_base_pose_1: (-2.3*m*, 0.159*m*, 0.0°),
robot_base_pose_2: (-3.2*m*, -0.741*m*, -90.0°),
robot_base_pose_3: (-4.1*m*, 0.159*m*, 180.0°),
robot_base_pose_4: (-3.2*m*, 1.059*m*, 90.0°)].

According to the spatial information of furniture in the environment, robot base poses occupied by obstacles were deleted. The obstacle-free robot base pose for searching the workspace was:

[robot_base_pose_1: (-2.3*m*, 0.159*m*, 0.0°)].

This robot base pose was sent to the task coordination service block. The task coordination service block sent a request along with the grounded robot base pose to the navigation service block for implementing the move robot base action *move(base, near, workspace_of(MilkBox0))*. After the robot moved to that pose, detection was carried out and the exact location of *MilkBox0* was extracted. The location of *MilkBox0*, which was (-2.95*m*, 0.2*m*), was sent to the semantic KB service block.

The robot base pose for searching the workspace and the movement trajectory of the robot base is shown in Fig. 6.12.

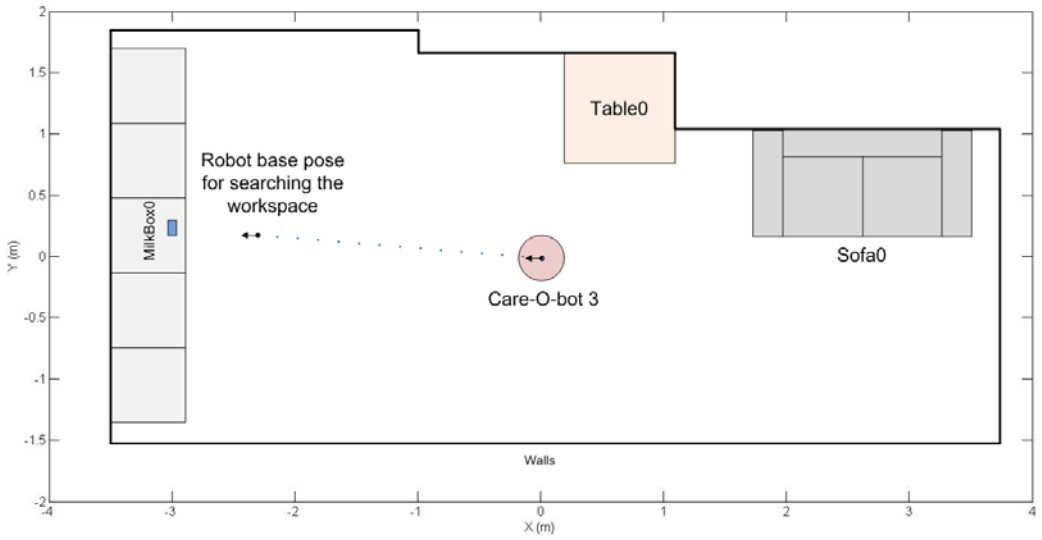


Figure 6.12 Robot base pose for searching the workspace and movement trajectory of the robot base (*MilkBox0* placed at (-2.95*m*, 0.2*m*))

The second mental action, *ground(near, base_region(grasp(MilkBox0)))*, was sent to the symbolic grounding service block for grasping objects. To implement the mental action, the symbolic grounding service block sent a request, *get(location_of(MilkBox0), spatial_info_of(furniture))*, to the semantic KB service block for retrieving spatial information. A response was sent back to the symbolic grounding service block. The response consisted of the location of *MilkBox0* and the spatial information of the furniture in the environment:

[location_of_MilkBox0: (-2.95*m*, 0.2*m*),

furniture_spatial_info:

(0.9*m*, 0.9*m*, 0.74*m*, (0.65*m*, 1.21*m*, 0.0°)),

(0.6*m*, 0.6*m*, 1.4*m*, (-3.2*m*, -1.04*m*, 0.0°)),

(0.6*m*, 0.6*m*, 0.85*m*, (-3.2*m*, -0.54*m*, 0.0°)),

(0.6*m*, 0.6*m*, 1.4*m*, (-3.2*m*, 1.36*m*, 0.0°)),

(0.89*m*, 1.75*m*, 0.45*m*, (2.625m, 0.595m, 0.0°)),

(0.6*m*, 0.6*m*, 1.4*m*, (-3.2*m*, 0.76*m*, 0.0°)),

(0.6*m*, 0.6*m*, 1.85*m*, (-3.2*m*, 0.159*m*, 0.0°))].

According to the location of *MilkBox0*, an FRS was established using the method described in Chapter 3. According to the spatial information of furniture in the

environment, fuzzy constraints were established using the method described in Chapter 4. Based on the FRS and the fuzzy constraints, the optimal base pose for grasping *MilkBox0* and its constraint value were calculated using the constrained fuzzy optimisation algorithm described in Chapter 4. The optimal base pose was (-2.11m, 0.2m, 0.0°) and its constraint value was 1.0. Based on the FRS and the constraint value of the optimal base pose, a fuzzy objective function was established using the method described in Chapter 5. Based on the fuzzy objective function and the fuzzy constraints, the optimal based region for grasping *MilkBox0* was calculated using the fuzzy optimisation algorithm described in Chapter 5. The optimal base region was:

$$\{(x,y)|x \geq -2.14, \sqrt{(x+2.95)^2+(y-0.2)^2} < 0.84\}$$

The optimal base region was sent to the task coordination service block. The task coordination service block sent a request along with the grounded optimal base region to the navigation service block for implementing the move robot base action *move(base, near, MilkBox0)*. After the robot base was within the optimal base region, the robot started to grasp *MilkBox0* and the milk box was successfully grasped.

The optimal base region for grasping *MilkBox0* and the movement trajectory of the robot base is shown in Fig. 6.13.

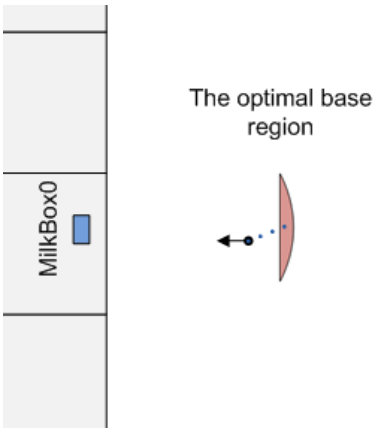


Figure 6.13 The optimal base region and movement trajectory of the robot base (*MilkBox0* placed at (-2.95m, 0.2m))

In this diagram, the area in red colour is the optimal base region. The robot moved from the base pose for searching the workspace to the optimal base region.

The second experiment was carried out in the same kitchen environment. However, *MilkBox0* was placed on *Table0* at (0.65*m*, 0.79*m*) this time. The same "get milk" task command was given to the robot. The symbolic task planner service block generated an action sequence for implementing the task and two mental actions for grounding the action sequence. The action sequence and the mental actions were the same as the first experiment. The mental action, *ground(near, base_poses(search(workspace_of(MilkBox0))))*, was sent to the symbolic grounding service block for searching workspaces. The symbolic grounding service block sent a request, *get(workspace_of(MilkBox0), spatial_info_of(furniture))*, to the semantic KB service block for retrieving spatial information. A response was sent back to the symbolic grounding service block. The response consisted of the spatial information of the workspace and furniture in the environment:

[workspace: (0.9*m*, 0.9*m*, 0.74*m*, (0.65*m*, 1.21*m*, 0.0°)),
furniture_spatial_info:
(0.6*m*, 0.6*m*, 1.4*m*, (-3.2*m*, -1.04*m*, 0.0°)),
(0.6*m*, 0.6*m*, 0.85*m*, (-3.2*m*, -0.54*m*, 0.0°)),
(0.6*m*, 0.6*m*, 1.4*m*, (-3.2*m*, 1.36*m*, 0.0°)),
(0.89*m*, 1.75*m*, 0.45*m*, (2.625*m*, 0.595*m*, 0.0°)),
(0.6*m*, 0.6*m*, 1.4*m*, (-3.2*m*, 0.76*m*, 0.0°)),
(0.6*m*, 0.6*m*, 1.85*m*, (-3.2*m*, 0.159*m*, 0.0°))].

According to the spatial information of the workspace, 8 robot base poses for searching the workspace were calculated using the method described in Section 6.1.2:

[robot_base_pose_1: (1.7*m*, 1.44*m*, 0.0°),
robot_base_pose_2: (1.7*m*, 0.99*m*, 0.0°),
robot_base_pose_3: (0.43*m*, 0.16*m*, -90.0°),
robot_base_pose_4: (0.88*m*, 0.16*m*, -90.0°),
robot_base_pose_5: (-0.4*m*, 1.44*m*, 180.0°),

robot_base_pose_6: (-0.4*m*, 0.99*m*, 180.0°),

robot_base_pose_7: (0.43*m*, 2.26*m*, 90.0°),

robot_base_pose_8: (0.88*m*, 2.26*m*, 90.0°)].

According to the spatial information of furniture in the environment, robot base poses occupied by obstacles were deleted. The obstacle-free robot base pose for searching the workspace was:

[robot_base_pose_3: (0.43*m*, 0.16*m*, -90.0°),

robot_base_pose_4: (0.88*m*, 0.16*m*, -90.0°)].

The two robot base poses were sent to the task coordination service block. The task coordination service block sent a request along with the grounded robot base poses to the navigation service block for implementing the move robot base action *move(base, near, workspace_of(MilkBox0))*. After the robot moved to robot_base_pose_3, detection was carried out and the exact location of *MilkBox0* was extracted. The location of *MilkBox0*, which was (0.65*m*, 0.79*m*), was sent to the semantic KB service block.

The robot base pose for searching the workspace and the movement trajectory of the robot base is shown in Fig. 6.14.

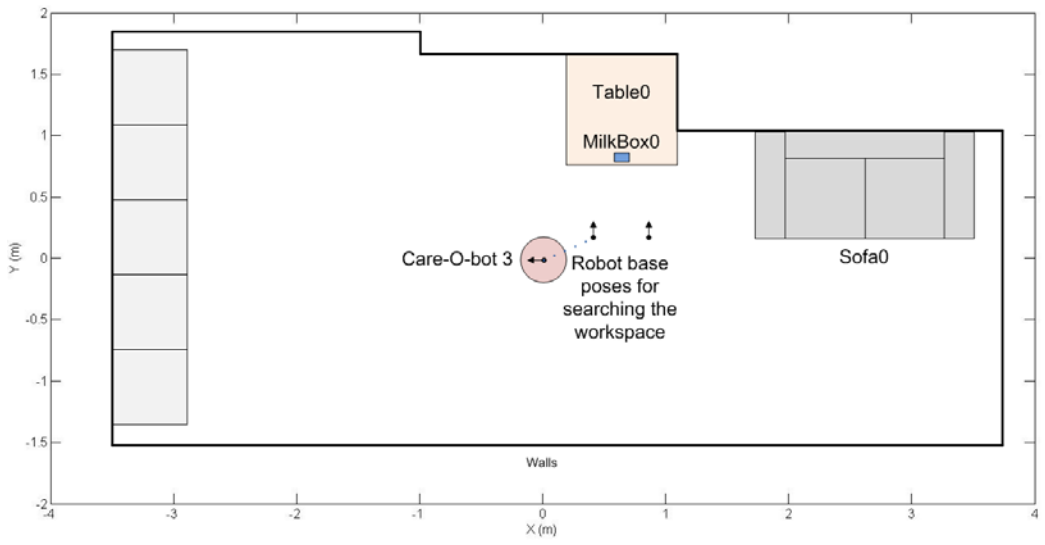


Figure 6.14 Robot base pose for searching the workspace and movement trajectory of the robot (*MilkBox0*placed at (0.65*m*, 0.79*m*))

The second mental action, *ground(near, base_region(grasp(MilkBox0)))*, was sent to the symbolic grounding service block for grasping objects. The symbolic grounding service block sent a request, *get(location_of(MilkBox0), spatial_info_of(furniture))*, to the semantic KB service block for retrieving spatial information. A response was sent back to the symbolic grounding service block. The response consisted of the location of *MilkBox0* and the spatial information of the furniture in the environment:

[location_of_MilkBox0: (0.65*m*, 0.79*m*),
furniture_spatial_info:
(0.9*m*, 0.9*m*, 0.74*m*, (0.65*m*, 1.21*m*, 0.0°)),
(0.6*m*, 0.6*m*, 1.4*m*, (-3.2*m*, -1.04*m*, 0.0°)),
(0.6*m*, 0.6*m*, 0.85*m*, (-3.2*m*, -0.54*m*, 0.0°)),
(0.6*m*, 0.6*m*, 1.4*m*, (-3.2*m*, 1.36*m*, 0.0°)),
(0.89*m*, 1.75*m*, 0.45*m*, (2.625m, 0.595m, 0.0°)),
(0.6*m*, 0.6*m*, 1.4*m*, (-3.2*m*, 0.76*m*, 0.0°)),
(0.6*m*, 0.6*m*, 1.85*m*, (-3.2*m*, 0.159*m*, 0.0°))].

The altitude of *MilkBox0*was 0.74*m*, an FRS for grasping lower-altitude objects was established (refer to Section 3.3 of Chapter 3).According to the spatial information of furniture in the environment, fuzzy constraints were established using the method described in Chapter 4. Based on the FRS and the fuzzy constraints, the optimal base pose for grasping *MilkBox0* and its constraint value were calculated using the constrained fuzzy optimisation algorithm described in Chapter 4. The optimal base pose was (0.65*m*, 0.08*m*, -90.0°) and its constraint value was 1.0. Based on the FRS and the constraint value of the optimal base pose, a fuzzy objective function was established using the method described in Chapter 5. Based on the fuzzy objective function and the fuzzy constraints, the optimal based region for grasping *MilkBox0* was calculated using the fuzzy optimisation algorithm described in Chapter 5. The optimal base region was:

$$\{(x,y)|y \leq 0.09, \sqrt{(x-0.65)^2+(y-0.79)^2} < 0.73\}$$

The optimal base region was sent to the task coordination service block. The task coordination service block sent a request along with the grounded optimal base region to the navigation service block for implementing the move robot base action *move(base, near, MilkBox0).* After the robot base was within the optimal base region, the robot started to grasp *MilkBox0* and the milk box was successfully grasped.

The optimal base region for grasping *MilkBox0* and the movement trajectory of the robot base is shown in Fig. 6.15.

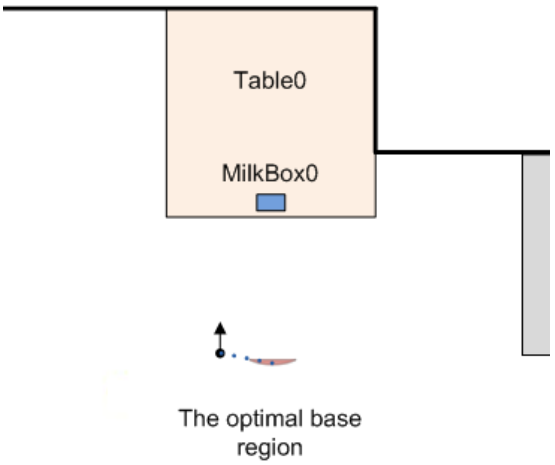


Figure 6.15 The optimal base region and movement trajectory of the robot base (*MilkBox0*placed at (0.65*m*, 0.79*m*))

In this diagram, the area in red colour is the optimal base region. The robot moved from the base pose for searching the workspace to the optimal base region.

The third experiment was carried out in the home environmentat Don Carlo Gnocchi hospital. Amedicine box (*Medicine0*) was placed on *IkeaShelfMilan* at (8.47*m*, 2.5*m*) in the world coordinate. A “get medicine” task command was given to the robot. The symbolic task planner service block generated an action sequence for implementing the task and two mental actions for grounding the

action sequence. The action sequence and the mental actions are shown in Fig. 6.16.

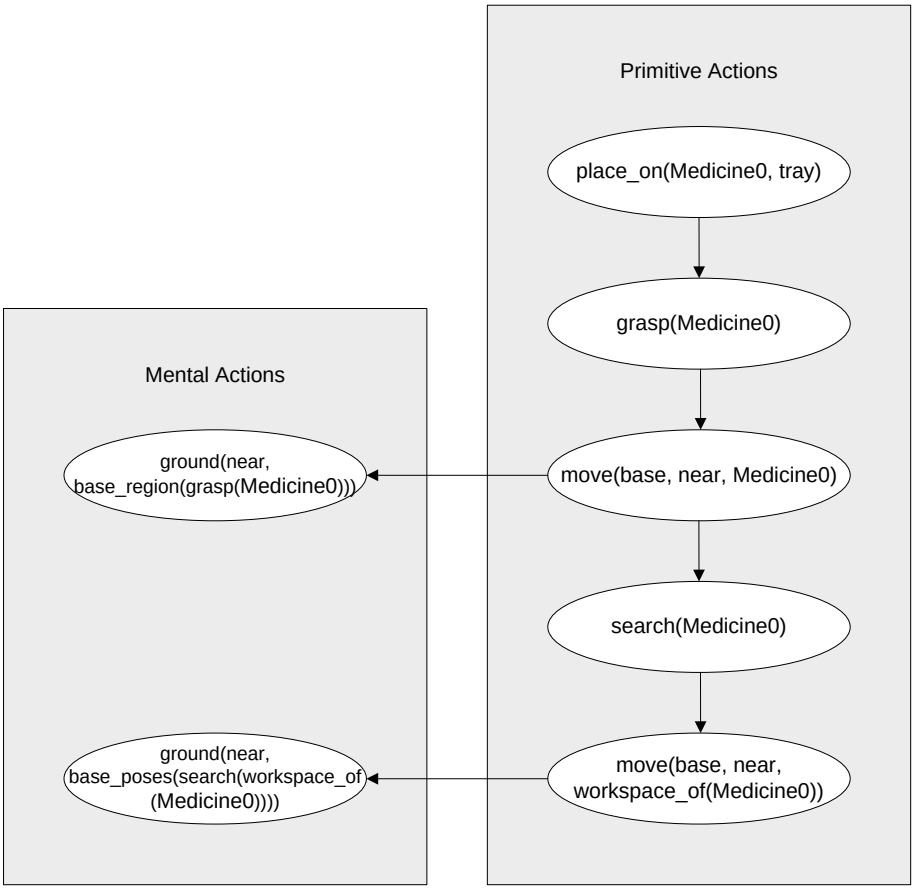


Figure 6.16 Action sequence and mental actions for implementing “get medicine” task

The mental action, *ground(near, base_poses(search(workspace_of(Medicine0))))*, was sent to the symbolic grounding service block for searching workspaces. The symbolic grounding service block sent a request, *get(workspace_of(Medicine0), spatial_info_of(furniture))*, to the semantic KB service block for retrieving spatial information. A response was sent back to the symbolic grounding service block. The response consisted of the spatial information of the workspace and furniture in the environment:

[workspace: (1.5*m*, 0.4*m*, 0.8*m*, (8.6*m*, 2.5*m*, 0.0°)),

furniture_spatial_info:

(1.2*m*, 0.6*m*, 0.8*m*, (5.3*m*, -1.5*m*, 90.0°)),

(1.2*m*, 0.6*m*, 0.8*m*, (6.6*m*, -2.1*m*, 0.0°)),

(0.65*m*, 0.6*m*, 1.45*m*, (-7.0*m*, -3.0*m*, 0.0°)),

(1.49*m*, 0.55*m*, 0.35*m*, (2.8*m*, 2.0*m*, 90.0°)),

(1.6*m*, 0.87*m*, 0.73*m*, (2.6*m*, 0.3*m*, -90.0°)),

(0.43*m*, 0.45*m*, 0.87*m*, (2.3*m*, 2.9*m*, -90.0°)),

(2.1*m*, 1.03*m*, 0.81*m*, (1.3*m*, 3.9*m*, 0.0°))].

According to the spatial information of the workspace, 8 robot base poses for searching the workspace were calculated using the method described in Section 6.1.2:

[robot_base_pose_1: (9.4*m*, 3.0*m*, 0.0°),

robot_base_pose_2: (9.4*m*, 2.5*m*, 0.0°),

robot_base_pose_3: (9.4*m*, 2.0*m*, 0.0°),

robot_base_pose_4: (8.6*m*, 1.15*m*, -90.0°),

robot_base_pose_5: (7.8*m*, 3.0*m*, 180.0°),

robot_base_pose_6: (7.8*m*, 2.5*m*, 180.0°),

robot_base_pose_7: (7.8*m*, 2.0*m*, 180.0°),

robot_base_pose_8: (8.6*m*, 3.85*m*, 90.0°)].

According to the spatial information of furniture in the environment, robot base poses occupied by obstacles were deleted. The obstacle-free robot base pose for searching the workspace was:

[robot_base_pose_5: (7.8*m*, 3.0*m*, 180.0°),

robot_base_pose_6: (7.8*m*, 2.5*m*, 180.0°),

robot_base_pose_7: (7.8*m*, 2.0*m*, 180.0°)]

The robot base poses were sent to the task coordination service block. The task coordination service block sent a request along with the grounded robot base poses to the navigation service block for implementing the move robot base action *move(base, near, workspace_of(Medicine0))*. *Medicine0* was detectedafter the robot moved to robot_base_pose_6. The location of *Medicine0*, which was (8.47*m*, 2.5*m*), was sent to the semantic KB service block.

The robot base pose for searching the workspace and the movement trajectory of the robot base is shown in Fig. 6.17.

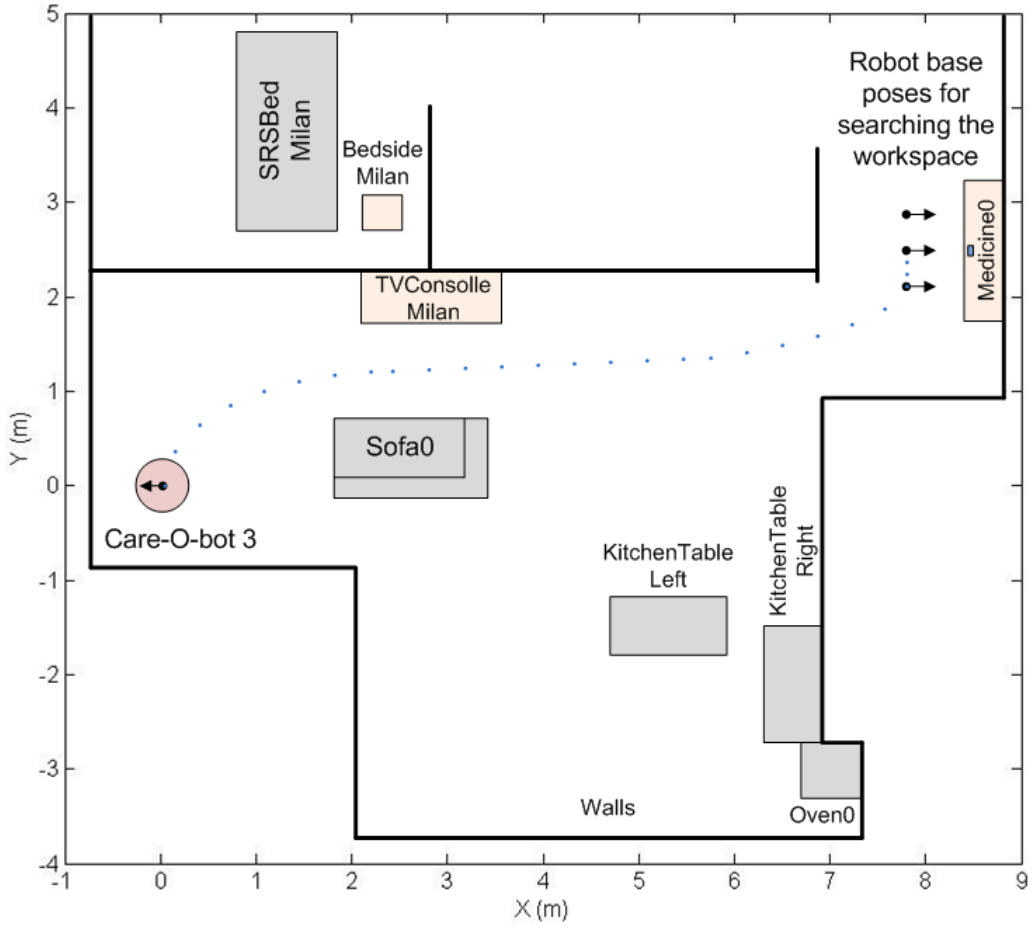


Figure 6.17 Robot base pose for searching the workspace and movement trajectory of the robot (*Medicine0* placed at (8.47*m*, 2.5*m*))

The second mental action, *ground(near, base_region(grasp(Medicine0)))*, was sent to the symbolic grounding service block for grasping objects. The symbolic grounding service block sent a request, *get(location_of(Medicine0), spatial_info_of(furniture))*, to the semantic KB service block for retrieving spatial information. A response was sent back to the symbolic grounding service block. The response consisted of the location of *Medicine0* and the spatial information of the furniture in the environment:

[location_of_Medicine0: (8.47*m*, 2.5*m*),
furniture_spatial_info:
(1.2*m*, 0.6*m*, 0.8*m*, (5.3*m*, -1.5*m*, 90.0°)),
(1.5*m*, 0.4*m*, 0.8*m*, (8.6*m*, 2.5*m*, 0.0°)),
(1.2*m*, 0.6*m*, 0.8*m*, (6.6*m*, -2.1*m*, 0.0°)),
(0.65*m*, 0.6*m*, 1.45*m*, (-7.0*m*, -3.0*m*, 0.0°)),
(1.49*m*, 0.55*m*, 0.35*m*, (2.8*m*, 2.0*m*, 90.0°)),
(1.6*m*, 0.87*m*, 0.73*m*, (2.6*m*, 0.3*m*, -90.0°)),
(0.43*m*, 0.45*m*, 0.87*m*, (2.3*m*, 2.9*m*, -90.0°)),
(2.1*m*, 1.03*m*, 0.81*m*, (1.3*m*, 3.9*m*, 0.0°))].

According to the location of *Medicine0,*an FRS was established using the method described in Chapter 3. According to the spatial information of furniture in the environment, fuzzy constraints were established using the method described in Chapter 4. Based on the FRS and the fuzzy constraints, the optimal base pose for grasping *Medicine0* and its constraint value were calculated using the constrained fuzzy optimisation algorithm described in Chapter 4. The optimal base pose was (7.64*m*, 2.5*m*, 180.0°) and its constraint value was 1.0. Based on the FRS and the constraint value of the optimal base pose, a fuzzy objective function was established using the method described in Chapter 5. Based on the fuzzy objective function and the fuzzy constraints, the optimal based region for grasping *Medicine0* was calculated using the fuzzy optimisation algorithm described in Chapter 5. The optimal base region was:

$$\{(x, y) | x \leq 7.64, \sqrt{(x - 8.47)^2 + (y - 2.5)^2} < 0.84\}$$

The optimal base region was sent to the task coordination service block. The task coordination service block sent a request along with the grounded optimal base region to the navigation service block for implementing the move robot base action *move(base, near, Medicine0).* After the robot base was within the optimal base region, the robot started to grasp *Medicine0* and the milk box was successfully grasped.

The optimal base region for grasping *Medicine0* and the movement trajectory of the robot base is shown in Fig. 6.18.

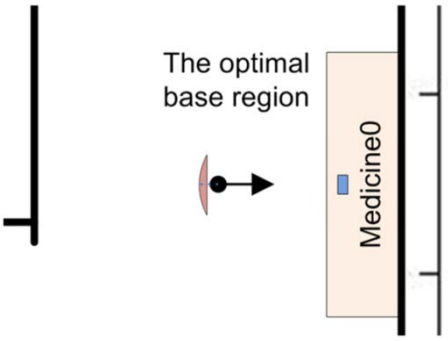


Figure 6.18 The optimal base region and movement trajectory of the robot base (*Medicine0* placed at (8.45*m*, 2.5*m*))

In this diagram, the area in red colour is the optimal base region. The robot moved from the base pose for searching the workspace to the optimal base region.

6.3 Analysis

The analysis of the experiment results is provided in this section. The following criteria are used to evaluate the fuzzy optimisation based symbolic grounding algorithm:

1) The grounded optimal base regions lie within the maximising subset (denoted as $\boldsymbol{A}_{c_1}$ or $\boldsymbol{A}_{R(L)}$) of fuzzy objective function (denoted as $\widetilde{\boldsymbol{A}}$). Robots are expected to be able to find valid arm configurations for grasping target objects when they are placed within the optimal base regions. According to Theorem 5.1, when a robot base is located within the maximising subset ($\boldsymbol{A}_{c_1}$ or $\boldsymbol{A}_{R(L)}$), its reachability value is greater than $R(L)$. According to the definition of FRS in Chapter 3, the robot can find a valid arm configuration for grasping the target object when the reachability value of its base pose is above zero.

2) The constraint value of the robot base poses within the optimal base region (denoted as c_{R^*}) is equal or greater than the constraint value of the optimal base pose (denoted as c_{p^*}). According to the definition of fuzzy constraints in Chapter 4, when $c_{R^*} \geq c_{p^*}$, the obstacles have equal or less influences to the robot base poses within the optimal base region than the optimal base pose. Therefore, the robot can navigate to the optimal base region without colliding with obstacles and a collision-free arm configuration can be found for grasping the target object.

3) The grounded optimal base regions should enable robots to successfully grasp objects with low or high altitudes (refer to Section 3.3 of Chapter 3). When grasping high or low altitude objects, FRSs for grasping these objects are established. The optimal base regions are calculated by performing fuzzy optimisation on the objective functions established based on the FRSs. The optimal base regions become smaller and closer to the target object to make sure the object can be successfully grasped.

4) With the existence of robot base localisation error (denoted as e_l), robots can still successfully grasp target objects. In unstructured environments, robots cannot know their exact locations. When a robot reached the optimal region, the robot base may still lie outside this region due the robot base localisation error. Therefore, in order to successfully grasp target objects, the reachability value of the robot base poses within the optimal base region needs to be greater than $R(L)$, where $R(L)$ is the reachability value of the robot base location set L defined in equation (5.9) in Chapter 5. According to Theorem 5.1, when the reachability value of a robot base pose is greater than $R(L)$, the robot can find a valid arm configuration for grasping the target object when the localisation error is taken into account.

5) The efficiency of the task implementation is improved. As described in the first paragraph of Chapter 5, when a robot is moving to the optimal base region, the robot only need to decide whether its base is located within the region. This will save the time for the robot to exactly position itself to a specific base pose. The efficiency is measured by the time spent for the robot to move from its last base pose (i.e. the robot base pose for searching the workspace) to the optimal base region. The time spent for the robot to move from the base pose for searching the workspace to the specific optimal base pose is denoted as t_{p^*}. The time spent for the robot to move from the base pose for searching the workspace to the optimal base region is denoted ast_{R^*}. The efficiency of task implementation is improved when $t_{p^*} > t_{R^*}$.

In the first experiment, the grounded optimal base region for grasping *MilkBox0* was $R^* = \{(x, y) | x \geq -2.14, \sqrt{(x + 2.95)^2 + (y - 0.2)^2} < 0.84\}$. The maximising subset $\boldsymbol{A}_{c_1}$ of the fuzzy objective function $\widetilde{\boldsymbol{A}}$ was $\boldsymbol{A}_{c_1} = \{(x, y) | 0.76 < \sqrt{(x + 2.95)^2 + (y - 0.2)^2} < 0.84\}$. The optimal base region was within the maximising subset ($R^* \subset \boldsymbol{A}_{c_1}$). The reachablity value $R(L)$ was 0.71. The reachability value of the robot base poses within the optimal base region was 1.0, which was greater than $R(L)$. The robot found a valid arm configuration for grasping *MilkBox0* after it moved to the optimal base region. *MilkBox0* was successfully grasped. The grounded optimal base region meets the first criterion.

The constraint value of the robot base poses within the optimal base region (c_{R^*})was 1.0. The constraint value of the optimal base pose c_{p^*} was 1.0. $c_{R^*} = c_{p^*}$. The robot navigated to the optimal base region without colliding with obstacles and a collision-free arm configuration was found for grasping *MilkBox0*. The grounded optimal base region meets the second criterion.

The reachability value of the robot base poses within the optimal base region was 1.0. The robot base localisation error e_l was up to $0.05m$. The reachability value of the robot base location set $R(L)$ was calculated according to e_l. $R(L) = 0.71$. The reachability value of the optimal base region was greater than $R(L)$. The robot found a valid arm configuration for grasping *MilkBox0* when e_l was taken into account. *MilkBox0* was successfully grasped. The grounded optimal base region meets the forth criterion.

The time spent for the robot to move from the base pose for searching the workspace to the optimal base pose (t_{p^*}) was approximately 30 seconds. The time spent for the robot to move from the base pose for searching the workspace to the optimal base region (t_{R^*}) was approximately 10 seconds. The time spent was reduced by 20 seconds. The efficiency of task implementation was improved. The grounded optimal base region meets the fifth criterion.

In the second experiment, the grounded optimal base region for grasping *MilkBox0* was $R^* = \{(x, y)|y \leq 0.09, \sqrt{(x - 0.65)^2 + (y - 0.79)^2} < 0.73\}$. The maximising subset $\boldsymbol{A}_{c_1}$ of the fuzzy objective function $\widetilde{\boldsymbol{A}}$ was $\boldsymbol{A}_{c_1} = \{(x, y)|0.66 < \sqrt{(x - 0.65)^2 + (y - 0.79)^2} < 0.73\}$. The optimal base region was within the maximising subset ($R^* \subset \boldsymbol{A}_{c_1}$). The robot found a valid arm configuration for grasping *MilkBox0* after it moved to the optimal base region. *MilkBox0* was successfully grasped. The grounded optimal base region meets the first criterion.

The constraint value of the robot base poses within the optimal base region (c_{R^*})was 1.0. The constraint value of the optimal base pose c_{p^*} was 1.0. $c_{R^*} = c_{p^*}$. The robot navigated to the optimal base region without colliding with obstacles and a collision-free arm configuration was found for grasping *MilkBox0*. The grounded optimal base region meets the second criterion.

The optimal base region for grasping *MilkBox0* was $R^* = \{(x, y)|y \leq 0.09, \sqrt{(x-0.65)^2 + (y-0.79)^2} < 0.73\}$. The fuzzy objective function for calculating the optimal base region was established based the FRS for grasping lower altitude objects. A valid arm configuration was found after the robot moved to the optimal base region and *MilkBox0* was successfully grasped. The grounded optimal base region meets the third criterion.

The time spent for the robot to move from the base pose for searching the workspace to the optimal base pose (t_{p^*}) was approximately 30 seconds. The time spent for the robot to move from the base pose for searching the workspace to the optimal base region (t_{R^*}) was approximately 15 seconds. The time spent was reduced by 15 seconds. The efficiency of task implementation was improved. The grounded optimal base region meets the fifth criterion.

In the third experiment, the grounded optimal base region for grasping *Medicine0* was $R^* = \{(x, y)|x \leq 7.64, \sqrt{(x-8.47)^2 + (y-2.5)^2} < 0.84\}$. The maximising subset $\boldsymbol{A}_{c_1}$ of the fuzzy objective function $\widetilde{\boldsymbol{A}}$ was $\boldsymbol{A}_{c_1} = \{(x, y)|0.76 < \sqrt{(x-8.47)^2 + (y-2.5)^2} < 0.84\}$. The optimal base region was within the maximising subset ($R^* \subset \boldsymbol{A}_{c_1}$). The reachablity value $R(L)$ was 0.71. The reachability value of the robot base poses within the optimal base region was 1.0, which was greater than $R(L)$. The robot found a valid arm configuration for grasping *Medicine0* after it moved to the optimal base region. *Medicine0* was successfully grasped. The grounded optimal base region meets the first criterion.

The constraint value of the robot base poses within the optimal base region (c_{R^*})was 1.0. The constraint value of the optimal base pose c_{p^*} was 1.0. $c_{R^*} = c_{p^*}$. The robot navigated to the optimal base region without colliding with obstacles and a collision-free arm configuration was found for grasping *Medicine0*. The grounded optimal base region meets the second criterion.

The reachability value of the robot base poses within the optimal base region was 1.0. The robot base localisation error e_l was up to $0.05m$. The reachability value of

the robot base location set $R(L)$ was calculated according to e_l. $R(L) = 0.71$. The reachability value of the optimal base region was greater than $R(L)$. The robot found a valid arm configuration for grasping *Medicine0* when e_l was taken into account. *Medicine0* was successfully grasped. The grounded optimal base region meets the forth criterion.

The time spent for the robot to move from the base pose for searching the workspace to the optimal base pose (t_{p^*}) was approximately 25 seconds. The time spent for the robot to move from the base pose for searching the workspace to the optimal base region (t_{R^*}) was approximately 5 seconds. The time spent was reduced by 20 seconds. The efficiency of task implementation was improved. The grounded optimal base region meets the fifth criterion.

6.4 Summery

In this chapter, the fuzzy optimisation based symbolic grounding algorithm described in Chapter 5 and the method for calculating robot base poses for searching workspaces are realised as ROS service blocks and integrated with the autonomous control framework. Experiments were carried out where a Care-O-bot 3 robot tried to fetch objects placed at different locations in domestic environments. The analysis of the experiment results indicates that the proposed symbolic grounding algorithm can be used to ground symbolic action sequences into specific robot configurations in terms of robot base poses and the optimal base regions. The grounded robot configurations can support the robot to implement object fetching tasks in domestic environments. The efficiency of task implementation is also improved by using the proposed algorithm.

CHAPTER 7 CONSLUSIONS AND FURTHER WORK

This chapter presents a summary of the important results from the previous chapters and discusses how the essential objectives have been achieved. Finally, other symbolic grounding issues are outlined for further research.

7.1 Conclusions

The conclusions are summarised as follow:

- The optimal robot base region for a robot to grasp a target object can be determined using the proposed fuzzy optimisation based symbolic grounding approach. Through experiments, when the robot was located within the optimal base region, the target object can be successfully grasped.

- The proposed approach solves the problems raised when applying service robots into domestic environments. The problems include: 1) due to the uncertainty of target object position and robot base navigation error, a robot cannot successfully grasp a target object; 2) due to the uncertainty of obstacle spatial information, it takes a long period of time for a robot to reach a specific base pose.

- Experiments proved that the proposed approach do not fully rely on the pervious successful experience, the establishment of probability distribution, the human intervention and the precise visibility of the target object. The computation time is also reduced in compare with other methods.

7.2 Further Work

This work can be furthered in two directions. The first is the development of adaptive fuzzy constraints. When applying service robots into domestic environments, moving obstacles, such as humans are often need to be considered. Moving obstacles may block the movement trajectory of a robot or occupy the grounded optimal base region. To tackle this problem, a method for modelling moving obstacles as adaptive fuzzy constraints may be developed. When determining the optimal base region, the influence of moving obstacles will be taken into account. This will enable a robot successfully implement a given task when humans are walking in the environment. The second is to establish FRSs for implementing other tasks such as surface exploration. New FRSs may be established according to the specific needs of other tasks. The proposed fuzzy optimisation approach can be applied to determine the optimal base regions for implementing the tasks. This will enable a robot to implement more tasks in domestic environments.

7.2.1 Adaptive fuzzy constraints

In the proposed symbolic grounding algorithm, objects in the environment are assumed to be static during the whole process of task implementation. However, in the situation where there is a human walking nearby the grounded optimal base region, this region may no longer suitable for implementing the task since the human may block the movement trajectory of the robot or even occupy the optimal region. An algorithm for dealing with moving obstacles is therefore needed. Based on the constrained fuzzy optimisation algorithm and fuzzy constraints presented in Chapter 4, adaptive fuzzy constraints may be established for modelling the influence of humans in the environment to the determination of the optimal base region. To establish adaptive fuzzy constraints, the movement trajectory of a human must be predicted. Laser scanners equipped on the robot may be used to measure the human's current location, moving direction and moving speed. Based on this information,a preliminary movement trajectory of a

human may be predicted. The adaptive fuzzy constraints may be established based on the predicted human location. For example, the membership function of an adaptive fuzzy constraint may be defined as:

$$c = f(p_{robot}(t), l_{human}(t))$$

where c is the constraint value of a robot base pose, $p_{robot}(t)$ is the robot base pose at time instant t, $l_{human}(t)$ is the human location at time instant t. The constrained fuzzy optimisation algorithm described in Chapter 4 may be applied to calculate the optimal base pose based on the adaptive fuzzy constraints.

7.2.2 Grounding symbolic move base commands for exploring workspaces

Another problem needs to be addressed in the further work is to ground symbolic move robot base commands for exploring workspaces. When a robot is given a task command of fetching objects, it doesn't know where the target object is located. The robot will need to search a workspace (e.g. the surface of a table) where the target object may be placed. Due to the limited range and coverage of scanning equipments, the searching may need to be conducted from several robot base poses. The problem of determining the optimal robot base poses for searching a workspace needs to be addressed. An FRS may be constructed based on the special information (i.e. length, width, height and pose) of a workspace. The reachability value of a robot base pose or region can be deduced based on the size of the area within the workspace that is covered by the laser scanner or camera equipped on the robot. The influences of obstacles to the determination of the optimal robot base poses need to be considered. The constrained fuzzy optimisation algorithm described in Chapter 4 may be applied to calculate the optimal base poses based on the FRS for exploring a workspace.

This approach can be wildly applicable on other applications such as Unmanned Aerial Vehicle (UAV). UVA has the potential to automate the surveillance process or the package delivery. The proposed fuzzy optimisation approach can be used to help an UVA to decide the optimal pose for implementing a given task while handling uncertainties raised in domestic environments.

REFERENCES

Arbeiter, G., Hagele, M., Verl, A., “Field of view dependent registration of point clouds and incremental extraction of table-tops using time-of-flight cameras”, IEEE International Conference on Robotics and Automation (ICRA), pp. 9-13, Shanghai, China, May 2011.

Asfour, T., Azad, P., Vahrenkamp, N., Regenstein, K., Bierbaum, A., Welke, K., Schröder, J. and Dillmann, R., “Toward humanoid manipulation in human-centred environments”, Robotics and Autonomous Systems, vol. 56, issue 1, pp. 54-65, January 2008.

Bandouch, J. and Beetz, M., “Tracking humans interacting with the environment using efficient hierarchical sampling and layered observation models”, IEEE International Workshop on Human-Computer Interaction, pp. 2040-2047, Kyoto, September 2009.

Beetz, M., Mosenlechner, L. and Tenorth, M., “CRAM – A cognitive robot abstract machine for everyday manipulation in human environments”, IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1012-1017, Taipei, 18-22 Oct 2010.

Berenson, D., Kuffner, J. and Choset, H., “An optimization approach to planning for mobile manipulation”, IEEE International Conference on Robotics and Automation, pp. 1187-1192, Pasadena, CA, 19-23 May 2008.

Berenson, D., Diankov R., Nishiwaki, K., Kagami, S. and Kuffner, J., “Grasp planning in complex scenes”, IEEE-RAS International Conference on Humanoid Robots (Humanoids), pp. 42-48, Pittsburgh, PA, 29-01 Nov 2007.

Berenson, D., Srinivasa, S., Ferguson, D., Collet, A. and Kuffner, J., “Manipulation planning with workspace goal regions”, IEEE International Conference on Robotics and Automation, pp. 618-624, Kobe, Japan, 12-17 May 2009.

Bohren, J., Rusu, R. B., Jones, E. G., Marder-Eppstein, E., Pantofaru, C., Wise, M., Mösenlechner, L., Meeussen, W. and Holzer, S., “Towards autonomous robotic butlers: Lessons learned with the PR2”, IEEE International Conference on Robotics and Automation, pp. 5568-5575, Shanghai, China, 9-13 May 2011.

Bowers, D. L. and Lumia, R., “Manipulation of unmodeled objects using intelligent grasping schemes”, IEEE Transactions on Fuzzy Systems, vol. 11, no. 3, June 2003.

Diankov, R., Ratliff, N., Ferguson, D., Srinivasa, S. and Kuffner, J., “Bispace planning: concurrent multi-space exploration,” Robotics: Science and Systems (RSS), June 2008.

Detry, R., Baseski, E., Popovic, M., Touati, Y., Krueger, N., Kroemer, O., Peters, J. and Piater J. H., “Learning object-specific grasp affordance densities”, 8th IEEE International Conference on Develop and Learning, pp. 1-7 Shanghai, China, 5-7 June 2009.

de Granville, C. and Fagg, A. H., “Learning grasp affordances through human demonstration”, International Conference on Development and Learning, pp. 23-29, 2006 [electronically published].

Detry, R. and Piater J. H., “Hierarchical integration of local 3D features for probabilistic pose recovery”, Robot Manipulation: Sensing and Adapting to the Real World (Workshop at Robotics, Science and Systems), 2007.

Detry, R., Pugeault N. and Piater J. H., “Probabilistic pose recovery using learned hierarchical object models”, In International Cognitive Vision Workshop (Workshop at the 6th International Conference on Vision Systems), 2008.

Desouky, S. F. and Schwartz, H. M., “Transformational fuzzy logic controllers for pursuit-evasion differential games”, Robotics and Autonomous Systems, vol. 59, issue 1, pp. 22-23, January 2011.

Efe, M, O. and Kaynak, O., “A novel optimization procedure for training of fuzzy inference systems by combining variable structure systems technique and Levenberg-Marquardt algorithm”, Fuzzy Sets and Systems, vol. 122, pp. 153-165, 2001.

Fedrizzi, A., Moesenlechner, L., Stulp, F. and Beetz, M., “Transformational planning for mobile manipulation based on action-related places”, International Conference on Advanced Robotics (ICAR), pp. 1-8, Munich, 22-26 June 2009.

Fuchs, M., Borst, C., Giordano, P.R., Baumann A., Kraemer, E. and Langwald, J., “Rollin’ Justin - design considerations and realization of a mobile platform for a humanoid upper body”, IEEE International Conference on Robotics and Automation, pp. 4131-4137, Kobe, 2009.

Guan, Y. and Yokoi, K., “Reachable space generation of a humanoid robot using the Monte Carlo method”, IEEE International Conference on Intelligent Robots and Systems (IROS), pp. 1984-1989, Beijing, 2006.

Graf, B., Reiser, U., Hagele, M., Mauz, K. and Klein, P., “Robotic home assistant Care-O-bot 3 - product vision and innovation platform”, IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO), pp. 139-144, Tokyo, Japan, November 23-25 2009.Garcia, M. A., Montiel, O., Castillo, O., Sepulveda, R. and Melin, P., “Path planning for autonomous mobile robot navigation with ant colony optimization and fuzzy cost evaluation”, Applicable Soft Computing, vol. 9, pp. 1102-1110, 2009.

Hsu, D., Latombe, J. C. and Sorkin, S., “Placing a robot manipulator amid obstacles for optimized execution”, IEEE International Symposium on Assembly and Task Planning, pp. 280-285, Porto, 21-24 July 1999.

Hart, S., Ou, S., Sweeney, J. and Grupen, R., “A Framework for learning declarative structure”, RSS Workshop on Manipulation for Human Environments, Philadelphia, PA, August 2006.

Hagras, H. and Sobh, T. “Intelligent learning and control of autonomous robotic agents operating in unstructured environments”, Information Science Journal, Elsevier Science Inc, vol. 145, issue 1-2, pp. 1-12, 2002.

Hagras, H., Callaghan, V. and Collin, M., “Learning and adaptation of an intelligent mobile robot navigator operating in unstructured environment based on a novel online fuzzy-genetic system”, Fuzzy Sets and Systems, vol. 141, pp. 107-160, 2004.

Ji, Z., Qiu, R., Noyvirt, A. E., Soroka, A. J., Packianather, M. S., Setchi, R., Li, D. and Xu, S., “Towards automated task planning for service robots using semantic knowledge

representation", 10th IEEE International Conference on Industrial Informatics (INDIN), pp. 1194-1201, Beijing, China, 25-27 July 2012.

Kira, Z., "Mapping grounded object properties across perceptually heterogeneous embodiments", In 22nd International FLAIRS Conference, 2009.

Klank, U., Zia, M. Z. and Beetz, M., "3D model selection from an internet database for robotic vision", International Conference on Robotics and Automation (ICRA), pp. 2406-2411, Kobe, Japan, 12-17 May 2009.

Koenig, N. and Howard, A., "Design and use paradigms for gazebo, an open-source multi-robot simulator", IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2149-2154, Sendai, Japan, September 2004.

Leven, P. and Hutchinson, S., "Using manipulability to bias sampling during the construction of probabilistic roadmaps", IEEE International Conference on Robotics & Automation, IEEE Transcations, vol. 19, issue 6, pp. 1020-1026, December 2003.

Luck, J., Little, C. and Hoff, W., "Registration of range date using a hybrid simulated annealing and iterative closest point algorithm", IEEE International Conference on Robotics and Automation, vol. 4, pp. 3739-3744, San Francisco, CA, April 2000.

Mavridis, N. and Roy, D., "Grounded situation models for robots: where words and perception meet," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4690-4697, China, 2006.

May, S., Droeschel, D., Holz, D., Wiesen, C. and Fuchs, S., "3D pose estimation and mapping with Time-of-Flight cameras", International IEEE/RSJ Conference on Intelligent Robots and Systems (IROS), 3D Mapping workshop, Nice, France, 2008.

McGuire, P., Fritsch, J., Steil, J., Roethling, F., Fink, G., Wachsmuth, S., Sagerer, G. and Ritter, H., "Multi-modal human-machine communication for instructing robot grasping tasks", IEEE/RSJ International Conference in Intelligent Robots and Systems (IROS), vol. 2, pp. 1082-1088, 2002.

Meeussen, M., Wise, M., Glaser, S., Chitta, S., McGann, C., Mihelich, P., Marder-Eppstein, E., Muja, M., Eruhimov, V., Foote, T., Hsu, J., Rusu, R. B., Marthi, B., Bradski, G., Konolige, K., Gerkey, B. and Berger, E., "Autonomous door opening and plugging in with a personal robot", 2010 IEEE International Conference on Robotics and Automation (ICRA), pp. 729-736, Anchorage, AK, 3-7 May 2010.

Miller, A. T., Knoop, S., Christensen, H. and Allen, P. K., "Automatic grasp planning using shape primitives", IEEE International Conference on Robotics and Automation, 2003, vol. 2, pp. 1824-1829, 2003.

Mitsi, S., Bouzakis, K. D., Sagris, D., Mansour, G., "Determination of optimum robot base location considering discrete end-effector positions by means of hybrid genetic algorithm", Robotics and Computer-integrated Manufacturing, vol. 24, pp. 50-59, 2008.

Montiel, O., Sepulveda, R., Melin, P., Castillo, O., Porta, M. A., Meza, I. M., "Performance of a simple tuned fuzzy controller and a PID controller on a DC motor", 2007 IEEE Symposium on Foundations of Computational Intelligence (FOCI'07), pp. 531-537, Honolulu, HI, , 2007.

Nagatani, K., Hirayama, T., Gofuku, A. and Tanaka, Y., "Motion planning for mobile manipulator with keeping manipulability", IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 2, pp. 1663-1668, 2002.

Nelles, O., Fischer, M. and Muller, B., "Fuzzy rule extraction by a genetic algorithm and constrained nonlinear optimization of membership functions", 5th International Conference on Fuzzy Systems, vol. 1, pp. 213-219, New Orleans, LA, 8-11 September 1996.

Okada, K., Kojima, M., Sagawa, Y., Ichino, T., Sato, K., and Inaba, M., "Vision based behaviour verification system of humanoid robot for daily environment tasks", 6th IEEE-RAS International Conference on Humanoid Robots (Humanoids), pp. 7-12, Genova, 4-6 December 2006.

Ortiz-de-la-Vega, A. H., Gómez-Ramírez, E. and Cortes-Ríos, J. C., “Simple Tuning Algorithm improvements for fuzzy logic controllers”, IEEE International Conference on Fuzzy Systems, pp. 1-8, Barcelona, 18-23 July 2010.

Popovic, M., Kraft, D., Bodenhagen, L., Baseski, E., Pugeault, N., Kragic, D., Asfour, T. and Kruger, N., “A strategy for grasping unknown objects based on co-planarity and colour information”, Robotics and Autonomous Systems, vol. 58, pp. 551-565, 2010.

Qiu, R., Noyvirt, A., Ji, Z., Soroka, A., Li, D., Liu, B., Georg, A., Weisshardt, F. and Xu, S., "Integration of symbolic task planning into operations within an unstructured environment", International Journal of Intelligent Mechatronics and Robotics, vol. 2, issue 2, pp. 128-147, April-June 2012.

Qiu, R., Ji, Z., Noyvirt, A., Soroka, A., Setchi, R., Pham, D. T., Xu, S., Shivarov, N., Pigini, L., Arbeiter, G., Weisshardt, F., Graf, B., Mast, M., Blasi, L., Facal, D., Rooker, M., Lopez, R., Li, D., Liu, B., Kronreif, G. and Smrz, P., “Towards robust personal assistant robots: experience gained in the SRS project”, 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1651-1657, Vilamoura, Portugal, 7-12 October 2012.

Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R. and Ng, A., “ROS: an open-source robot operating system”, International Conference on Robotics and Automation on Open Source Software, vol. 3, issue 3.2, 2009.

Rosenbaum, D. A., “Human movement initiation: specification of arm, direction, and extent”, Journal of Experimental Psychology: General, vol. 109, no. 4, pp. 444-474, 1980.

Rusu, R. B., Blodow, N., Marton, Z. C. and Beetz, M., “Close-range scene segmentation and reconstruction of 3D point cloud maps for mobile manipulation in domestic environments”, 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1-6, St. Louis, USA, 11-15 October 2009.

Rusu, R. B., Marton, Z. C., Blodow, N., Holzbach, A. and Beetz, M., “Model-based and learned semantic object labelling in 3D point cloud maps of kitchen environments”,

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3601-3608, St. Louis, MO, USA, 2009.

Reiser, U., Connette, C., Fischer, J., Kubacki, J., Bubeck, A., Weisshardt, F., Jacobs, T., Parlitz, C., Hagele, M. and Verl, A., "Care-O-bot 3 "Creating a product vision for service robot applications by integrating design and technology", IROS 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1992-1998, St. Louis, MO, 2009.

Roy, D., Hsiao, K. and Maridis, N., "Conversational robots: building blocks for grounding word meanings", HLT-NAACL03 Workshop on Learning Word Meaning from Non-Linguistic Data, vol. 6, pp. 70-77, 2003.

Seraji, H., "Reachability analysis for base placement in mobile manipulators", Journal of Robotic Systems, vol. 12, issue 1, pp. 29-43, 1995.

Srinivasa, S. S., Ferguson, D., Helfrich, C. J., Berenson, D., Collet, A., Diankov, R., Gallagher, G., Hollinger, G., Kuffner, J. and Weghe, M. V., "HERB: a home exploring robotic butler", Auton Robots, vol. 28, Issue 1, pp. 5-20, January 2010.

Stulp, F., Fedrizzi, A. and Beetz, M., "Action-related place-based mobile manipulation", International Conference on Intelligent Robots and Systems (IROS), pp. 3115-3120, St. Louis, MO, 10-15 October, 2009.

Sulzberger, S. M., Tschicholg-Gurman, N. N. and Vestli, S. J., "FUN: optimization of fuzzy rule based systems using neural networks, IEEE Conference on Neural Networks, pp. 312-316, San Francisco, March, 1993.

Stulp, F., Fedrizzi, A., Mosenlechner, L. and Beetz, M., "Learning and reasoning with action-related places for robust mobile manipulation", Journal of Artificial Intelligence Research, vol. 43, pp. 1-42, 2012.

Stulp, F., Fedrizzi, A., Zacharias, F., Tenorth, M., Bandouch, J. and Beetz, M., "Combining analysis, imitation, and experience-based learning to acquire a concept of

reachability", 9th IEEE-RAS International Conference on Humanoid Robots, pp. 161-167, Paris, 7-10 December, 2009.

Saffiotti, A., Ruspini, E. H. and Konolige, K., "Using fuzzy logic for mobile robot control", Chapter 5 of the International Handbook of Fuzzy Sets, Kluwer Academic Publisher, 1999.

Stulp, F. and Beetz, M., "Refining the execution of abstract actions with learned action models", Journal of Artificial Intelligence Research, vol. 32, pp. 487-523, 2008.

Takahama, T., Nagatani, K., Tanaka, Y., "Motion planning for dual-arm mobile manipulator –realization of tidying a room motion", International Conference on Robotics and Automation, pp.4338-4343, New Orleans, LA, April, 2004.

Tenorth, M. and Beetz, M., "Towards practical and grounded knowledge representation systems for autonomous household robots", 1st International Workshop on Cognition for Technical Systems, Germany, 6-8 October, 2008.

Tenorth, M. and Beetz, M., "KnowRob—Knowledge processing for autonomous personal robots", IEEE/RSJ International Conference Intelligent Robots and Systems (IROS), pp. 4261-4266, St.Louis, MO, 10-15 October 2009.

Tegin, J., Ekvall, S., Kragic, D., Iliev, B. and Wikander, J., "Experience based learning and control of robotic grasping," in Workshop of Towards Cognitive Humanoid Robots, 2006.

Tenorth, M., Bandouch, J. and Beetz, M., "The TUM kitchen data set of everyday manipulation activities for motion tracking and action recognition", 2009IEEE 12th Internaitonal Conference on Computer Vision Workshops (ICCV Workshops), pp. 1089-1096, 27-4 October, 2009.

Vahrenkamp, N., Berenson, D., Asfour, T., Kuffner, J. and Dillmann, R., "Humanoid motion planning for dual-arm mannipulation and re-grasping tasks", the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2464-2470, St.Louis, USA, 11-15 October, 2009.

Wachsmuth, S., Brandt-Pook, H., Socher, G., Kummert, F. and Sagerer, G., “Multilevel integration of vision and speech understanding using Bayesian networks”, In H. I. Christensen, editor, Computer Vision Systems: First InternationalConference, pp. 231-254, 1999.

Wang, Z., “Fuzzy set and its application,” Shanghai Science and Technology Press, ISBN 13119.1082, November, 1983.

Zhang, J., Schmidt, R. and Knoll, A., “Appearance-based visual learning in a neuro-fuzzy model for fine-positioning of manipulators”, IEEE International Conference on Robotics and Automation, pp. 1164-1169, Detroit, MI, 10-15 May, 1999.

Zacharias, F., Borst, C. and Hirzinger, G., “Capturing robot workspace structure: representing robot capabilities”, IEEE International Conference on Intelligent Robots and Systems (IROS), pp. 3229-3236, San Diego, CA, 29-2 October 2007.

Zhao, M., Ansari, N. and Hou, E., “Mobile manipulator path planning by a genetic algorithm”, IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 681-688, Raleigh, NC, 7-10 July 1992.

PUBLICATIONS

Liu, B., Li, D., Qiu, R., Yue, Y., Maple, C. and Gu, S., “Fuzzy Optimisation Based Symbolic Grounding for Service Robots”, 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1658-1664, Viamoura, Portugal, 7-12, October 2012 (This paper is one of the finalists for the CoTeSys Best Paper Award)

Liu B., Li, D., Yue, Y., Maple, C., Qiu R. andGu,S., Fuzzy Logic Based Symbolic Grounding for Best Grasp Pose for Homecare Robotics, 2010 10th IEEE International Conference on Industrial Informatics (INDIN), pp. 1164-1169, Beijing, China, 25-27 July 2012

Qiu, R., Noyvirt, A., Ji, Z., Soroka, A., Li, D., Liu, B., Georg, A., Weisshardt, F. and Xu, S., "Integration of symbolic task planning into operations within an unstructured environment", International Journal of Intelligent Mechatronics and Robotics, vol. 2, issue 2, pp. 128-147, April-June 2012

Qiu, R., Ji, Z., Noyvirt, A., Soroka, A., Setchi, R., Pham, D. T., Xu, S., Shivarov, N., Pigini, L., Arbeiter, G., Weisshardt, F., Graf, B., Mast, M., Blasi, L., Facal, D., Rooker, M., Lopez, R., Li, D., Liu, B., Kronreif, G. and Smrz, P., “Towards robust personal assistant robots: experience gained in the SRS project”, 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1651-1657, Vilamoura, Portugal, 7-12 October 2012

Gu, S., Yue, Y., Maple, C., Wu, C. and Liu, B., “Classification of multi-channels SEMG signals using wavelet and neural networks on assistive robot”, 2012 10th IEEE International Conference on Industrial Informatics (INDIN), pp. 1158-1163, Beijing, China, 25-27 July 2012

Gu, S., Yue, Y., Wu, C., Maple, C., Li,D. and Liu, B.,“sEMG based intention identification of human body movement research on assistive robot, Romanian

Academy Journal Technology Science, Vol. 57, No 2-3, pp. 217-233, Bucharest, 2012

Gu, S., Yue, Y., Maple, C., Wu,C. and Liu, B.,“Three-dimensional localisation for wireless sensor networks using probabilistic fuzzy logic, 2013 IEEE 10th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2013), 23-25 July 2013

Gu, S., Yue, Y., Maple, C., Wu,C. and Liu, B., “Challenges in mobile localisation in wireless sensor networks for disaster scenarios”, 2013 IEEE Proceedings of the 19th International Conference on Automation & Computing, Brunel University, Uxbridge, UK, 13-14 September 2013

Gu, S., Wu, C., Yue, Y., Maple, C., Li, D. and Liu, B., “Real-time compliance control of an assistive joint using QNX operating system”, International Journal of Automation & Computing (IJAC) (ISSN 1476-8186), Vol. 10, No. 5, October 2013