THE FLORIDA STATE UNIVERSITY

COLLEGE OF ENGINEERING

AERIAL ROBOT NAVIGATION IN CLUTTERED URBAN ENVIRONMENTS

By

DONGQING SHI

A Dissertation submitted to the Department of Mechanical Engineering in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Degree Awarded:
Summer Semester, 2006
The members of the Committee approve the Dissertation of Dongqing Shi defended on June 20, 2006.

Emmanuel G. Collins, Jr.
Professor Directing Dissertation

Srinivas Palanki
Outside Committee Member

Anjaneyulu Krothapalli
Committee Member

Carl A. Moore
Committee Member

The Office of Graduate Studies has verified and approved the above named committee members.
To Haiyan and Sophia
ACKNOWLEDGEMENTS

Knowledge is in the end based on acknowledgement. –Ludwig Wittgenstein (1889-1951)

I would first like to acknowledge sincere gratitude to my supervisor Dr. Emmanuel G. Collins for his introducing me to the exciting field of artificial intelligence, especially intelligent control for autonomous vehicles. He provided great support and guidance, while ensuring freedom during the whole work. I would like to thank Dr. Carl A. Moore for offering me the chance to pursue Ph.D. in USA and serving on my committee. My sincere thanks also go to Dr. Anjaneyulu Krothapalli and Dr. Srinivas Palanki for accepting to be my committee members and for their valuable comments.

I want to thank my colleagues, Dr. Majura Selekwa, Camilo Ordonez, and Damion Dunlap for their helpful advise during this project. I also thank the administrative staff of the Department of Mechanical Engineering who made my international study smoother.

I wish to acknowledge the financial support of the U.S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD 19-01-2-0012.

Finally, I would like to express my whole-hearted gratitude to my lovely wife and daughter for their selfless love. Also I am deeply indebted to my parents and parents-in-law for their great understanding and support.
# TABLE OF CONTENTS

List of Tables .......................................... vii

List of Figures .......................................... viii

Abstract ..................................................... xii

1. INTRODUCTION ........................................... 1
   1.1 Challenges for 3-D Navigation and Dissertation Contributions ..... 2
   1.2 Assumptions Made in This Research ........................................... 3
   1.3 Robot Control Architectures ..................................................... 4
   1.4 Current Methods for Navigation of Aerial Robots ...................... 7
   1.5 2-D Counterparts of the Proposed Navigation Algorithms .......... 10
   1.6 Helicopter Kinematics and Simulation Environment ............. 13
   1.7 Dissertation Outline ............................................................ 15

2. 3-D NAVIGATION USING FUZZY BEHAVIORS ...................... 18
   2.1 A Possible Method for 3-D Navigation
       Using Fuzzy Behaviors .......................................................... 18
   2.2 The Proposed Approach for 3-D Navigation
       Using Fuzzy Behaviors .......................................................... 20
   2.3 The Fuzzy Speed Controller .................................................... 26
   2.4 Simulation Results ............................................................... 28
   2.5 Conclusions ................................................................. 37

3. 3-D NAVIGATION APPROACH OF VECTOR FIELD HISTOGRAM ........ 39
   3.1 Distinguished Features of 3-D VFH ........................................... 39
   3.2 3-D VFH Algorithm ............................................................... 40
   3.3 Simulations ................................................................. 48
   3.4 Conclusions ................................................................. 54

4. SIMULATION AND EXPERIMENTAL RESULTS ........................ 57
   4.1 Comparisons of Simulation Results
       for the Two Proposed Approaches ........................................... 57
   4.2 Comparison of the Experimental Results
       for the 2-D Counterparts ..................................................... 60
   4.3 Analysis of the 3-D Algorithms and the 2-D Counterparts .......... 63
5. CONCLUSIONS AND FUTURE WORK ........................................ 74
   5.1 Conclusions .......................................................... 74
   5.2 Future Work ......................................................... 75

APPENDICES ................................................................. 76

A. CONVEX/(MAXIMUM CONVEX) FINDING ALGORITHM ................. 76

B. FUZZY RULES FOR THE SPEED CONTROLLER ......................... 78

C. WIDTH CHECKING ....................................................... 79
   C.1 Width Calculation of One Sector .................................. 80
   C.2 Calculation of the Number of the Sectors Corresponding to the Window Filter .................................................. 80

REFERENCES ................................................................. 82

BIOGRAPHICAL SKETCH .................................................... 87
LIST OF TABLES

2.1 Undefuzzified Outputs from the Sub-problems ........................................ 25
2.2 Labelled Outputs from the Sub-problems ........................................ 25
4.1 Comparisons between 3-D fuzzy behavioral algorithm and 3-D VFH algorithm 59
4.2 Comparisons between the 2-D fuzzy behavioral algorithm and the 2-D VFH algorithm ................................................................. 64
B.1 Fuzzy rules for the speed controller ........................................ 78
LIST OF FIGURES

1.1 Classes of Autonomous Robot Control Architectures ................................. 5
1.2 Traditional AI Model ............................................................................. 6
1.3 Behavior-based Model ............................................................................ 7
1.4 The Vector Addition Problem ................................................................. 9
1.5 The Preference-Based Behavioral Control System for the Heading Control System .......................................................... 11
1.6 A Typical Set of Fuzzy Command Preferences for Defuzzification .............. 12
1.7 The Helicopter Coordinate Frames ......................................................... 14
1.8 The Interface in Matlab to Access VRML Worlds ................................... 16
2.1 A Possible Division for the Forward Perception Region ............................. 19
2.2 Decomposition of the 3-D Navigation Problem ......................................... 21
2.3 2-D Sensor Configuration of the Fuzzy Behavioral Controller .................... 22
2.4 Sensor Spectrums .................................................................................... 22
2.5 A Possible Sensor Setup for Obstacle Avoidance .................................... 23
2.6 The General Framework of the 3-D Fuzzy Behavioral Navigation System .......... 23
2.7 Visualization of 3-D Solution Region ...................................................... 24
2.8 Examples of Three Types of Defuzzification Conditions ......................... 25
2.9 Fuzzy Sets for $CS$ .................................................................................. 27
2.10 Fuzzy Sets for $SD$ ................................................................................. 28
2.11 Fuzzy Sets for $\mu_1$, $\mu_2$, $\mu_3$, $\mu_4$, and $\mu_5$ .............................. 29
2.12 Fuzzy Sets for $\Delta v$ ............................................................................. 30
4.13 Experimental Results for Scenario 10 ................................ 72
4.14 Paths in 2-D Motions: the solid line for the path of the VFH, the dot line for the path of the fuzzy behavioral method ........................................ 72
4.15 Rough Division of a Wide Candidate .................................... 73
4.16 Fine Division of a Wide Candidate ........................................... 73
A.1 Single-column Blocks ........................................................... 76
A.2 Multi-column Blocks ............................................................ 77
A.3 Candidate Convex Blocks ....................................................... 77
A.4 Maximum convex Block ........................................................ 77
C.1 Diagram for Width Calculation of One Sector ......................... 79
C.2 Diagram for Calculation of $N_\phi$ ........................................... 81
C.3 Diagram for Calculation of $N_\theta$ .......................................... 81
ABSTRACT

Autonomous navigation systems for mobile robots have been successfully deployed for a wide range of planar ground-based tasks. However, very few counterparts of the previous planar navigation systems were developed for three-dimensional (3-D) motion, which is needed for unmanned aerial vehicles (UAVs). Safe maneuvering in complex environments is a major challenge for UAVs. Future urban reconnaissance and search missions will require UAVs to autonomously navigate through cluttered urban spaces. This research proposes two approaches for unmanned helicopter navigation in cluttered urban environments: a 3-D fuzzy behavioral approach and a 3-D vector field histogram (VFH) approach.

Behavior-based control has been very successful for planar mobile robots navigation in unknown environments. A novel fuzzy behavioral scheme for navigating an unmanned helicopter in cluttered 3-D spaces is developed. The 3-D navigation problem is decomposed into several identical two-dimensional (2-D) navigation sub-problems, each of which is solved by using preference-based fuzzy behaviors. Due to the shortcomings of vector summation during the fusion of the 2-D sub-problems, instead of directly outputting steering sub-directions by their own defuzzification processes, the undefuzzified intermediate results of the sub-problems are fused to a 3-D solution region, representing degrees of preference for the robot movement. A new defuzzification algorithm that steers the robot by finding the centroid of a 3-D convex region of maximum volume in the 3-D solution region is developed. A fuzzy speed control system is also developed to ensure the efficiency and safety of the navigation.

The VFH approach is very popular for planar mobile robots. A 3-D VFH approach to UAV navigation in cluttered urban environments is developed. A 3-D laser measurement system is used to obtain the obstacle distribution in this method. Instead of a 2-D
Cartesian histogram grid as a world model, a 3-D spherical histogram mesh is applied. This 3-D histogram mesh is updated continuously with range data. The 3-D VFH method subsequently employs a two-stage data-reduction process in order to compute the desired control commands for the robot. In the first stage the 3-D histogram mesh is reduced to a 2-D polar histogram corresponding to all possible steering directions for the robot. In the second stage, a novel convex finding algorithm is applied to efficiently find candidate directions from the 2-D polar histogram. The most suitable sector within the candidates with the lowest value of a particular cost function is selected, and the steering of the robot is aligned with that direction.

Substantial simulations have been carried out to demonstrate that the two algorithms proposed in this dissertation can smoothly and effectively guide an unmanned helicopter through unknown and cluttered urban environments. Comparison simulation results show that the 3-D VFH has the ability to travel shorter and smoother paths at most of scenarios. However, the feature doesn’t apply to the 2-D counterparts. The 2-D fuzzy behavioral method usually has a smoother path, but the 2-D VFH travels a shorter path in most of scenarios.
CHAPTER 1
INTRODUCTION

Examine what is said, not him who speaks. – anonymous

For decades unmanned air vehicles (UAVs) have been a hot topic. Many current missions involve urban environment battlefields and this is expected to continue in the future. Accomplishment of such missions requires use of UAVs and unmanned ground vehicles (UGVs). Unmanned air and ground vehicles have actually been in combat in both Afghanistan and Iraq [1, 2, 3]. However, these vehicles have a common problem of lacking sufficient intelligence to act autonomously; normally they are remotely controlled by a human operator who operates them from a ground control center. Because of this, they are subject to human error, especially due to fatigue, which is inescapable under combat environments. In addition, if the environment changes due to either combat or natural activities, the current unmanned vehicles do not react until the remote operator sends a new command; the corresponding delay can be catastrophic. In particular, a robot team in these combat environments cannot cooperate efficiently without autonomous navigation systems. Developing a fully autonomous navigation system for an unmanned vehicle is urgent.

A great number of different techniques has been and are still being developed for efficient obstacle avoidance for UGVs [4, 5, 6, 7, 8]. However, only a few algorithms developed for two-dimensional (2-D) navigation have been extended to three-dimensional (3-D) navigation. Safe maneuvering in complex urban environments is a major challenge for UAVs. The easiest UAV to deploy in these missions is probably a helicopter, which has a significant advantage over traditional fixed-wing aircraft; it can easily perform vertical take-off/landing, hovering, longitudinal/lateral flight, pirouette, and back-to-turn maneuvers. In accordance, unmanned helicopters become the top choice of UAVs. Developing a navigation system that enables an
unmanned helicopter to autonomously fly through complex urban environments is the main objective of this research.

This chapter is organized as follows. Section 1.1 describes the main challenges for 3-D navigation and the contributions made in this dissertation. Section 1.2 presents the assumptions made in the development of the 3-D navigation algorithms in this research. Section 1.3 describes the classes of robot control architectures. Section 1.4 gives some existing methods for aerial robot navigation. Section 1.5 reviews the 2-D counterparts of the two proposed 3-D navigation algorithms. Section 1.6 gives the helicopter kinematics and shows how to create the simulation environment used in this research. Finally, Section 1.7 gives the dissertation outline.

1.1 Challenges for 3-D Navigation and Dissertation Contributions

Navigation is the problem of finding a collision-free path for a robot from one state to another. Developing a 3-D navigation system that enables an aerial robot to autonomously fly through an unknown environment is very difficult. The main challenges are as follows:

1. Requirement to develop novel control schemes for 3-D navigation. There are very few existing paradigms for 3-D navigation of aerial robots, especially non-vision based navigation. The current 2-D navigation structures cannot be used directly. The development of novel control schemes for 3-D navigation is necessary.

2. Unknown/Incomplete knowledge of the environment. The robot usually cannot assume complete environmental knowledge, which is necessary to plan courses of action ahead of time. Often planning and execution should be executed concurrently.

3. Consideration of the vertical dimension. The vertical dimension of aerial robots must be taken into account in contrast to ground vehicles whose width is usually the only dimension considered.

4. Difficulty in mining 3-D range data. Models and software for sensor data integration play a very important role in the navigation system. An efficient analysis of the range data can dramatically improve the performance of the algorithm. The computation
of the desired steering direction from the obtained range data is the essence of the navigation.

In this dissertation, two different 3-D navigation algorithms are developed for UAVs: a 3-D fuzzy behavioral algorithm and a 3-D vector field histogram algorithm (3-D VFH). Their simulation results are compared. The 2-D counterpart of 3-D VFH, employing a Sick laser, has been successfully implemented in a Pioneer II robot. The experimental results are compared with those of the 2-D fuzzy behavioral algorithm of [8], which is essentially a special case of the 3-D fuzzy behavioral algorithm. Summarily, the main contributions of this research are as follows:

1. Developed a novel framework for aerial robots navigation in cluttered urban environments by using preference-based fuzzy behaviors (i.e., a 3-D fuzzy behavioral algorithm). Corresponding functions of sensor data integration and extraction are developed.

2. Developed a 3-D VFH algorithm for aerial robots navigation in cluttered urban environments. The simulation results are compared to those of the 3-D fuzzy behavioral method.

3. Implemented the 2-D counterpart of the 3-D VFH in a Pioneer II robot by using a Sick laser. Experimental results are compared to those of the 2-D fuzzy behavioral algorithm.

1.2 Assumptions Made in This Research

Navigation is the problem of finding collision-free motion for the robot system from one configuration to another. This motion can be a simple path.

Robotic systems, and in particular mobile robotic systems, are the embodiment of a set of complex computational processes, mechanical systems, sensors, actuators, communications hardware and so on. The navigation problem belongs to the computational part, mainly involving high-level software functions, such as path planning and obstacle avoidance. The following assumptions are made in this research.
• An accurate range measurement system is used. The sensor system can scan a 3-D range and is mainly responsible for obtaining the range distance from the robot to the nearest obstacles.

• The robot knows its position and orientation. In particular, localization errors are ignored. The Global Positioning System (GPS) can provide accurate position information. An inertial measurement unit (IMU) can give orientation information.

• Environments to be explored are unknown to the robot, i.e., no global maps are available.

• No disturbance is considered. Any disturbance such as wind, can be adjusted by the flying system apart from the navigation system.

1.3 Robot Control Architectures

A robot control architecture provides a principled way of organizing a robot control system [9]. Usually, the robot control architectures are divided into the following three classes: deliberative also known as planning, reactive, and hybrid (see Fig. 1.1). The following gives the basic idea of each architecture.

1.3.1 Deliberative Systems

Traditionally, in planning systems, given an initial state, a desired goal state and the world map a planner synthesizes a plan, a sequence of actions, that when executed correctly will in principle ensure goal achievement. After the planning phase the problem of action selection and execution is reduced to selecting the next action and following the plan step-by-step in a more or less open loop way. This type of planning is performed off-line (also called a traditional AI system). Fig. 1.2 shows the model of this planning system. The most powerful aspect of these approaches is that they lend themselves to theoretical analysis and verification of properties such as plan correctness, optimality and so forth. Examples of robots that are based on off-line planning approaches are [10, 11]. However, replanning is deemed necessary in unpredictable and dynamic environments. Instead of following the planned path, the path may be replanned when the new sensor data updates the environment model. The key is the replanning algorithm needs to be done
quickly since optimal planning algorithms are usually computationally demanding. Examples that are based on on-line planning approaches are [12, 13, 14].

1.3.2 Reactive Systems

Reactive systems, in contrast to planning to the completion of the task, plan one step ahead at a time. In reactive control there is no planning and reasoning; nor are there world models. Simple reflexes tie action to perceptions, resulting in faster response to outside stimuli. Examples of algorithms that are based on reactive control are Bug algorithms [15], and the VFH algorithm [16].

Behavior-based systems are another class of reactive systems. In these control systems, intelligent action is achieved through coordination of a set of purposive perception-action units, called behaviors. Based on carefully selected sensory information, each behavior produces commands to control the robot with respect to a well-defined aspect of the overall task. However, it is not given how to select a set of appropriate behaviors for a given task. Another major problem is how to coordinate the behaviors to produce a rational next action. While this myopic approach allows generation of timely responses to runtime contingencies,
Fig. 1.3 shows the model of the behavior-based system. For a representative set of behavior-based approaches see [17, 18, 19, 8].

1.3.3 Hybrid Systems

In situations where the world can be accurately modelled, and uncertainty is restricted, deliberative methods are often preferred. However, the conditions favoring purely deliberative planners generally do not exist in the real world. Many researchers feel that hybrid systems capable of incorporating both deliberative reasoning and behavior-based execution are needed. In hybrid architectures, the view, is that it is advantageous to integrate reactive
system components with deliberative planning components that enable long-term planning. For a representative set of hybrid approaches see [20, 21, 22].

1.4 Current Methods for Navigation of Aerial Robots

Sensors considered for obstacle avoidance can be characterized as active or passive sensors. Active sensors are those that make observations by emitting energy into the environment or by modifying the environment [23]. For example, touch or shouting and listening for the echo are active sensing techniques. Active sensors such as millimeter-wave, microwave RADAR, and laser range scanner are currently under investigation for detect, see and avoid (DSA) capability for large UAVs [24]. These sensors exhibit all weather operation with resolution appropriate for wire detection.

Passive sensors, on the other hand, are those that passively receive energy to make their observations [23]. For example, human vision and olfaction involve the use of passive sensors. Passive sensors based on visual electro-optical (EO) or forward looking infrared (FLIR) are of low size, weight and power requirements. They are usually applied to micro air vehicles.

Based on the different sensors for perceiving the surrounding world, there are mainly two types of algorithms: range-sensor based algorithms and vision based algorithms. This
section introduces some existing navigation approaches for UAVs using two different types of sensors.

1.4.1 Range-sensor Based Algorithms

Very few range-sensor based algorithms have been developed. Current range-sensor based navigation systems for planar ground robots cannot be directly applied to aerial robots. In planar navigation systems, the range sensors only view a relatively narrow range in the vertical domain. 3-D navigation systems require the sensor obtain a much wider view in the vertical domain. The sensor data structure for planar navigation systems is no longer suitable for 3-D navigation systems. A novel control architecture is required. Two algorithms for 3-D navigation are given below.

Bug-type algorithm

Bug-type 3-D navigation schemes were developed in [25, 26]. A scheme for 3-D path planning was developed in [25] by combining a 2-D Bug algorithm with a 3-D surface exploration algorithm. This planner required learning the entire shapes of unknown objects based on visual sensors, which is usually difficult. Instead of exploration of the entire obstacle surface, a new bug-type algorithm was developed by using a reduced range data structure [26]. In the latter method, a globally shortest path in simple scenarios was ensured. Moreover, the algorithm generates reasonably short paths even in concave, room-like environments. Both Bug algorithms have shortcomings. In particular, they assume the environments are populated by polyhedral obstacles and do not consider the robot’s size.

Behavior-based Algorithm

A behavioral robotic system for 3-D navigation was introduced in [27], which is essentially an extension of a schema-based reactive navigation system for planar robots. Each schema, a behavior that is instantiated based on the robot’s needs and the environmental conditions, generates a vector reflecting the reaction of the robot to its perceived world. The desired steering velocity is obtained by summation of the schema vectors. In order to deal with incompatible behaviors in some situations, the relative contribution of each schema is determined by a gain factor. The difficulty in choosing suitable schema gains for different scenarios is a limitation of this method. The vector addition during the output coordination
can result in a very poor decision when deciding on a swerve direction to avoid an obstacle [28]. For example, as shown in Fig. 1.4, behavior 1 asks to follow the wall to the right, behavior 2 asks to move the left to avoid the obstacle, and hence the addition of the two behaviors results in hitting the obstacle. Additional disadvantages are that this method assumes the obstacles have a sphere-like shape and the scenarios are relatively simple.

![Figure 1.4: The Vector Addition Problem](image)

1.4.2 Vision-based Algorithms

As stated above, the main advantages of vision sensors are low cost and suitability for micro air vehicles. However, significant image processing is required to detect obstacles using vision sensors. These sensors also require a narrow field of view for wire detection, which largely limits sensor coverage. Additionally, vision sensors are easily affected by variations in light intensity.

Vision-based algorithms are more explored than range-sensor based algorithms. However current UAVs, using vision sensors, still cannot fly through a cluttered urban environment. Vision-based techniques include differential invariants of optical flow [29, 30, 31], multibase-line and omnidirectional stereo [32, 33, 34], structure from motion [35, 36], bearing only
methods [37], obstacle classification [38] and so on. A full review is beyond the scope of this dissertation. However, for illustration two techniques are listed below.

Navigation Using Optical Flow

In [31], a biomimetic reactive navigation system using optical flow was developed. This navigation system is based on the fact that the nervous system of flying insects can exploit visual information to extract the optical flow, which tells them the distance to surrounding obstacles. This algorithm has two major deficiencies: it is computationally intensive and lacks accuracy. In particular, the complexity of the optical flow extraction requires excessive computations, and the variation in light intensity affects the algorithm accuracy.

Navigation Using Stereo

The navigation system in [34] presents the visual threat awareness (VISTA) system for passive, stereo image based obstacle detection for an unmanned air vehicle. The VISTA system combines block matching stereo computed on the Acadia I vision processor designed by Sarnoff Corporation [39] with image segmentation based on a special purpose graph representation appropriate for collision detection, perceptual organization and efficient graph partitioning based on the minimum graph cut. However the algorithm was only tested in environments with sparse obstacles.

1.5 2-D Counterparts of the Proposed Navigation Algorithms

In this section, the corresponding 2-D counterparts of the proposed 3-D navigation algorithms are briefly reviewed. Both proposed algorithms are inspired from their 2-D counterparts. However, none of the proposed navigation problems can be solved directly using the same control architecture as their counterparts. In both proposed algorithms, either the 2-D counterpart is used as a part of the system, or only the basic idea is applied.

1.5.1 2-D Preference-Based Fuzzy Behavioral Navigation System

A primary feature of behavioral control is the ability to respond quickly even in unknown environments. Moreover, fuzzy behavioral methods, which use fuzzy logic controllers, have an inherent ability to handle uncertainty in the robot information [40, 41, 5]. Fuzzy behavioral
methods can be classified in terms of the type of behavioral outputs as standard fuzzy behavioral systems \cite{40, 42} and preference-based fuzzy behavioral systems \cite{43, 5, 44, 45, 6}. Standard fuzzy behavioral systems use behaviors that output a single command, while in contrast preference-based fuzzy behavioral systems use behaviors that output their preferences to various command options. The first of the preference-based behavioral systems were voting architectures like the DAMN architecture \cite{44, 6}. The fuzzy extension of the DAMN architecture was presented in \cite{45}. Independently a similar system using desirability functions and preference logic was proposed in \cite{43} and developed further in \cite{5}, which established a general framework for preference-based systems. A successful application of \cite{43, 5} for navigating in very cluttered environments was presented in \cite{46}. The algorithm of \cite{46} was used to successfully navigate a Pioneer 2 robot through very cluttered environments as detailed in \cite{8}.

The following is a brief review of how to design such a 2-D preference-based fuzzy behavioral controller. Fig. 1.5 shows the structure for the heading control activity.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{figure1.5.png}
\caption{The Preference-Based Behavioral Control System for the Heading Control System}
\end{figure}

The labels in this figure are the five control command alternatives: Large Right Turn (LRT), Slight Right Turn (SRT), No Turn (NT), Slight Left Turn (SLT), and Large Left Turn (LLT). Four behaviors are shown in the above figure: goal seeking, obstacle avoidance, left wall tracking and right wall tracking. Each behavior $i$ assigns a preference $\alpha_{i,j}$ to each command alternative $j$ according to some fuzzy rules. The resultant preference $\alpha_j$ of each
alternative is the minimum of preference cast from all the behaviors to the alternative:
\[
\alpha_j = \bigcap_i \alpha_{i,j}.
\] (1.1)

Each resulting fuzzy \( \alpha_j \) for each command alternative \( j \) is then defuzzified using the standard center of area method into a real number \( \bar{\alpha}_j \in [0, 1] \), which is the measure of the importance of each command alternative \( j \). The final heading command is defuzzified using the centroid of the largest area method as illustrated in Fig. 1.6. The design details can be found in [46, 8].

![Figure 1.6: A Typical Set of Fuzzy Command Preferences for Defuzzification](image)

The main advantage of this architecture is that the command fusion process does not choose a command that is deemed completely unacceptable by one behavior because the behaviors are cooperative. As a fuzzy logic system, it provides an additional aid in the design process in that it allows the coding of the algorithm in words, which is a more natural process for the designer.

### 1.5.2 2-D VFH Algorithm

A great number of different techniques have been and are still being developed for efficient obstacle avoidance for autonomous ground vehicles (AGVs). The vector field histogram method (VFH) [16] is undoubtedly a very successful one. The beginning of VFH can be traced back to the potential field method (PFM) [47]. In PFM a potential field, where the robot is attracted to its goal position and is repulsed away from the obstacles, is built
around the robot. The virtual force field method (VFF) [48] was developed by integrating the concept of potential fields with the concept of histogram grids. VFH was developed in consideration of the inherent limitations of PFM and VFF [49]: trap situations due to local minima, oscillations in the presence of obstacles and oscillations in narrow passages.

VFH has become a very popular obstacle avoidance method due to the reduced possibility of being trapped in local minima and the ability to travel at fast speeds. The enhanced VFH (VFH+) [4] is able to take into account limitations on the turn radius of the vehicle. The vector polar histogram (VPH) method [50] was developed for robots equipped with a laser ranger finder instead of ultrasonic sensors. Recently, the VFH algorithm has been modified and extended to navigation on rough terrain [51]; a grid-type traversability map is built to determine the steering command for the robot. Although the extended algorithm handles 3-D terrain, it cannot be directly used in aerial robots since it outputs only one steering control command (the yaw orientation).

In the VFH approach, a 2-D Cartesian histogram grid is built and updated with range data sampled by sonar. This grid is used as the world model. An obstacle force in the direction of the robot is generated for each cell in the grid that may contain an obstacle. The magnitude of this force is proportional to the certainty value of the cell in the grid and inversely proportional to the distance between the cell and the center of the vehicle. A one-dimensional (1-D) polar histogram is constructed based on the obstacle forces. By applying thresholds, the binary polar histogram is obtained. Candidates are found and filtered by analyzing consecutive sectors in the binary polar histogram. Finally a cost function is used to select the desired steering direction from the candidates.

1.6 Helicopter Kinematics and Simulation Environment

This section presents the helicopter kinematics and simulation environment that are used in the development of the two proposed navigation algorithms.

1.6.1 Helicopter Kinematics

Figure 1.7 shows the helicopter coordinate frames on which the next equations are based. The axes set $XYZ$ denotes the inertial frame, and $xyz$ set denotes the helicopter body fixed frame with origin at the center of mass of the helicopter. Assume that: (a) Frame $XYZ$ and
xyz initially coincide; and (b) No roll movement is involved. Let $\Delta \theta_k$ and $\Delta \phi_k$ respectively represent the pitch angle and yaw angle changes relative to the frame $xyz$ at any instant $k$. The orientation angles $\theta_k$ and $\phi_k$ can be determined by

$$\theta_k = \theta_{k-1} + \Delta \theta_k,$$
$$\phi_k = \phi_{k-1} + \Delta \phi_k.$$ (1.2)

The desired flying direction $\vec{v}_{xyz}$ in Frame $xyz$ can be expressed as

$$\vec{v}_{xyz} = \cos \Delta \phi_k \cos \Delta \theta_k \vec{i} + \cos \Delta \phi_k \sin \Delta \theta_k \vec{j} + \sin \Delta \phi_k \vec{k}.$$ (1.3)

The desired flying direction $\vec{v}_{XYZ}$ in inertial Frame $XYZ$ can be written as

$$\vec{v}_{XYZ} = T \vec{v}_{xyz},$$ (1.4)

where $T$ is the transformation matrix, which is a product of the translation matrix $Trans_k$ and rotation matrix $Rot_k$, i.e.,

$$T = Tran_k \cdot Rot_k.$$ (1.5)

Expressions for $Tran_k$ and $Rot_k$ are

$$Tran_k = \begin{bmatrix} 1 & 0 & 0 & H_X \\ 0 & 1 & 0 & H_Y \\ 0 & 0 & 1 & H_Z \\ 0 & 0 & 0 & 1 \end{bmatrix},$$ (1.6)
\[ Rot_k = Rot(z, \theta) \cdot Rot(y, \phi) \]
\[ = \begin{bmatrix}
\cos \theta & -\sin \theta & 0 & 0 \\
0 & \cos \theta & 0 & 0 \\
\sin \theta & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix} \times 
\begin{bmatrix}
\cos \phi & 0 & \sin \phi & 0 \\
0 & 1 & 0 & 0 \\
-\sin \phi & 0 & \cos \phi & 0 \\
0 & 0 & 0 & 1
\end{bmatrix} \]

where \( H_X, H_Y \) and \( H_Z \) are helicopter translation coordinates relative to Frame \( XYZ \).

### 1.6.2 Simulation Environment

The urban environment and helicopter models are built using the virtual reality modelling language (VRML). Usually there are two ways to build a VRML world. One way is to write VRML code directly in any text editor, for example Notepad in Windows. (Special VRML editors are also commercially available for fast coding.) The second way is to graphically create 3-D models using professional 3-D graphics modelling software such as 3DS max, Pro/E and so on. The models in this research are built using the second method. Section 3.3 in Chapter 2 gives two pictures of the virtual world.

The VRML world is further loaded into the Matlab virtual reality toolbox, which allows the user to access the virtual world in order to test the algorithms. Fig. 1.8 shows the interface in Matlab for loading the VRML world.

### 1.7 Dissertation Outline

The dissertation is organized into five chapters. Each chapter is described briefly below.

In Chapter 2, a 3-D navigation approach using preference-based fuzzy behaviors is proposed. A new framework for 3-D navigation problem is developed. The 3-D navigation problem has been solved by decomposing it into several identical 2-D navigation sub-problems. Each sub-problem uses a particular sensor data extraction model to handle the navigation problem in a small pitch range. The intermediate outputs of the 2-D navigation controllers are fused. A novel 3-D defuzzification algorithm is developed to find the desirable flying orientation for the robot. Moreover, a fuzzy logic based speed controller is applied for
efficient traversal through cluttered environments. The algorithm is simulated in a virtual urban environment.

In Chapter 3, a 3-D VFH approach to navigation of aerial robots is presented. In this method, a 3-D spherical histogram mesh is used as a world model. A two-stage data-reduction process is employed in order to compute the desired control commands for the robot. In the first stage the 3-D histogram mesh is reduced to a 2-D polar histogram corresponding to all possible steering directions for the robot. In the second stage, a novel convex finding and filtering algorithm is developed to find candidates from the 2-D polar histogram. The most suitable sector within the candidates with the lowest value of a particular cost function is selected, and the steering of the robot is aligned with that direction. The algorithm is simulated in a virtual urban environment.

In Chapter 4, simulation results of the two proposed algorithms are compared based on six different scenarios. The 2-D counterpart of the 3-D VFH is implemented in a Pioneer
II robot. The 2-D VFH algorithm is compared with a 2-D fuzzy behavioral algorithm, which had previously been implemented on the Pioneer II, and is a special case of the fuzzy behavioral algorithm developed in this research. Both simulation results and experimental results are analyzed.

Finally, Chapter 5 provides the concluding remarks as well as recommendations for future research.
CHAPTER 2

3-D NAVIGATION USING FUZZY BEHAVIORS

Aerial and underwater applications for behavioral control require reformulation of the existing fuzzy behavioral techniques that have been successfully used for ground-based navigation. The proposed algorithm decomposes the 3-D navigation problem into several 2-D navigation problems, each of which is solved by using preference-based fuzzy behavioral controller detailed in Subsection 1.5.1 of Chapter 1. The final flying orientation is obtained by defuzzying the 3-D solution region that is the fusion of the intermediate outputs of the decomposed 2-D navigation problems. At last, the algorithm was simulated in a virtual urban environment built by virtual reality modeling language (VRML).

This chapter is organized as follows. Section 2.1 shows a possible method for 3-D navigation using fuzzy behaviors. The main problem of this method is its high computational intensity. Section 2.2 presents the proposed algorithm using fuzzy behaviors. Section 2.3 develops a fuzzy logic based speed controller for more efficient transversal. Section 3.3 shows the simulation results in a virtual urban environment. Section 3.4 provides the conclusions.

2.1 A Possible Method for 3-D Navigation Using Fuzzy Behaviors

Inspired by the idea of the existing 2-D fuzzy behavioral techniques, perhaps the easiest way to do 3-D navigation is to clone the basic idea directly. Hence, assume each behavior $i$ casts its preference $\alpha(i,j)$ for each of the available command alternatives $j$. Then, the general fuzzy
rule for each behavior is

\[
\text{IF (Stimuli have particular values)}
\]

\[
\text{THEN (Command 1 scores } \alpha_{(1,j)})
\]

\[
\text{AND(Command 2 scores } \alpha_{(2,j)})
\]

\[
\cdots \text{AND(Command } n \text{ scores } \alpha_{(n,j)}).
\]  \hspace{1cm} (2.1)

Now consider a forward obstacle avoidance behavior. Assume there exists a 3-D range finder which can detect a range, as shown in Fig. 2.1. By equivalently separating the whole range into nine small regions, as shown in Fig. 2.1, the fuzzy rules for this behavior can be written as

\[
\text{IF } (D_1 \text{ AND } D_2 \cdots \text{ AND } D_9)
\]

\[
\text{THEN (Command 1 scores } \alpha_{(1,j)})
\]

\[
\text{AND(Command 2 scores } \alpha_{(2,j)})
\]

\[
\cdots \text{AND(Command } n \text{ scores } \alpha_{(n,j)}).
\]  \hspace{1cm} (2.2)

where the \( D_i \) (\( i \in [1, 2, \ldots, 9] \)) represent the sensor readings indicating the distance from the robot to the nearest obstacle in the corresponding region. If the distance is fuzzified using the classifications short, medium and long, there are \( 3^9(= 19,683) \) total rules. Compared
with the same behavior in the equivalent 2-D system which has only $3^3 (= 27)$ rules, there is a huge increase in the computational cost. This makes it necessary to consider this problem from another perspective.

## 2.2 The Proposed Approach for 3-D Navigation Using Fuzzy Behaviors

Instead of applying the behavioral idea directly as in the previous section, an indirect approach that views the 3-D navigation problem as several 2-D navigation sub-problems at different pitch angles can be considered. The proposed approach is detailed below.

### 2.2.1 2-D Decompositions of the 3-D Navigation Problem

To increase the accuracy of the above approximation of the 3-D navigation problem, a large number of 2-D navigation problem may be used. However, this is computationally expensive. A tradeoff can be achieved by applying a controller that can handle the sensor data in a small pitch range $\theta$, e.g., $\Theta \in \{\theta_o + \Delta\theta : -\Delta\theta \leq \Delta\theta \leq \Delta\bar{\theta}\}$, instead of at a particular pitch angle, say $\theta_o$. In this research, the overall pitch angle range that the sensor can detect is assumed to be from $-52.5^\circ$ to $52.5^\circ$ and the yaw angle range is chosen from $-90^\circ$ to $90^\circ$. $\theta_o$ is one of the angles in the set \{0$^\circ$, ±15$^\circ$, ±30$^\circ$, ±45$^\circ$\}, and $\Delta\bar{\theta}$ is 7.5$^\circ$. Hence seven navigation sub-problems are involved ($52.5^\circ \times 2/15^\circ = 7$).

As shown in Fig. 2.2, the seven sub-problems are labelled as $Sub_{\text{prob}1}$, $Sub_{\text{prob}2}$, $Sub_{\text{prob}3}$, ..., $Sub_{\text{prob}7}$. It is possible to solve each sub-problem using a regular fuzzy, behavioral 2-D controller as stated in Section 1.5.1. Fig. 2.3 shows the sensor configuration for the 2-D controller. The labels are Very Far Left range (VFL), Far Left range (FL), Left range (L), Left Center range (LC), Center range (C), Right Center range (RC), Right range (R), Far Right range (FR), Very Far Right range (VFR). As stated above, the difference is the 2-D controller here is required to deal with sensor data within the small pitch range $\Theta$.

To extract the data to feed as inputs to the 2-D controller, assume the pitch resolution is 5$^\circ$ as shown in Fig. 2.4a. Hence there are four sets of sensor readings within $\Theta$. Then, as shown in Fig. 2.4b, for each yaw range considered the reading for $\Theta$ is the minimum of the four readings in Fig. 2.4a. These compressed data are the inputs to the 2-D controller.

In practice, the above idea may be implemented by a normal 2-D range finder that can spin to reach different pitch angles as shown in Fig. 2.5. However, to increase the sweep...
speed in the pitch direction, it would be best to design a new laser range finder that has the inherent ability to sweep in a 3-D region of space.

### 2.2.2 Fusion of the 2-D Sub-problems

The key task is to fuse the outputs of the seven sub-systems to obtain the final command. Vector summation, which adds all the outputs from the 2-D navigation controllers into the final steering direction, encounters problems when deciding on a swerve direction to avoid an obstacle [28]. In the proposed algorithm, the command fusion problem is solved by preventing the seven sub-systems from performing their own defuzzification processes that output corresponding steering sub-directions. Instead, their undefuzzified results are combined to form a 3-D solution region representing degrees of preference for the UAV movement. The undefuzzified results of each sub-system are the degrees of preference to the five control command alternatives: Large Right Turn (LRT), Slight Right Turn (SRT), No Turn (NT), Slight Left Turn (SLT), and Large Left Turn (LLT). The final steering orientation is defuzzified from this 3-D solution region. This process is called 3-D defuzzification. Fig. 2.6 shows this framework of the 3-D navigation system.

Fig. 2.7 provides an example of a 3-D solution region that uses one dimension for the yaw angle, one dimension for the pitch angle, and the final dimension for the degree of
Figure 2.3: 2-D Sensor Configuration of the Fuzzy Behavioral Controller

(a) Original Sensor Data Spectrum at Each Pitch Angle

(b) Compressed Sensor Data Spectrum for a Sub-problem

Figure 2.4: Sensor Spectrums
preference. A suitable threshold is set to exclude the sub-regions that have relatively low degrees of preference from the overall 3-D solution region. A threshold of 0.2 is chosen in this paper. In the 3-D solution region, each bar with the degree of preference higher than the threshold is called a high preference bar, and all connected high preference bars comprise a candidate. (Two candidates are shown in Fig. 2.7.)

### 2.2.3 A Novel 3-D Defuzzification Algorithm

One possibility for 3-D defuzzification is to find the centroid of the candidate of largest volume. If this method is chosen, as illustrated in Fig. 2.8, there are three general classifications of locations of the centroid’s projection $C'$ in the pitch-yaw plane relative
to the contour of a candidate in the pitch-yaw plane: 1) the projection is inside the contour and it is far away from the contour boundary; 2) the projection is inside the contour but very close to the contour boundary; 3) the projection is outside of the contour. Classifications 2 and 3 indicate that 3-D defuzzification using the centroid of a candidate may lead to an undesirable solution. In practice this occurs when the contour of the candidate is highly concave. The above deficiencies motivate the development of a novel 3-D defuzzification algorithm that ensures that the defuzzified output lies well within the contour of a candidate. The key to the proposed algorithm is to consider all the candidates and find the centroid of a 3-D convex region of maximum volume. Here 3-D convex region actually refers to a 3-D region of a convex contour. The following is the detailed 3-D defuzzification algorithm.

**3-D Defuzzification Algorithm**

1. Apply the threshold to exclude sub-regions with relatively low degrees of preference;

2. Find all the candidate subregions by using the connected components labelling algorithm [52]. To illustrate, Table 2.1 shows the outputs from all the 2-D sub-problems corresponding to Fig. 2.7; the cell value represents the degrees of preference to all the control alternatives. The labelling algorithm highlights two candidates, labelled 1 and 2 respectively, as shown in Table 2.2.

3. A particular maximum convex finding algorithm is developed to ensure the centroid’s projection will be located well inside the contour. For this purpose, the particular
Figure 2.8: Examples of Three Types of Defuzzification Conditions

Table 2.1: Undefuzzified Outputs from the Sub-problems

<table>
<thead>
<tr>
<th>item</th>
<th>preference to LRT</th>
<th>preference to SRT</th>
<th>preference to NT</th>
<th>preference to SLT</th>
<th>preference to LLT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sub_prob1</td>
<td>0.71</td>
<td>0.46</td>
<td>0.40</td>
<td>0</td>
<td>0.86</td>
</tr>
<tr>
<td>Sub_prob2</td>
<td>0.75</td>
<td>0.47</td>
<td>0.41</td>
<td>0</td>
<td>0.85</td>
</tr>
<tr>
<td>Sub_prob3</td>
<td>0.72</td>
<td>0.47</td>
<td>0</td>
<td>0</td>
<td>0.85</td>
</tr>
<tr>
<td>Sub_prob4</td>
<td>0.52</td>
<td>0.47</td>
<td>0</td>
<td>0</td>
<td>0.82</td>
</tr>
<tr>
<td>Sub_prob5</td>
<td>0.51</td>
<td>0.35</td>
<td>0</td>
<td>0.63</td>
<td>0.51</td>
</tr>
<tr>
<td>Sub_prob6</td>
<td>0.53</td>
<td>0.33</td>
<td>0</td>
<td>0.61</td>
<td>0.52</td>
</tr>
<tr>
<td>Sub_prob7</td>
<td>0.44</td>
<td>0.34</td>
<td>0</td>
<td>0.62</td>
<td>0.54</td>
</tr>
</tbody>
</table>

Table 2.2: Labelled Outputs from the Sub-problems

```
1 1 1 0 2
1 1 1 0 2
1 1 0 0 2
1 1 0 0 2
1 1 0 2 2
1 1 0 2 2
1 0 0 2 2
```
algorithm is to find the 3-D convex region of maximum volume from the candidates. The details of this sub-algorithm are presented in Appendix A. (Note that this particular algorithm is different from the general convex decomposition algorithms [53, 54], which split a concave figure into pieces of convex components and each convex component is not overlapped. Besides, the general convex decomposition algorithm cannot guarantee finding a convex region of maximum area.) Once the maximum convex region is found, the final flying orientation at that instant is the yaw and pitch coordinates of the centroid, which can be calculated by the following equations.

\[
\Delta \theta = \frac{\sum V_i \theta_i}{V}, \quad (2.3)
\]

\[
\Delta \phi = \frac{\sum V_i \phi_i}{V}, \quad (2.4)
\]

where \(\Delta \theta\) is the pitch angle change; \(\Delta \phi\) is the yaw angle change; \(V_i\) is the volume of the \(i\)th high preference bar in the maximum convex part; \(V\) is the volume of the maximum convex part; \(\theta_i\) is the average pitch angle for the \(i\)th high preference bar; \(\phi_i\) is the average yaw angle for the \(i\)th high preference bar.

### 2.2.4 Computational Complexity

Compared with the possible 3-D navigation method mentioned in Section 2.1, the computational intensity of the scheme proposed here is small. The total number of rules for the obstacle avoidance behavior with the same sensor configuration and identical fuzzy sets is \(7 \times 3^3 = 189\). Although the computations are more costly than the ground-based counterpart \(3^3 = 27\) rules, it is still a very low-cost methodology for 3-D navigation. An additional benefit is that only one 2-D fuzzy controller is physically required.

### 2.3 The Fuzzy Speed Controller

A speed controller is necessary for efficient and safe navigation. In missions such as urban reconnaissance and search, better stealth and concealment may be provided as well. Such a speed controller is developed using fuzzy logic. The speed controller determines whether the speed should be increased or decreased. A maximum speed of \(20\text{m/s}\) and a minimum speed of \(1\text{m/s}\) are assumed in the development of this controller.
Normally, human-piloted aircraft slow down when they approach an obstacle and speed up when they are moving away from an obstacle. Similarly, the speed change $\Delta v$ is determined by the robot’s Current Speed ($CS$), and the Shortest Distance ($SD$) of the robot to the nearest forward obstacle. The intermediate outputs of the fuzzy system are degrees ($\mu_1, \mu_2, \ldots, \text{and} \mu_5$) to five corresponding commands: Decrease Significantly (DS), Decrease (D), No Change (NC), Increase (I), and Increase Significantly (IS). The general form of the fuzzy rule is

$$\text{IF } (CS \text{ AND } SD) \text{ THEN } (\mu_1 \text{ AND } \mu_2 \text{ AND } \mu_3 \text{ AND } \mu_4 \text{ AND } \mu_5). \quad (2.5)$$

The detailed fuzzy rules are presented in Appendix B. The fuzzy sets of all the normalized variables are illustrated in Figs. 2.10-2.12. The speed command $\Delta v$ is obtained from the degrees of the five commands by the same defuzzification method used in Section 1.5.1. Figure 2.13 illustrates how the speed controller works: the output command is largely positive when $CS = \text{slow}$ and $SD = \text{long}$; the output is largely negative when $CS = \text{fast}$ and $SD = \text{short}$.

![Figure 2.9: Fuzzy Sets for $CS$](image)
Figure 2.10: Fuzzy Sets for SD

2.4 Simulation Results

The proposed method was tested on a virtual unmanned helicopter in a 3-D virtual environment, built using the Virtual Reality Modelling Language (VRML) and simulated in the MATLAB Virtual Reality Toolbox. The urban landscape is approximately \(1600\text{ ft} \times 1230\text{ ft} \approx 500\text{ m} \times 375\text{ m}\). The heights of the buildings vary from \(30\text{ ft} \approx 9\text{ m}\) to \(300\text{ ft} \approx 90\text{ m}\). The helicopter is approximately \(16\text{ ft} \times 1.66\text{ ft} \times 6\text{ ft} \approx 4.9\text{ m} \times 2.0\text{ m} \times 1.8\text{ m}\). The gaps between two neighboring buildings vary from \(30\text{ ft} \approx 9\text{ m}\) to \(60\text{ ft} \approx 18\text{ m}\). Figs. 2.14 and 2.15 show the urban environment used in this research. The sample travelling routes in Figs. 3.14 - 3.25 were plotted based on the positions of the center of mass of the helicopter. In these figures, the distances are all in feet. For each experimental scenario the robot was required to reach an end point from a start point, both of which were specified in Cartesian coordinates. In each scenario the start point and the target were at different altitudes, which required the robot to vary its vertical position.

Scenario 1 (Figs. 3.14 and 3.15) has three very tall and parallel buildings directly in front of the helicopter, which starts on the ground. A target at an altitude of 50 feet is set just behind the buildings. In this case, the robot found that the gap between the buildings was too narrow to pass. Hence, the robot took a path around the buildings. Scenario 2 (Figs. 3.16 and 3.17) has all the settings of Scenario 1 except a slightly increased gap between the
two buildings on the left in Fig. 3.16. The robot agilely passed through the gap and achieved the target. Scenario 3 (Figs. 3.18 and 3.19) represents the same obstacle configuration of Scenario 1 except with a reversal of the positions of the start point and the target. Moreover, the robot was moving away from the target at the beginning, as shown in Fig. 3.18. The goal seeking behaviors commanded the robot to turn 180° along the path for completion of the task.

Another interesting experiment is climbing. As shown in Figs. 3.20 and 3.21 the robot started from the bottom of a tall building, ascended the building in a spiral-like path and reached the target which was 30 feet above the top of the building. The reason that the robot did not simply rise vertically is because the sensor setting, as mentioned in Subsection 2.2.1, was limited to the pitch angular range of $[-52.5^\circ, +52.5^\circ]$.

Scenarios 5 and 6 (Figs. 3.22 - 3.25) show more complex situations that illustrate the integrative capabilities (passing through small gaps, ascending/descending, tracking walls, etc.) to navigate through a densely cluttered urban space. The robot equipped with the proposed navigation system efficiently and safely reached the targets despite the complexity of the environment and the variation of the start positions and the end positions.
Figure 2.12: Fuzzy Sets for $\Delta v$

Figure 2.13: Illustration of the Speed Controller
Figure 2.14: Top View of the Virtual Urban Environment

Figure 2.15: Snapshot within the Virtual Urban Environment

Figure 2.16: Experimental Results for Scenario 1
Figure 2.17: Altitude Trace for Scenario 1

Figure 2.18: Experimental Results for Scenario 2
Figure 2.19: Altitude Trace for Scenario 2

Figure 2.20: Experimental Results for Scenario 3
Figure 2.21: Altitude Trace for Scenario 3

Figure 2.22: Experimental Results for Scenario 4
Figure 2.23: Altitude Trace for Scenario 4

Figure 2.24: Experimental Results for Scenario 5
Figure 2.25: Altitude Trace for Scenario 5

Figure 2.26: Experimental Results for Scenario 6
2.5 Conclusions

A new fuzzy behavioral navigation system for unmanned aerial or underwater vehicles in unknown, cluttered 3-D environments has been proposed. The main contributions are as follows:

1. A novel framework for 3-D navigation problem was developed. The 3-D navigation problem has been solved by decomposing it into several identical 2-D navigation sub-problems. Each sub-problem uses a particular sensor data extraction model to handle the navigation problem in a small pitch range.

2. A new 3-D defuzzification algorithm is developed. After fusing the intermediate outputs of the 2-D navigation controllers, this defuzzification algorithm intelligently finds a desirable and safe flying orientation for the robot.

3. A fuzzy logic based speed controller is developed to enable the robot to efficiently travel through a cluttered space.

The proposed behavioral control algorithm is low cost both in computation and construction. Substantial simulations have been carried out to demonstrate that an unmanned helicopter
is able to reach its goal for a wide variety of obstacle configurations.
CHAPTER 3

3-D NAVIGATION APPROACH OF VECTOR FIELD HISTOGRAM

This chapter outlines a VFH method for 3-D navigation. The approach inherits and extends the ideas of the original VFH. A 3-D spherical histogram mesh is applied as the world model. The 3-D VFH method subsequently employs a two-stage data-reduction process in order to compute the desired control commands for the robot. The algorithm was tested in the same virtual urban environment described in Chapter 2.

This chapter is organized as follows. Section 3.1 highlights distinguished features of the 3-D VFH compared to the original VFH approach. Section 3.2 presents the proposed 3-D VFH. Section 3.3 shows the simulation results in the virtual urban environment. Section 3.4 provides the conclusions.

3.1 Distinguished Features of 3-D VFH

Compared to the original VFH method, the proposed 3-D VFH has the following distinguishing features:

1. Instead of a 2-D Cartesian histogram grid, a 3-D spherical histogram mesh is used as the world model.

2. Instead of sonar, a 3-D laser measurement system is applied as the rangefinder. As a result, in order to construct the polar histogram, only the distance and direction of the obstacle relative to the vehicle is needed as opposed to the calculation of the obstacle force. This data can be directly obtained from the spherical histogram mesh. This feature is also found in the VPH method [50].
3. Both the polar histogram and the binary polar histogram are 2-D rather than 1-D. Since the vehicle is 3-D, rather than 1-D openings, 2-D openings in the polar histogram are identified by a much more complex candidate finding and filtering mechanism. The obtained 2-D openings are called candidates.

4. Classification of the candidates and finding sub-directions from the classified candidates are more complex since the candidates are 2-D.

### 3.2 3-D VFH Algorithm

This section develops a 3-D VFH method for navigation of UAVs. As shown in Fig. 3.1, this method includes a two-stage data-reduction process in order to compute the desired control commands. In the first stage, a 3-D spherical histogram mesh is transformed into a 2-D polar histogram, which is then mapped to a 2-D binary polar histogram using a threshold mechanism for smooth movement and decision making. In the second stage, a set of candidate steering orientations are determined from the binary histogram after the labelling, decomposition and filtering procedures; a cost function is then used to determine the best steering orientation \((\Delta \theta_s, \Delta \phi_s)\). These two stages are detailed below.

![Figure 3.1: Schematic Diagram of the 3-D VFH Algorithm](image)
3.2.1 First Stage Data-reduction

A 3-D laser measurement system that can detect the 3-D world surrounding the robot’s momentary position is assumed in this method. A yaw angle range of 0° to 180° and a pitch angle range of −52.5° to +52.5° are assumed. Because of the high accuracy of the laser system in range finding, a point can be simply represented by spherical coordinates \((r, \theta, \phi)\); moreover, the neighborhood around the point is approximately represented by a spherical mesh as shown in Fig. 3.2: where \(C\) is the center of mass of the robot and the center of the sphere of radius \(r\); \(r\) is also the distance from the center of the robot to the nearest point on an obstacle in the direction defined by the pitch angle \(\theta\) and the yaw angle \(\phi\). The size of each mesh is determined by \(\Delta\phi\) and \(\Delta\theta\). In this paper, it is assumed that \(\Delta\phi = \Delta\theta = 5°\). The obstacle distribution seen by the range sensor is represented the concatenation of each mesh (i.e., the mesh surface). Note that each mesh only represents the portion of the obstacle aligned with the direction determined by the mesh. The sector corresponding to a mesh is defined as shown in Fig. 3.3.

![Figure 3.2: Mesh Definition](image)

During the first data-reduction, the 3-D obstacle distribution is mapped to a 2-D polar histogram, illustrated by Fig. 3.4, which has yaw angle as its first coordinate and pitch angle as its second coordinate. Each bar corresponds to the distance \(r\) to the nearest obstacle in the corresponding sector.

41
A threshold mechanism is further applied to exclude low value sectors from the 2-D polar histogram. To illustrate, the threshold value $T (=10m)$ is applied to generate the 2-D binary histogram of Fig. 3.5 from Fig. 3.4 using the following rules:

$$
\begin{align*}
    r^b &= 1 \text{ if } r \geq T \\
    r^b &= 0 \text{ otherwise}
\end{align*}
$$

(3.1)

A sector of value of $r^b (=1)$ indicates a possible direction for the robot.

Figure 3.4: 2-D Polar Histogram
3.2.2 Second Stage Data-reduction

The second data-reduction computes the desired steering directions ($\Delta \theta_s, \Delta \phi_s$). This subsection explains how to find the candidates and how the steering orientation is computed.

Candidate’s Convex Decomposition and Filtering

In the beginning, it is better to understand why each sector satisfying $r_b = 1$ in the 2-D polar histogram is not directly a steering direction. The reasons are as follows: 1) the opening corresponding to a single sector is not big enough for the robot to fly through as detailed in Appendix C.1; 2) it is not necessary to evaluate every sector satisfying $r_b = 1$ in the polar histogram as detailed below. This discussion motivates the development of the following decomposition and filtering algorithms.

The connected components labelling algorithm [52] is applied to find the connected regions in the 2-D binary polar histogram. Fig. 3.6 illustrates the connected regions of Fig. 3.5; each different gray scale identifies a connected region corresponding to possible steering orientations while the white regions correspond to the directions that are obstructed by obstacles. The connected regions are classified either as a convex region or a concave region.
Each concave connected region is decomposed into a set of convex parts by using the novel convex decomposition algorithm presented in Appendix A. This algorithm is different from the general convex decomposition algorithms that split a concave figure into non-overlapping convex components [53], [54]. In particular, the new algorithm allows the decomposed convex parts to overlap because this provides more potential steering orientations as explained below. For convenience, the convex decompositions of concave regions and convex regions are both called convex parts here.

In order to ensure the openings corresponding to the convex parts are safe enough for the robot to pass through, a window filter is introduced. 2-D VFH [16] inherently uses a 1-D window, although it is not explicitly called a “window.” The window filter has twice the width of the helicopter (corresponding to 4 sectors as shown in Appendix C.2) and is twice as high as the helicopter (also corresponding to 4 sectors as shown in Appendix C.2). As illustrated in Fig. 3.7, each convex part is examined by the window filter. The convex part is selected as a candidate if it is big enough to contain the window filter. This filtering also explains why the overlapped convex parts give more possible orientations: the small convex part 1 in Fig. 3.8a may be excluded by the filter although it is qualified in Fig. 3.8b. Fig. 3.9 shows the decomposition using the new algorithm given in Appendix A. Fig. 3.10 shows the selected candidates in Fig. 3.6 after the convex decomposition and filtering processes; note that region 2 has two candidates, two partly-overlapped convex parts.

Classification of Candidates

Similar to 2-D VFH, two types of the candidates are distinguished, namely, wide and narrow. A candidate is considered wide if more than $W_{\text{max}}$ consecutive windows can be fitted in the yaw range and more than $H_{\text{max}}$ consecutive windows can be fitted in the pitch range as illustrated in Fig. 3.11. In the simulation results of Chapter 4, $W_{\text{max}} = 2$ and $H_{\text{max}} = 2$. A candidate is narrow if only one window filter can be fitted in either the yaw range or in the pitch range as illustrated in Figs. 3.12 and 3.13. A significant advantage of this classification is that instead of evaluating all the sectors within a candidate, only a few of them are considered. For example, in the case of Fig. 3.11, if the goal is located outside of the candidate, one of sectors just inside the candidate’s boundary is more likely to be the best sector. Hence only these sectors need to be evaluated. The following explains how to select the sectors for the two types of candidates.
Figure 3.6: Candidate Orientations

Figure 3.7: Zoom of the Highlighted Rectangle in Fig. 3.6
Figure 3.8: Convex Components Using a General Decomposition Algorithm

Figure 3.9: Convex Components Using the Decomposition Algorithm of Appendix A

For a wide candidate, the sectors to be evaluated, marked “X” in Fig. 3.11, are those located at the center of the window filters that are evenly distributed near the boundary of the candidate. For a narrow candidate, the sectors are determined as in Figs. 3.12 and 3.13. The sector corresponding to the goal orientation is also a sector to be evaluated if it is well located within a candidate, i.e., a window filter that has its center aligned with the goal direction can be fitted in the candidate.
Steering Direction Selection

This subsection presents the cost function used to select the most suitable sector among those obtained in above steps in order to define the new direction of motion. Below, it is assumed that \( c = (\theta_c, \phi_c) \) denotes the orientation of each sector obtained in above steps, \( \alpha_g = (\theta_g, \phi_g) \) denotes the goal orientation and \( \alpha_{n-1} = (\theta_{n-1}, \phi_{n-1}) \) denotes the previous steering orientation.

The cost function is now defined as follows:

\[
g(c, \alpha_g, \alpha_{n-1}) \triangleq k_1 \Delta(c, \alpha_g) + k_2 \Delta(c, \alpha_{n-1}), \tag{3.2}
\]

where

\[
\Delta(c, \alpha_g) \triangleq |\theta_c - \theta_g| + |\phi_c - \phi_g|, \tag{3.3}
\]

\[
\Delta(c, \alpha_{n-1}) \triangleq |\theta_c - \theta_{n-1}| + |\phi_c - \phi_{n-1}|, \tag{3.4}
\]

and the real coefficients \( k_1 \) and \( k_2 \) are positive. The higher the value of \( k_1 \), the more goal-oriented the robot’s behavior, while the higher the value of \( k_2 \), the more the robot tries to execute a smooth path. For a goal-oriented robot, the following relationship between \( k_1 \), and \( k_2 \) is required:

\[
k_1 > k_2 > 0. \tag{3.5}
\]

In the simulations of Section 3.3, \( k_1 = 8 \) and \( k_2 = 6 \). Note that the robot behavior can easily be changed by modifying either the cost function parameters or the cost function itself.
3.3 Simulations

A same fuzzy speed controller is also applied into the robot. The proposed 3-D VFH algorithm was tested on the same helicopter and scenarios. For better understanding, the task of difference scenarios are listed again.

Scenario 1 (Figs. 3.14 and 3.15) has three very tall and parallel buildings directly in front of the helicopter, which starts on the ground. A target at an altitude of 50 feet is set just behind the buildings. In this case, the robot found that the gap between the buildings was
too narrow to pass. Hence, the robot took a path around the buildings. Scenario 2 (Figs. 3.16 and 3.17) has all the settings of Scenario 1 except a slightly increased gap between the two buildings on the left in Fig. 3.16. The robot agilely passed through the gap and achieved the target. Scenario 3 (Figs. 3.18 and 3.19) represents the same obstacle configuration of Scenario 1 except with a reversal of the positions of the start point and the target. Moreover, the robot was moving away from the target at the beginning, as shown in Fig. 3.18. The goal seeking behaviors commanded the robot to turn 180° along the path for completion of the task.

Another interesting experiment is climbing. As shown in Figs. 3.20 and 3.21 the robot started from the bottom of a tall building, ascended the building and reached the target which was 30 feet above the top of the building. The reason that the robot did not simply rise vertically is because the sensor setting, as mentioned in Subsection 3.2.1, was limited to the pitch angular range of $[-52.5^\circ, +52.5^\circ]$.

Scenarios 5 and 6 (Figs. 3.22 - 3.25) show more complex situations that illustrate the integrative capabilities (passing through small gaps, ascending/descending, tracking walls, etc.) to navigate through a densely cluttered urban space. The robot equipped with the
Figure 3.14: Experimental Result for Scenario 1

Figure 3.15: Altitude Trace for Scenario 1
Figure 3.16: Experimental Result for Scenario 2

Figure 3.17: Altitude Trace for Scenario 2
Figure 3.18: Experimental Result for Scenario 3

Figure 3.19: Altitude Trace for Scenario 3
Figure 3.20: Experimental Result for Scenario 4

Figure 3.21: Altitude Trace for Scenario 4
proposed navigation system efficiently and safely reached the targets despite the complexity of the environment and the variation of the start positions and the end positions.

Figure 3.22: Experimental Result for Scenario 5

3.4 Conclusions

A vector field histogram approach for unmanned aerial vehicles navigation in unknown, cluttered 3-D environments has been developed. A 3-D spherical histogram mesh is used as a world model and a 2-D polar histogram is created to find the candidates. A new convex finding algorithm is developed to decompose concave candidates. All the convex candidate or convex parts of concave candidates are evaluated using a window filter to ensure the safety of sub-directions within the candidates. Substantial simulations demonstrate that an unmanned helicopter navigating in an urban environment is able to reach its goal for a wide variety of obstacle configurations.
Figure 3.23: Altitude Trace for Scenario 5

Figure 3.24: Experimental Result for Scenario 6
Figure 3.25: Altitude Trace for Scenario 6
CHAPTER 4

SIMULATION AND EXPERIMENTAL RESULTS

This chapter first compares the simulation results of the previous two chapters. The comparison is based on the six scenarios considered in the previous two chapters. Two criteria are applied for the comparison: the length of the robot’s traveling path (TP) and the bending energy of the path (BE). Moreover, the simplified version of 3-D VFH (called 2-D VFH using Laser) was implemented in Pioneer II robot. The experimental results are compared to those of 2-D fuzzy behavioral algorithm.

The chapter is organized as follows. Section 4.1 presents the comparison of the simulation results of the two proposed algorithms. Section 4.2 gives the experimental results of the 2-D VFH algorithm. The experimental results are also compared to those of the 2-D fuzzy behavioral controller detailed in [8]. Section 4.3 summarizes the comparison results.

4.1 Comparisons of Simulation Results for the Two Proposed Approaches

As stated in Chapters 2 and 3, the two proposed methods were tested on a virtual unmanned helicopter in a 3-D virtual urban environment. The simulation results are compared based on the six scenarios detailed in Chapter 2. For the robot motion, two criteria are used in the comparison: the robot’s travelling path length ($P_L$) and the total bending energy of the path ($B_E$) [40, 6, 55].

The path length of trajectory is the total distance covered by the vehicle from the start location to the goal location. For a 2-D curve sampled by $n$ points in Cartesian coordinates, assume the start location ($x_1, f(x_1)$) and the goal location ($x_n, f(x_n)$), $P_L$ can be calculated by

$$P_L = \sum_{i=1}^{n} \sqrt{(x_{i+1} - x_i)^2 + (f(x_{i+1}) - f(x_i))^2},$$

(4.1)
where \((x_i, f(x_i)), i = 1, 2, \ldots, n\) are the \(n\) trajectory points in Cartesian coordinates.

For a 3-D curve sampled by \(n\) points in Cartesian coordinates, assume the start location \((x_1, y_1, f(x_1, y_1))\) and the goal location \((x_n, y_n, f(x_n, y_n))\), \(P_L\) can be calculated by

\[
P_L = \sum_{i=1}^{n} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2 + (f(x_{i+1}, y_{i+1}) - f(x_i, y_i))^2},
\]

(4.2)

The smoothness of a mobile robot trajectory is a measure of the energy and time requirements for the motion; a smooth trajectory enables energy and time saving because it is easily and readily negotiable. The bending energy \(B_E\), a function of curvature \(k\) is used to evaluate the smoothness of the robot motion. For 2-D curve, the curvature \(k\) at any point \((x_i, f(x_i))\) along the path is given by

\[
k(x_i) = \frac{f''(x_i)}{(1 + (f'(x_i))^2)^{3/2}},
\]

(4.3)

\(B_E\) is then defined by

\[
B_E = \int_a^b k^2(x) \, dx;
\]

(4.4)

numerically,

\[
B_E = \sum_{i=1}^{n} k^2(x_i, f(x_i)).
\]

(4.5)

For 3-D curve with the position vector \(\vec{r}\), the curve can be parameterized using spherical coordinates shown in Fig. 4.1. The computation of the curvature \(k\) is given by [56]

\[
k = \frac{|\dot{\vec{r}} \times \ddot{\vec{r}}|}{|\dot{\vec{r}}|^3},
\]

(4.6)

where [57, 58]

\[
\dot{\vec{r}} = \dot{r} \hat{r} + r \sin(\phi) \dot{\theta} \hat{\theta} + r \dot{\phi} \hat{\phi},
\]

(4.7)

\[
\ddot{\vec{r}} = (\ddot{r} - \dot{r} \dot{\phi}^2 - r \sin^2(\phi) \dot{\theta}^2) \hat{r} + (2 \sin(\phi) \dot{\theta} \dot{\phi} + 2r \cos(\phi) \dot{\phi} + r \sin(\phi) \dot{\theta}) \hat{\theta} + (2 \dot{r} \dot{\phi} + r \ddot{\phi} - r \sin(\phi) \cos(\phi) \dot{\theta}^2) \hat{\phi}.
\]

(4.8)

\(P_L\) and \(B_E\) are calculated based on the above equations. Table 4.1 shows the results from the two proposed algorithms. From this table, it can be seen that the robot with the 3-D VFH travels shorter paths in 5 of 6 scenarios; the bending energies for the 3-D VFH are
Figure 4.1: Spherical Coordinate System

Table 4.1: Comparisons between 3-D fuzzy behavioral algorithm and 3-D VFH algorithm

<table>
<thead>
<tr>
<th>Scenario</th>
<th>$P_L$ (m)</th>
<th>$B_E$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3-D Fuzzy Behavior</td>
<td>3-D VFH</td>
</tr>
<tr>
<td>1</td>
<td>272.66</td>
<td>287.94</td>
</tr>
<tr>
<td>2</td>
<td>195.82</td>
<td>171.57</td>
</tr>
<tr>
<td>3</td>
<td>333.20</td>
<td>288.85</td>
</tr>
<tr>
<td>4</td>
<td>448.60</td>
<td>97.44</td>
</tr>
<tr>
<td>5</td>
<td>476.49</td>
<td>447.10</td>
</tr>
<tr>
<td>6</td>
<td>553.28</td>
<td>482.98</td>
</tr>
</tbody>
</table>

much smaller than those of the 3-D fuzzy behavioral method in all scenarios. Based on these comparisons, the conclusions are: 1) the 3-D VFH provides a smoother motion than the 3-D Fuzzy behavioral method; 2) the 3-D VFH usually enables the robot to travel a shorter path. However, smoother motion does not guarantee motion with a shorter path.

4.2 Comparison of the Experimental Results
for the 2-D Counterparts

Due to the hardware limitations for testing the proposed 3-D algorithms, their 2-D counterparts are implemented and compared. In [8], the algorithm and experiments of 2-D fuzzy behavioral method were detailed. The following presents the implementation of the 2-D VFH and the comparisons.

4.2.1 Test Bed and Scenario Selection

Just like the 2-D fuzzy behavioral method, the 2-D VFH was tested in a Pioneer II robot manufactured by Activ-Media Robotics. The robot, as shown in Fig. 4.2 is a differentially driven platform configured with two drive wheels and one swivel caster for balance. Each wheel is driven independently by a motor with 19.5:1 gear ratio which enables the robot to drive at a maximum speed of 1.2 m/s and climb a 25% grade [59].

![Figure 4.2: The Pioneer II Robot](image)

The laser range finder used as the range sensor is a SICK LMS 200; it has a resolution of 10mm, a typical measurement accuracy of ±15mm, a 180° scanning angle, and 10m typical measured distance range [60]. Measurements can be made for scan angles as small as 0.25° that can be composed into rectangular and cone shaped regions [60].

Localization information was achieved computationally by using the wheel encoders. Each motor on the mobile robotic platform is equipped with a 500 tick encoder [59]. These
measure the change in orientation of the motors in increments of 1/500 of a rotation. This information along with the drive gear ratio provide the change in orientation of each wheel, which is differentiated to provide wheel velocities. The obtained wheel velocities are used in the calculation of the position of the vehicle relative to its initial position, and hence localization is achieved. However, due to the non-ignorable localization error, a camera calibration system [61] is used to obtain accurate robot positions instead of the encoder output data.

Ten experiments were conducted, each mimicking a dense forest in which trees become obstacles to robot motion. These obstacles are very difficult to navigate through because they are relatively small with irregular spacing. Trees were simulated by 2’ long by 2”, 3”, and 4” diameter PVC pipe sections. These pipe diameters scale appropriately to the vehicle size and accurately depict the trunks of trees. Fig. 4.3 shows a snapshot of such a scenario. For a better comparison, the experimental results are shown together with those of the 2-D fuzzy behavioral algorithm in the following subsection.

![Figure 4.3: A Snapshot of the Forest](image)

### 4.2.2 Comparison with 2-D Fuzzy Behavioral Algorithm

The robot start position is (0, 0) in all the scenarios. The Units in the figures are in centimeter cm. The experimental results of the 2-D VFH using a laser are described below. They are also compared with those of 2-D fuzzy behavioral algorithm.
Scenario 1, shown in Fig. 4.4, first requires the robot to pass through a narrow “corridor”. When the robot gets close to the goal, a wall of obstacles keeps the robot from going straight to the goal. The robot follows the wall even though this requires it to temporarily move away from the goal. Finally the robot avoids the wall to reach the goal. The two algorithms produce very similar paths because the path in this scenario is largely determined by the obstacle configuration.

Scenario 2, shown in Fig. 4.5, represents more complex situations that illustrate the ability to navigate very small gaps and even turn away from the goal when necessary to avoid obstacles. The robot can even turn substantially away from the goal after getting very close to it as long as it finds no traversable path directly to the goal. Again, both algorithms produced similar paths due to the restrictions of the obstacle configuration.

Scenarios 3 through 7 shown in Figs. 4.6 through 4.10) show dense forests environments that have multiple paths to the goal. The robot needs to choose the best way to travel through these environments. The algorithms choose fundamentally different path in all these scenarios.

Scenario 8, shown in Fig. 4.11, represents a situation that illustrates the ability to follow a wall of obstacles, then find and pass through a “door”. Both algorithms follow similar path.

Scenario 9, shown in Fig. 4.12, represents an obstacle configuration that allows the robot to choose different directions at the very beginning of the path. The VFH method enables the robot to turn directly to goal even when the local world in the direction of the goal is relatively cluttered but passable. The behavioral method chooses a direction away from the goal to follow a path with smoother motion.

Scenario 10, shown in Fig. 4.13, represents a situation where the goal is aligned with the robot’s initial heading. However, the goal is surrounded by obstacles forming an arc-like shape. The robot demonstrates an ability to avoid the obstacles ahead and find a way around in the arc to reach the goal. Again the algorithms choose two different paths.

Table 4.2 compares the results from the two 2-D counterparts based on the path length and the bending energy. From the table, it can be seen that the 2-D VFH travels shorter paths in 7 of 10 scenarios just like the 3-D case; however the bending energies for the 2-D VFH are slightly larger than those of the 2-D fuzzy behavioral method in 6 Scenarios, which is different than the 3-D case. Another feature shared with the 3-D case is that smoother
motion does not guarantee a shorter path.

### 4.3 Analysis of the 3-D Algorithms and the 2-D Counterparts

The section focuses on the connections and disconnections of the 3-D approaches and their 2-D counterparts based on the observations of the simulation and experiment comparisons.

Both comparisons show that the VFHs travel shorter path in most scenarios. Another words, the VFHs are more goal-oriented simply because the goal direction usually is the steering direction if it is not blocked by obstacles. However, in the fuzzy behavioral approaches the goal seeking behavior always needs to cooperate with other behaviors. This tradeoff usually cannot guide the robot directly to the goal even when the environment is relatively obstacle-free. Moreover, none of the five candidate directions (LLT, SLT, NT, SRT, and LRT) is directly a goal direction.

The main distinction is that the 3-D VFH demonstrates a more smooth path while the
Figure 4.5: Experimental Results for Scenario 2

Table 4.2: Comparisons between the 2-D fuzzy behavioral algorithm and the 2-D VFH algorithm

<table>
<thead>
<tr>
<th>Scenario</th>
<th>$P_L$ (cm)</th>
<th>2-D Fuzzy Behavior</th>
<th>2-D VFH</th>
<th>$B_E$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>570.95</td>
<td>572.75</td>
<td>1.50</td>
<td>18.17</td>
</tr>
<tr>
<td>2</td>
<td>940.89</td>
<td>965.24</td>
<td>54.43</td>
<td>14.07</td>
</tr>
<tr>
<td>3</td>
<td>465.51</td>
<td>455.46</td>
<td>4.08</td>
<td>5.35</td>
</tr>
<tr>
<td>4</td>
<td>432.99</td>
<td>379.72</td>
<td>2.50</td>
<td>2.21</td>
</tr>
<tr>
<td>5</td>
<td>512.72</td>
<td>721.47</td>
<td>1.35</td>
<td>3.90</td>
</tr>
<tr>
<td>6</td>
<td>478.63</td>
<td>425.02</td>
<td>1.22</td>
<td>3.84</td>
</tr>
<tr>
<td>7</td>
<td>538.81</td>
<td>457.59</td>
<td>2.07</td>
<td>4.02</td>
</tr>
<tr>
<td>8</td>
<td>399.69</td>
<td>390.10</td>
<td>6.38</td>
<td>1.16</td>
</tr>
<tr>
<td>9</td>
<td>436.17</td>
<td>236.95</td>
<td>23.62</td>
<td>8.04</td>
</tr>
<tr>
<td>10</td>
<td>444.06</td>
<td>399.89</td>
<td>1.14</td>
<td>2.43</td>
</tr>
</tbody>
</table>
2-D VFH doesn’t. As stated above, the fuzzy behavioral methods are not directly goal-oriented. For 2-D motions, the fuzzy behavioral method usually travels a smoother path as shown in Fig. 4.14. However, this result does not hold for 3-D motion because the robots have many more flying choice directions. Scenario 4 (Figs. 2.22 and 3.20) illustrates this fact. This difference can also be seen from the way the candidates are determined in both VFH approaches. In the 2-D VFH, there are three types of candidates: narrow openings, wide openings and goal direction. For each narrow opening, only the center is the candidate direction. For each wide opening, the left-most direction and the right-most direction are candidates. If these divisions are applied in 3-D VFH, as shown in Fig. 4.15, only four candidates are available in the wide candidate. For the goal mentioned in the above figure, it is better to have the much closer candidate shown in Fig. 4.16 that is mentioned in Subsection 3.2.2. As a result, the rough classification in the 2-D VFH will not work for the 3-D VFH any more. The fine candidate selection algorithm provides the possibility of a
Figure 4.7: Experimental Results for Scenario 4

smoother path.
Figure 4.8: Experimental Results for Scenario 5
Figure 4.9: Experimental Results for Scenario 6
Figure 4.10: Experimental Results for Scenario 7
Figure 4.11: Experimental Results for Scenario 8
Figure 4.12: Experimental Results for Scenario 9
Figure 4.13: Experimental Results for Scenario 10

Figure 4.14: Pathes in 2-D Motions: the solid line for the path of the VFH, the dot line for the path of the fuzzy behavioral method
Figure 4.15: Rough Division of a Wide Candidate

Figure 4.16: Fine Division of a Wide Candidate
CHAPTER 5

CONCLUSIONS AND FUTURE WORK

5.1 Conclusions

Many tasks such as urban search and reconnaissance require UAVs to autonomously fly through an urban environment. Current navigation methods only enable UAVs to travel in simple environments with sparse obstacles. In this research, two different approaches to autonomous navigation were developed. The simulation results demonstrated their effectiveness.

Chapter 1 described the motivation and main challenges, reviewed some current methods for 3-D navigation, including range-sensor based algorithms and vision-based algorithms, reviewed the 2-D counterparts of the algorithms proposed in this research, and described the kinematics model of the helicopter and creation of the virtual urban environment.

Chapter 2 developed a fuzzy behavioral navigation approach to navigation for UAVs. The paradigm decomposed the 3-D navigation problem into several identical 2-D navigation subproblems each of which is solved by a 2-D preference-based fuzzy behavioral controller. The undefuzzified intermediate results of the sub-problems are fused to a 3-D solution region. A novel 3-D defuzzification algorithm that steers the robot by finding the centroid of a 3-D convex region of maximum volume in the 3-D solution region is developed.

Chapter 3 developed a 3-D VFH approach to navigation for UAVs. In this method, a 3-D spherical histogram mesh is applied as the world model. This 3-D histogram mesh is updated continuously with range data. The 3-D VFH method subsequently employs a two-stage data-reduction process in order to compute the desired control commands for the robot. In the first stage the 3-D histogram mesh is reduced to a 2-D polar histogram corresponding to all possible steering directions for the robot. In the second stage, a novel convex finding algorithm is applied to efficiently find candidate directions from the 2-D polar histogram.
The most suitable sector within the candidates with the lowest value of a particular cost function is selected, and the steering of the robot is aligned with that direction.

Chapter 4 compared the simulation results of two proposed algorithms, and compared experimental results of the 2-D counterparts. Based on the comparison results, the connection and disconnection between the 3-D approaches and the 2-D counterparts are analyzed.

5.2 Future Work

In this research, no globally optimal travelling path can be guaranteed in each of the proposed approaches since no planner is used. Essentially, the approaches proposed in this research mainly focus on the obstacle avoidance based on the partial input information (such as from sensors). For achieving optimal objectives such as the shortest path, the smoothest path and so on, a planner that can take into account both the whole world model and the local world model is needed. The local world model is needed for replanning when uncertainty occurs.

Two different terms are usually mentioned: path planner and motion planner (sometimes called trajectory planner). Motion planner, a much wider category, considers both the kinematics and the dynamics of the robot. In other words, velocities and accelerations are also computed when planning. In 3-D spaces, sampling-based methods are widely used for path planning problems. The well-known planners are the probabilistic roadmap planner (PRM) [62], the expansive-space tree planner (EST) [63, 64], and the rapidly-exploring random tree planner (RRT) [65, 66, 67]. The main problem of these planners is that they assume knowledge of the complete 3-D world model (object location, size and shape). The methods proposed in this research are able to deal with the environmental uncertainties. Hence, it should be beneficial to combine them with planners in the future.

Another issue is that the algorithms are currently simulated in Matlab. In the future, it is desirable to test the algorithms on a real helicopter.
APPENDIX A

CONVEX/(MAXIMUM CONVEX) FINDING ALGORITHM

The *convex finding algorithm* is used to find all the convex part. It uses a *divide-and-combine* strategy described in Sub-algorithm 1 to search all possible convex parts. *Divide()* generated all the connected blocks with only one column as shown in Fig. A.1. *Combine()* smartly identifies which single-column blocks can be combined to multi-column blocks as large as possible as shown in Fig. A.2. As a result, the multi-column blocks and some left single-column blocks are selected as the candidate convex blocks shown in Fig. A.3.

Further, the convex block with maximum volume is obtained as shown in Fig. A.4. For identification, it is then called *maximum convex finding algorithm*.

**Sub-algorithm 1 divide-and-combine strategy**

Input. N labelled candidates.
Output. All convex parts of candidates.
1. single-column blocks = *Divide*(labelled candidates).
2. multi-column blocks = *Combine*(single-column blocks).

---

![Figure A.1: Single-column Blocks](image)

76
Figure A.2: Multi-column Blocks

Figure A.3: Candidate Convex Blocks

Figure A.4: Maximum convex Block
APPENDIX B

FUZZY RULES FOR THE SPEED CONTROLLER

The detailed fuzzy rules for the speed controller is presented in Table B.1.

<table>
<thead>
<tr>
<th>inputs</th>
<th>outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS</td>
<td>CD</td>
</tr>
<tr>
<td>slow short</td>
<td>NA</td>
</tr>
<tr>
<td>slow medium</td>
<td>NA</td>
</tr>
<tr>
<td>slow long</td>
<td>NA</td>
</tr>
<tr>
<td>fair short</td>
<td>F</td>
</tr>
<tr>
<td>fair medium</td>
<td>NA</td>
</tr>
<tr>
<td>fair long</td>
<td>NA</td>
</tr>
<tr>
<td>fast short</td>
<td>HF</td>
</tr>
<tr>
<td>fast medium</td>
<td>F</td>
</tr>
<tr>
<td>fast long</td>
<td>NA</td>
</tr>
</tbody>
</table>

Table B.1: Fuzzy rules for the speed controller
APPENDIX C

WIDTH CHECKING

In this paper, the distance threshold $T (= 10m)$ is chosen; the maximum detection range $r_{\text{max}} (= 20m)$ of the laser is assumed. Therefore, the distances of the openings range from 10m to 20m. The helicopter fuselage is approximately $4.9m \times 2.0m \times 1.8m$ ($H_{\text{length}} \times H_{\text{width}} \times H_{\text{height}}$).

Figure C.1: Diagram for Width Calculation of One Sector
C.1 Width Calculation of One Sector

Based on Fig. C.1, the width calculation of the opening corresponding to a particular sector is as follows:

\[
\frac{1}{2} w = r_{\max} \tan \left( \frac{1}{2} \Delta \phi \right)
\]

\[
w = 2 r_{\max} \tan \left( \frac{1}{2} \Delta \phi \right) = 2 \times 20 m \times \tan(2.5^\circ) = 1.7 m < H_{\text{width}}
\]

C.2 Calculation of the Number of the Sectors Corresponding to the Window Filter

In order to have safe margins, the minimum distance \( T \) to the openings is used. Based on Fig. C.2, the number of sectors in the yaw range \( N_\phi \) is obtained as follows:

\[
\frac{1}{2} \Delta \phi_f = \arctan \left( \frac{H_{\text{width}}}{T} \right)
\]

\[
\Delta \phi_f = 22.6^\circ
\]

\[
N_\phi = (\text{int}) \left( \frac{\Delta \phi_f}{\Delta \phi} \right) = (\text{int}) \left( \frac{22^\circ}{5^\circ} \right) = 4
\]

Similarly, using Fig. C.3, the number of sectors in the pitch range is given by \( N_\theta = 4 \).
Figure C.2: Diagram for Calculation of $N_{\phi}$

Figure C.3: Diagram for Calculation of $N_{\theta}$
REFERENCES


[34] Jeffrey Byrne, Martin Cosgrove, and Raman Mehra. Stereo based obstacle detection for an unmanned air vehicle. In *IEEE International Conference on Robotics and Automation*, pages 2830–2835, Orlando, May 2006. 1.4.2, 1.4.2


84


86
DONGQING SHI

Dongqing Shi was born on June 25, 1977, in Zhejiang, China. In June 1999, he completed his Bachelor’s degree in Mechatronics at Harbin Engineering University. From 1999 to 2002, he worked at the national key lab of fluid power transmission and control at Zhejiang University. Under the advisement of Prof. Ying Chen, he obtained his Master’s degree in spring of 2002, from the Department of Mechanical Engineering at Zhejiang University. He enrolled in the doctoral program at the Florida State University in the spring of 2003. From 2004, he worked at the center of intelligent systems, control and robotics (CISCOR) with Dr. Emmanuel Collins who is the director of the CISCOR.

His research interests include dynamic modelling, control in manufacturing, intelligent control systems for autonomous vehicles, computer graphics, and scientific visualization.