1-1-2016

Autonomous Collision Avoidance for a Teleoperated UAV Based on a Super-Ellipsoidal Potential Function

Mohammed Salim Qasim
University of Denver

Autonomous Collision Avoidance for a Teleoperated UAV Based on a Super-Ellipsoidal Potential Function

Abstract

This thesis presents the design of a super-ellipsoidal potential function (SEPF) that can be used, in a static and dynamic environment, for autonomous collision avoidance of an unmanned aerial vehicle (UAV) in a 3-dimensional space. In the design of the SEPF, we have the full control over the shape and size of the potential function. In our proposed approach, a teleoperated UAV can not only autonomously avoid collision with surrounding objects but also track the operator' control input as closely as possible. As a result, an operator can always be in control of the UAV for his/her high-level guidance and navigation task without worrying too much about the UAV collision avoidance while it is being teleoperated. The effectiveness of the proposed approach is demonstrated through a human-in-the-loop simulation using virtual robot experimentation platform (v-rep) and Matlab programs and experimentation using a physical quadrotor UAV in a laboratory environment.

Document Type

Thesis

Degree Name

M.S.

Department

Mechatronics Systems Engineering

First Advisor

Kyoung-Dae Kim, Ph.D.

Second Advisor

Ali Besharat

Third Advisor

Kimon Valavanis

Keywords

Artificial potential function, Autonomy, Collision avoidance, Quadrotor, Teleoperation, Unmanned aerial vehicle

Subject Categories

Electrical and Computer Engineering

Autonomous Collision Avoidance for a Teleoperated UAV Based on a Super-ellipsoidal Potential Function

A Thesis

Presented to

the Faculty of the Daniel Felix Ritchie School of Engineering

and Computer Science

University of Denver

in Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Mohammed S. Qasim

August 2016

Advisor: Kyoung-Dae Kim

Author: Mohammed S. Qasim
Title: Autonomous Collision Avoidance for a Teleoperated UAV Based on a Super-ellipsoidal Potential Function
Advisor: Kyoung-Dae Kim
Degree Date: August 2016

Abstract

This thesis presents the design of a super-ellipsoidal potential function (SEPF) that can be used, in a static and dynamic environment, for autonomous collision avoidance of an unmanned aerial vehicle (UAV) in a 3-dimensional space. In the design of the SEPF, we have the full control over the shape and size of the potential function. In our proposed approach, a teleoperated UAV can not only autonomously avoid collision with surrounding objects but also track the operator's control input as closely as possible. As a result, an operator can always be in control of the UAV for his/her high-level guidance and navigation task without worrying too much about the UAV collision avoidance while it is being teleoperated. The effectiveness of the proposed approach is demonstrated through a human-in-the-loop simulation using virtual robot experimentation platform (v-rep) and Matlab programs and experimentation using a physical quadrotor UAV in a laboratory environment.

Acknowledgements

I would like to thank my sponsor, the Higher Committee for Education Development (HCED) in Iraq, for giving me the opportunity to study in the united states and supporting financially me throughout my master study. I would also like to thank my family for their motivation and support during my study and my advisor, Dr. kyoung-Dae Kim, for his invaluable advices and guidance during this research.

Table of Contents

List of Figures

Abbreviations

rviz 3D visualization tool for ROS.

API Application Program Interface.

ROS Robot Operating System.

SEPF Super-Ellipsoidal Potential Function.

UAV Unmanned Aerial Vehicles.

v-rep virtual robot experimentation program.

VTOL Vertical Take-off and Landing.

FOV Field of View.

TTI Time to Impact.

DPF Dynamic Parametric Field.

VS Virtual Spring.

SLAM Simultaneous Localization and Mapping.

TTC Time to Collision.

Nomenclature

$\mathbf{V}_{ca}$ collision avoidance motion vector.

R_s The range of the 3D range sensor.

d_{min} The minimum stopping distance.

a_{max} The maximum deceleration.

n The amount of flattening at the SEPF tips.

R_U The radius of the smallest circle that encircles the UAV.

d_s The minimum safety distance between the UAV and the obstacle.

a_{fo} The length of the outer half super-ellipse in the front direction of the SEPF.

b_{fo} The width of the outer half super-ellipse in the front direction of the SEPF.

c_{fo} The height of the outer half super-ellipse in the front direction of the SEPF.

a_{bo} The length of the outer half super-ellipse in the back direction of the SEPF.

b_{bo} The width of the outer half super-ellipse in the back direction of the SEPF.

c_{bo} The height of the outer half super-ellipse in the back direction of the SEPF.

a_{fi} The length of the inner half super-ellipse in the front direction of the SEPF.

b_{fi} The width of the inner half super-ellipse in the front direction of the SEPF.

c_{fi} The height of the inner half super-ellipse in the front direction of the SEPF.

a_{bi} The length of the inner half super-ellipse in the back direction of the SEPF.

b_{bi} The width of the inner half super-ellipse in the back direction of the SEPF.

c_{bi} The height of the inner half super-ellipse in the back direction of the SEPF.

o a point that lies in between the inner and the putter super-ellipsoids of the SEPF.

$\mathbf{p}_{ro}$ The distance vector from the center of the UAV to the point o.

$\mathbf{p}_{or}$ The distance vector from the point o to the center of the UAV.

$\mathbf{P}_r$ The current robot position.

$\mathbf{P}_o$ The position of the point o, i.e, an obstacle.

$\mathbf{v}_r$ The current robot velocity.

$\mathbf{v}_o$ The velocity of the point o, i.e, an obstacle.

d_i The distance from the point o to the inner super-ellipsoid along the direction of $\mathbf{p}_{ro}$.

d_o The distance from the inner super-ellipsoid to the outer super-ellipsoid along the direction of $\mathbf{p}_{ro}$.

$\mathbf{P}_{\mathbf{S}rep}$ The repulsive SEPF for static environments.

$\mathbf{P}_{\mathbf{D}rep}$ The repulsive SEPF for dynamic environments.

μ Design parameter that is used to scale the repulsive SEPF.

R_i The distances from the center of the UAV to the inner super-ellipsoid of the SEPF.

R_o The distances from the center of the UAV to the outer super-ellipsoid of the SEPF.

$\mathbf{V}_{Sca}$ Collision avoidance motion vector resulting from static obstacles.

$\mathbf{V}_{Dca}$ Collision avoidance motion vector resulting from dynamic obstacles.

$\mathbf{A}_{Sca}$ Collision avoidance angle vector resulting from static obstacles.

$\mathbf{A}_{Dca}$ Collision avoidance angle vector resulting from dynamic obstacles.

$\mathbf{v_{ro}}$ The relative velocity between the robot and point o in the direction from the robot to point o.

$\mathbf{n}_{ro}$ A unit vector directing from the robot to the point o.

$\mathbf{n}_{or}$ A unit vector directing fromthe point o to the robot.

v_{rel} The relative velocity between the robot and point o.

$\mathbf{V}_{ref}$ The reference velocity for the UAV.

$\mathbf{V}_{in}$ The operator's command input velocity.

$\mathbf{A}_{ref}$ The reference angle for the UAV.

$\mathbf{A}_{in}$ The operator's command input angle.

∇ Gradient operator.

Chapter 1

Introduction

1.1 Motivation

Recently, due to various advantages in terms of their size, cost, weight, and, more importantly, versatile mobility such as hovering, vertical take-off and landing (VTOL), omnidirectional agile movement, etc., quadrotors UAVs have gained a lot of attention from scientists and have been used successfully in many tasks such as search and rescue, remote sensing, mapping, exploration, surveillance and many other civil and military applications [5], [6], [7], [8]. However, nowadays robots autonomy is still restricted by the deficiency of a robust and reliable perception, and of higher cognitive abilities that permit sophisticated decision making in a real world environment [9]. Thus, human supervisory is required to perform high-level decision making while the robots execute their local autonomy such as obstacle avoidance.

In teleoperation, the operator is physically separated from the robot. This leads to a difficult teleoperation process due to poor situation awareness [10]. One main way to transfer the information to the operator is through a camera mounted on the UAV. This visual information is often restricted due to limited camera resolution and field of view (FOV) [11]. In the case of quadrotor UAVs, the teleoperation is usually non-trivial due to its inherent nonlinear underactuated dynamics, fast and agile omnidirectional mobility, etc. If a quadrotor UAV is not controlled carefully, it can easily collide with obstacles, especially in cluttered indoor environments. And therefore it requires a high level of expertise as well as enough training to teleoperate a quadrotor UAV safely. This is our main motivation to develop an algorithm that assists the operator by making the robot avoid collision autonomously.

1.2 Thesis Objectives

The main goal of this thesis is to address the problems discussed in the motivation section by developing an autonomous collision avoidance algorithm that aids the operator in teleoperation tasks. The proposed algorithm aims to make the teleoperation easy and safe by not only preventing the collision with obstacles but also tracking operator's command, if there is one, in the collision free path so that the operator can focus on the main task rather than avoiding collision with surrounding objects. As a result, the teleoperation efficiency is improved as well.

1.3 Thesis Outline

This thesis is organized as follows.

Chapter 2 presents the literature review in which we introduce the unmanned aerial vehicles (UAVs) with the focus on quadrotor UAVs. We also give a brief information about the teleoperation of robots. In addition, we review the potential function based motion planning. Finally, we present the related work.

Chapter 3 presents the design of the SEPF in a 2-dimensional space and the extension of the design to a 3-dimensional space. The SEPF is then extended to include the dynamic obstacles case. Collision Avoidance under Teleoperation is finally explained.

Chapter 4 presents the validation of the proposed method through a human-in-the-loop simulation. The simulation programs and setup are explained. And the simulation results for the teleoperation of a quadrotor UAV in a static and dynamic environment are demonstrated and discussed.

Chapter 5 presents a set of human-in-the-loop experiments of a physical quadrotor UAV teleoperation in order to further verify the proposed approach. The implementation details are also explained. The results of the experiments are finally presented and discussed.

Chapter 6 presents the teleoperation interface program. The tools used to develop the user interface are first explained. Then, two methods for visualizing the robot's surrounding environment are discussed and demonstrated through simulation.

Chapter 7 presents the conclusion of the work presented in this thesis and provides recommendations for future research.

We make note that some parts of this thesis are reprinted from our published paper [12] where we presented the design of the SEPF and simulation results for the static obstacles case. After this paper, we extended the SEPF to include the dynamic obstacles case and validated it through simulation. Finally, we validated our algorithm, for both static and dynamic cases, with a physical quadrotor UAV in a laboratory environment.

Chapter 2

Literature Review

2.1 Unmanned Aerial Vehicles

Unmanned aerial vehicles (UAVs) are generally classified into two main types: fixed wing UAVs such as an airplane and rotary wing UAVs such as a helicopter. For other classifications schemes, see [13], [14]. Fixed wing UAVs are usually faster than rotary wing UAVs but with less maneuverability. Furthermore, they can fly for long distances. Rotary wing UAVs have the ability of vertical take off and landing (VTOL) and hovering which fixed wing UAVs do not have. They also have high maneuverability. In this thesis, a rotary wing UAV is used. More specifically, a quadrotor UAV is used which has four rotors because of its unique features as illustrated in the next section.

2.1.1 Quadrotor UAVs

Quadrotors have four propellers powered by electric motors mounted on a frame shaped like 'X'. The schematic of a quadrotor UAV is shown in Fig. 2.1. Each rotor

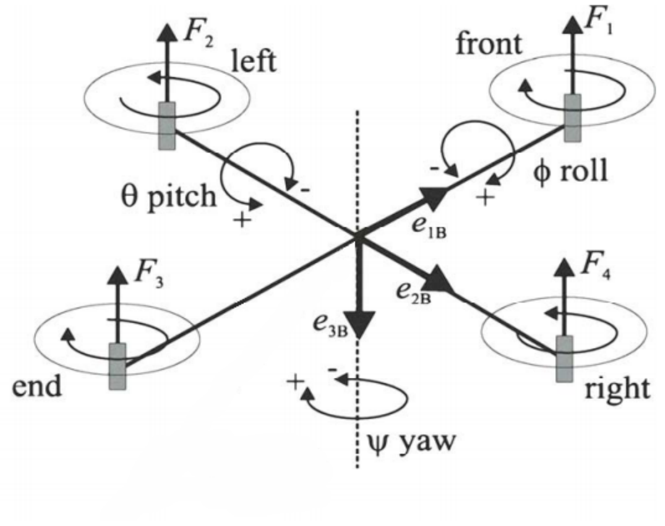


Figure 2.1: Qaudrotor schematic [1].

produces a force (e.g, F_1, F_2, F_3 and F_4 in Fig. 2.1) and a torque about the vehicle's body z-axis (e_{3B} in Fig. 2.1). To make the net torque about the body z-axis equal to zero when all the propellers are spinning, adjacent propellers spin opposite to each other.

By adjusting rotor forces, i.e., adjusting the spinning speed of the motors, different movements can be achieved. We can control the thrust for translation along the body z-axis, roll angle for translation along the body y-axis, pitch angle for translation along the body x-axis and yaw for rotation about the body z-axis.

Quadrotors have many attracting features such as small weight and size, low cost, omnidirectional agile movement, hovering and VTOL. They have been used in many applications such as filming, surveillance, delivering packages, etc [15]. Despite these features, the control of quadrotors is still a challenge because of its inherent underactuated nonlinear dynamics.

2.2 Teleoperation

Teleoperation can be defined as the operation of a device/machine at or over a distance where the term tele means at or over a distance. Teleoperation scheme is shown in Fig. 2.2. As can be seen from the figure, there are two sites: the operator site where the operator commands the remote robot through a haptic device or joy-stick, etc. and receives the information about the robot surrounding environment via a visual display and sometimes a haptic device, and the remote site where the robot receives and implements operator's commands with the help of its sensors and control system.

Human intervention makes the teleoperation process very beneficial as the human intelligence can be exploited to perform high-level planning and decision making. Hence, teleoperation has numerous applications such as dealing with hazardous materials, space and underwater exploration, surveillance, etc. [16], [17]. Control architectures used in teleoperation are:

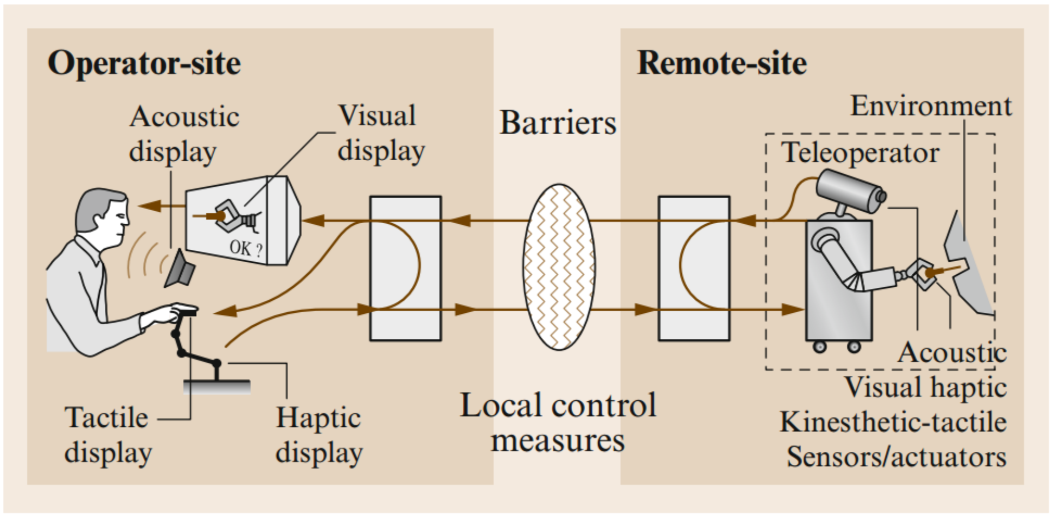


Figure 2.2: Teleoperation scheme [2].

- **Direct Control:** A human operator directly controls the robot without any level of autonomy on the robot side, remote site. This method has the higher operator workload.

- **Shared Control:** A human operator controls the robot with some level of autonomy on the robot side to assist the human operator. This method has a moderate operator workload.

- **Supervisory Control:** A human operator controls the robot with a high level of autonomy and intelligence on the robot side to assist the human operator. This method has the least operator workload.

In this thesis, the second control architecture is used. The human operator is always in control of the UAV while the UAV has some level of autonomy, i.e., autonomous collision avoidance.

2.3 Potential Function Based Motion Planning

The potential function approach is very common in autonomous path planning for robots because of its mathematical elegance and simplicity. It was first introduced by Khatib [18] and used for collision avoidance of a robot arm with an object based on the relative distance between them.

A potential function or potential field is defined as a differentiable real-valued function $P : \mathbb{R}^m \to \mathbb{R}$ [19]. In the potential function method, the robot's workspace is filled with an artificial potential field. This field is a sum of two potential fields: an attractive potential field that steers the robot towards the goal and a repulsive potential field that repels the robot away from the obstacles.

The gradient of the potential function represents a driving force ($\mathbf{F}$) that drives the robot towards the goal while avoiding obstacles. The attractive potential is a function of the relative distance to the goal and the repulsive potential is a function of the relative distance to the obstacles. When the robot starts to move, the force vector field $\mathbf{F}$ is at its maximum and gradually diminishes to zero as the robot approaches the goal. Thus, the robot gradually slows down and stops at the goal. The potential function is defined as follows

$$U(q) = U_{att}(q) + U_{rep}(q) \tag{2.3.1}$$

where q is the robot position. The attractive and repulsive forces vectors are defined as the negative gradient of attractive and repulsive potentials respectively

$$F_{att} = -\nabla U_{att}(q) \tag{2.3.2}$$

$$F_{rep} = -\nabla U_{rep}(q) \tag{2.3.3}$$

then the reference force is the sum of the attractive and repulsive forces

$$F_{ref} = F_{att} + F_{rep} \tag{2.3.4}$$

One common problem with the potential function method is the local minima problem where the vector field $\mathbf{F}$ approaches zero in positions other than the goal position because of the presence of the obstacles. This local minima may trap the robot before reaching the goal. Research on this area led to several solutions, e.g., [20], [21]. In teleoperation, if the robot gets trapped by a local minima, the human operator can easily escape it. Thus, the local minima problem is less of concern in teleoperation.

The potential function method can be used for path planning in static and dynamic environments and for static or moving goal [22].

2.4 Related Work

Collision avoidance is one of the essential tasks for mobile robots. Therefore, collision avoidance has been widely studied in the literature. Potential function based methods [10], [23], [24], [25] are developed for collision avoidance in a UAV teleoperation. In [26], [27], [28], [29], teleoperation with force feedback as a cue for the operator has been studied and applied to mobile robots to navigate in a cluttered environment. Potential function for collision avoidance in a group of UAVs teleoperation has been studied and implemented in [30], [31], [32].

Our main interest in this thesis is the collision avoidance in UAV teleoperation. The research in this area is limited compared to collision avoidance in ground robots. Next, we present some of the research in UAV teleoperation *in a static environment*.

Brandt et al. [33] used a haptic feedback to assist the operator in avoiding collision with static obstacles. The haptic feedback aims to increase the operator's situation awareness as the information about the environment from the camera mounted on the UAV is often limited. In this study, the amount of force feedback is determined using three different algorithms. The first algorithm uses the time to impact (TTI) which is the relative distance to the nearest obstacle divided by the robot's current velocity. The amount of feedback force is inversely proportional to TTI.

The second algorithm uses a so-called dynamic parametric field (DPF) where the distance to the obstacle is divided into four zones: safe zone, warning zone, transition zone and collision zone. These zones are dynamic and determine the amount of force feedback. The last algorithm is the virtual spring (VS) in which the amount of force feedback is determined based on the distance to the obstacle. This research found that the best algorithm in avoiding collision is the TTI algorithm.

Mendes et al. [34] used FastSLAM (Fast Simultaneous Localization and mapping) to map the environment and calculate the relative distance to the obstacles. The current robot's velocity and the relative distance to the obstacles are used to estimate the time to collision (TTC). TTC is classified into threat levels and the response, i.e., no action; slow; stop and evasive maneuver, is taken accordingly to override the operator control input and slow, stop, or move the robot oppositely to its current direction.

Israelsen et al. [15] used a different approach from the previous two methods; they took the actual dynamics, states, and the operator command input into account to estimate the future trajectory of the quadrotor UAV, and they used this trajectory to minimize the deviation from control input to automatically avoid collision with obstacles. However, the SLAM is required for this method to work. This method is later expanded in [35] to take the uncertainties in the on-board sensing and state estimation into account.

It can be seen that first two methods [33] and [34] do not take the robot dynamics into account and their algorithms override the operator command and stop the robot. In addition, method [15] is computationally expensive because it requires a SLAM to work. While method [15] performs automatic collision avoidance, it only works in a 2-dimensional space. Note that all the above algorithms work in a static environment only. In this thesis and for the case of a static environment, we address all the above problems. Our algorithm works in a 3-dimensional space, takes the vehicle's dynamics into account, does not require a SLAM, autonomously avoids obstacles, and ensures that the operator is in the control of the vehicle at all times.

The research in the area of collision avoidance in UAV teleoperation *in a dynamic environment* is very limited. Thus, we present some of the research about autonomous navigation in a dynamic environment.

In [22], a novel potential function for the navigation of mobile robots in a dynamic environment is presented. In this study, the target and the obstacles are moving. It is assumed that the robot encounters one moving obstacle at a time, and all other moving obstacles are far away and their effect is negligible. Furthermore, it is assumed the obstacles are ball-like-shape of radius r_i. The relative position and velocity between the robot and moving obstacle are required for collision avoidance. The proposed method was validated through simulation.

Fulgenzi et al. [36] presented an algorithm for navigation in an unknown dynamic environment. This algorithm extended the rapidly-exploring random tree algorithm where the probability of the collision is considered and is used in a partial motion planner. The performance of this algorithm is tested through simulation.

In this thesis and for the case of UAV teleoperation in a dynamic environment, we designed a new potential function that is able to deal with multiple moving obstacles at the same time and obstacle shape independent. Additionally, our potential function can deal with static and moving obstacles at the same time.

Chapter 3

SEPF Design for Autonomous Collision Avoidance

In robot motion planning literature, the gradient of a repulsive potential function typically represents a motion vector ($\mathbf{V}_{ca}$) that repels the robot away from the obstacles. In designing such a repulsive potential function, it must be ensured that a vehicle will not collide with other objects under any circumstances as well as avoids any undesirable motions. More specifically, when a vehicle is teleoperated, a vehicle should be stationary regardless of its relative distance to an obstacle when there is no motion command from the operator and also, more importantly, a vehicle should be able to stop before an obstacle no matter how fast it is approaching the obstacle. In this chapter, we present the design of the super-ellipsoidal potential function (SEPF) which provides enough flexibility in terms of designing the size and shape, and also addresses all of the above-mentioned issues.

3.1 SEPF for Collision Avoidance

We begin this section with the following assumptions.

1. The UAV is a rotary wing UAV type that can move omnidirectionally in a 3-dimensional space.

2. The UAV is equipped with a sensor that can detect obstacles around the robot within a sphere of radius (Rs) as a point cloud.

3. The UAV has dynamics that are the same in every direction with a constant deceleration limit [10].

First, to include the vehicle dynamics in the design of a repulsive potential function, we take into account the minimum stopping distance (d_{min}), which is the required distance for a vehicle to decrease its velocity to zero using the vehicle's maximum deceleration (a_{max}) allowed in the direction of motion, that is given by

$$d_{min} = \frac{\|\mathbf{v}_r\|^2}{2a_{max}} \tag{3.1.1}$$

where $\mathbf{v}_r$ is the current vehicle's velocity. In addition, to remove the unnecessary avoidance vectors due to surrounding obstacles when

- there is no motion command from the operator, the length of the repulsive potential function is set to be equal to its width as shown in Fig. 3.2;

- there is a motion command from the operator, the repulsive potential function is assigned to both current vehicle's motion and operator's command directions as shown in Fig. 3.2.

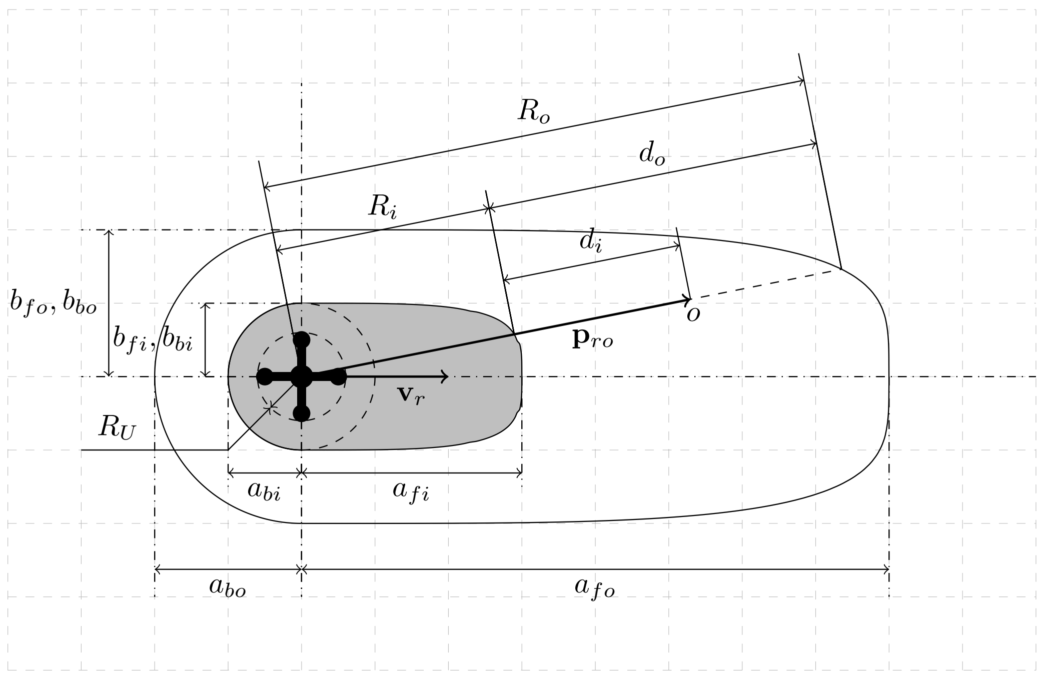


Figure 3.1: Representation of a 2-dimensional SEPF, showing the control parameters and variables.

As briefly mentioned above, we construct a repulsive potential function, called SEPF, using a pair of super-ellipses as shown in Fig. 3.1. One main advantage of using super-ellipse in the design of a potential function is that we can easily adjust the length (a), width (b), and the amount of flattening (n) at the SEPF tips. As can be seen from Fig. 3.1, an SEPF consists of an inner and outer two halves of a super-ellipse centered at the center of the vehicle.

The length (a_{fo}) of the outer half super-ellipse in the front direction of a vehicle is designed to be equal to the maximum distance sensor range when there is a motion command from an operator and to be equal to its width (b_{fo}) when there is no motion command from an operator. The width (b_{fo}) of the same front half super-ellipse is chosen to be, at least, two times greater than the radius (R_U) of the smallest circle that encircles the UAV (See Fig. 3.1). The another outer half super-ellipse facing toward the back of a vehicle has a length (a_{bo}) that is equal to its width (b_{bo}) to form a half circle behind the vehicle. To ensure the continuity between the two outer half super-ellipses, the width of the backward facing super-ellipse (b_{bo}) should be chosen to be equal to (b_{fo}). The amount of tip flatting (n) of the outer front half super-ellipse is chosen to be more than 2 so that the repulsive function can cover more area at the tip of the SPEF in the front of the vehicle. Next, for the inner half super-ellipse which is in the front direction of a vehicle, the length (a_{fi}) of the super-ellipse is designed to be

$$a_{fi} = R_U + d_s + d_{min} \tag{3.1.2}$$

where d_{min} is as in (3.1.1), R_U is as in Fig. 3.1, and d_s is the fixed safety distance that is pre-defined to restrict the minimum closest distance between a UAV and obstacles. Note that a_{fi} should be chosen to be less than a_{fo} with enough margin so that the repulsive collision avoidance motion vectors can grow smoothly from zero to its maximum allowed magnitude and also in order to ensure a technical condition in (3.1.4) for not having an infinite collision avoidance motion vector due to the division by zero. This condition is easy to meet and can be satisfied by increasing the distance sensor maximum range or limiting the vehicle speed. For the same reason, the width b_{fi} should be chosen to be less than b_{fo}. The length (a_{bi}) and width (b_{bi}) of the other inner half super-ellipse which is facing to the backward direction of a vehicle can be designed in the similar way that we design the outer backward half super-ellipse. The amount of tip flatting (n) of the inner front and back half super-ellipse are chosen to be equal to the values of (n) of the outer front and back half super-ellipse respectively. Conceptually, the inner two halves of a super-ellipse represent the forbidden region, the shaded region in Fig. 3.1. The obstacles should stay outside this region for a safe collision avoidance.

Now, let us consider a point (e.g., o) that lies in between the inner and outer super-ellipses as shown in Fig. 3.1. Let $\mathbf{p}_{ro}$ be the distance vector from the center of a UAV to the point and $\mathbf{v}_r$ be the velocity vector of the UAV. We then define two distance variables $d_i(\mathbf{p}_{ro}, \mathbf{v}_r)$ and $d_o(\mathbf{p}_{ro}, \mathbf{v}_r)$ where $d_i(\mathbf{p}_{ro}, \mathbf{v}_r)$ is the distance from the point to the inner super-ellipse along the direction of $\mathbf{p}_{ro}$ and $d_o(\mathbf{p}_{ro}, \mathbf{v}_r)$ is the distance from the inner super-ellipse to the outer super-ellipse along the direction of $\mathbf{p}_{ro}$. (Note that specific values for both d_i and d_o depend on the vector $\mathbf{p}_{ro}$ and $\mathbf{v}_r$. This dependency is represented explicitly in our notations of d_i and d_o. However, in the sequel, we use d_i to denote $d_i(\mathbf{p}_{ro}, \mathbf{v}_r)$ and d_o to denote $d_o(\mathbf{p}_{ro}, \mathbf{v}_r)$ for simplicity of notation.) Now, using these two variables, we can represent the relative position of the point with respect to the inner and outer super-ellipses of the SEPF so that the repulsive potential function value can be determined as a function of the ratio d_i/d_o. Note that if the point is on the boundary of the outer super-ellipse, then $d_i/d_o = 1$. And if the point is on the boundary of the inner super-ellipse, then $d_i/d_o = 0$.

Let $f : \mathbb{R} \to \mathbb{R}$ be a continuous real-valued function that satisfies two boundary conditions, that are $f(0) = 1$ and $f(1) = 0$. Then we formally define an SEPF ($\mathbf{P}_{Srep}$) as follows:

$$\mathbf{P_S}_{rep}(\mathbf{p}_r, \mathbf{p}_o, \mathbf{v}_r) = \begin{cases} 0, & \text{if } o \text{ is outside outer super-ellipse} \\ 1, & \text{if } o \text{ is inside inner super-ellipse} \\ \mu f\left(\frac{d_i(\mathbf{p}_{ro}, \mathbf{v}_r)}{d_o(\mathbf{p}_{ro}, \mathbf{v}_r)}\right), & \text{otherwise} \end{cases} \tag{3.1.3}$$

where μ is a design parameter used to scale the repulsive potential function, $\mathbf{p}_r$ and $\mathbf{p}_o$ are the current robot's and obstacle's position respectively. In principle, the function $f(\cdot)$ can be any function as long as it is continuous and satisfies both boundary conditions mentioned above. It could be a linear, quadratic, sine, or cosine function. However, we choose a quadratic function because the magnitude of its gradient, i.e., repulsive collision avoidance motion vector ($\mathbf{V}_{ca}$), evolves linearly from zero at the outer super-ellipse to the required maximum magnitude at the inner super-ellipse. Then, the repulsive potential field becomes

$$\mathbf{P_S}_{rep}(\mathbf{p}_{or}, \mathbf{v}_r) = \begin{cases} 0, & \text{if } o \text{ is outside outer super-ellipse} \\ 1, & \text{if } o \text{ is inside inner super-ellipse} \\ \mu\left(\frac{\|\mathbf{p}_{or}\| - R_i(\mathbf{v}_r)}{R_o - R_i(\mathbf{v}_r)} - 1\right)^2, & \text{otherwise} \end{cases} \tag{3.1.4}$$

where R_o and R_i are the distances from the center of a UAV to the outer and inner super-ellipse respectively as shown in Fig. 3.1. Note that R_i is a function of the robot's current velocity. We can see this if we substitute (3.1.1) in (3.1.2), and then (3.1.2) in (3.1.6). Also, note that R_i and R_o can be easily calculated using the following super-ellipse equation represented in polar coordinates [37]

$$R = \frac{ab}{\sqrt[n]{|acos(\theta)|^n + |bsin(\theta)|^n}} \tag{3.1.5}$$

where a, b, and n are the length, width, and the amount of tip flattening respectively and $\theta \in [-\pi, \pi]$ is the angle of the vector $\mathbf{p}_{ro}$ with respect to the horizontal axis. Two-dimensional contour plot of the SEPF is shown in Fig. 3.2.

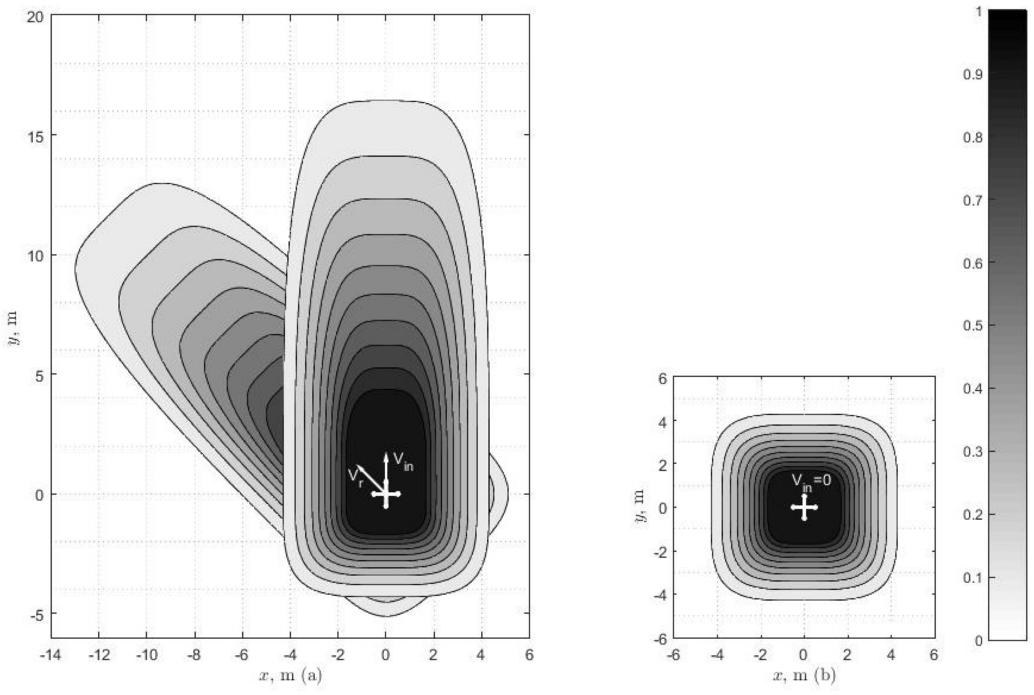


Figure 3.2: Two-dimensional contour plot of the SEPF. (a) SEPF in both operator's command ($\mathbf{V}_{in}$) and current UAV's motion directions when there is a command input from the operator. (b) SEPF length is set to be equal to its width when there is no command input from the operator.

3.1.1 SEPF in 3-Dimensional Space

Since the proposed potential function is designed based on super-ellipses, it is indeed a simple matter to extend the potential function from a 2-dimensional space to a 3-dimensional space. To extend the repulsive SEPF into a 3-dimensional space, we extend R_U in (3.1.2) to be the radius of the smallest sphere that encircles the UAV, and R in (3.1.5) to be the equation of a super-ellipsoid in spherical coordinates

[38] which is given by

$$R = \frac{1}{\sqrt[n]{\left|\frac{cos(\theta)sin(\phi)}{a}\right|^n + \left|\frac{sin(\theta)sin(\phi)}{b}\right|^n + \left|\frac{cos(\theta)}{c}\right|^n}} \tag{3.1.6}$$

where a, b, c, and n are length, width, height, and the amount of tip flattening respectively and $\theta \in [-\pi, \pi]$, $\phi \in [0, \pi]$ are two angles of a vector represented in spherical coordinate system. The contour slice plot of a 3-dimensional repulsive SEPF is shown in Fig.3.3. The height (c_{fo}) of the front half super-ellipsoid is chosen to be, at least, two times greater than the R_U. And the height (c_{bo}) of the back half super-ellipsoid is chosen to be equal to c_{fo} to guarantee the continuity between the two outer half super-ellipsoids. Note that c_{fi} should be chosen to be less than c_{fo} with enough margin so that the avoidance vectors can evolve smoothly from zero to its maximum allowed magnitude and also in order to avoid division by zero, i.e., infinite $\mathbf{V}_{ca}$ in (3.1.4). Similarly, c_{bi} is chosen to be less than c_{bo} and equal to c_{fi} to ensure the continuity between them.

3.1.2 Repulsive Collision Avoidance Vector

The SEPF is designed to provide us with the repulsive collision avoidance motion vector ($\mathbf{V}_{ca}$). The gradient of a SEPF with respect to the relative position between the vehicle and an obstacle located at (o) gives the repulsive collision avoidance motion vector. From the fact that $\nabla_{\mathbf{x}}\|\mathbf{x}\| = \mathbf{x}/\|\mathbf{x}\|$ where $\mathbf{x}$ is a vector, the repulsive collision avoidance motion vector of the SEPF when $\|\mathbf{p}_{ro}\| \in (R_i, R_o)$ is given by

$$\nabla_{\mathbf{p}_{or}}\Big\{\mu f(\mathbf{p}_{or}, \mathbf{v}_r)\Big\} = \bar{\mu}\Big(\frac{\|\mathbf{p}_{or}\| - R_i(\mathbf{v}_r)}{R_o - R_i(\mathbf{v}_r)} - 1\Big)\mathbf{n}_{or} \tag{3.1.7}$$

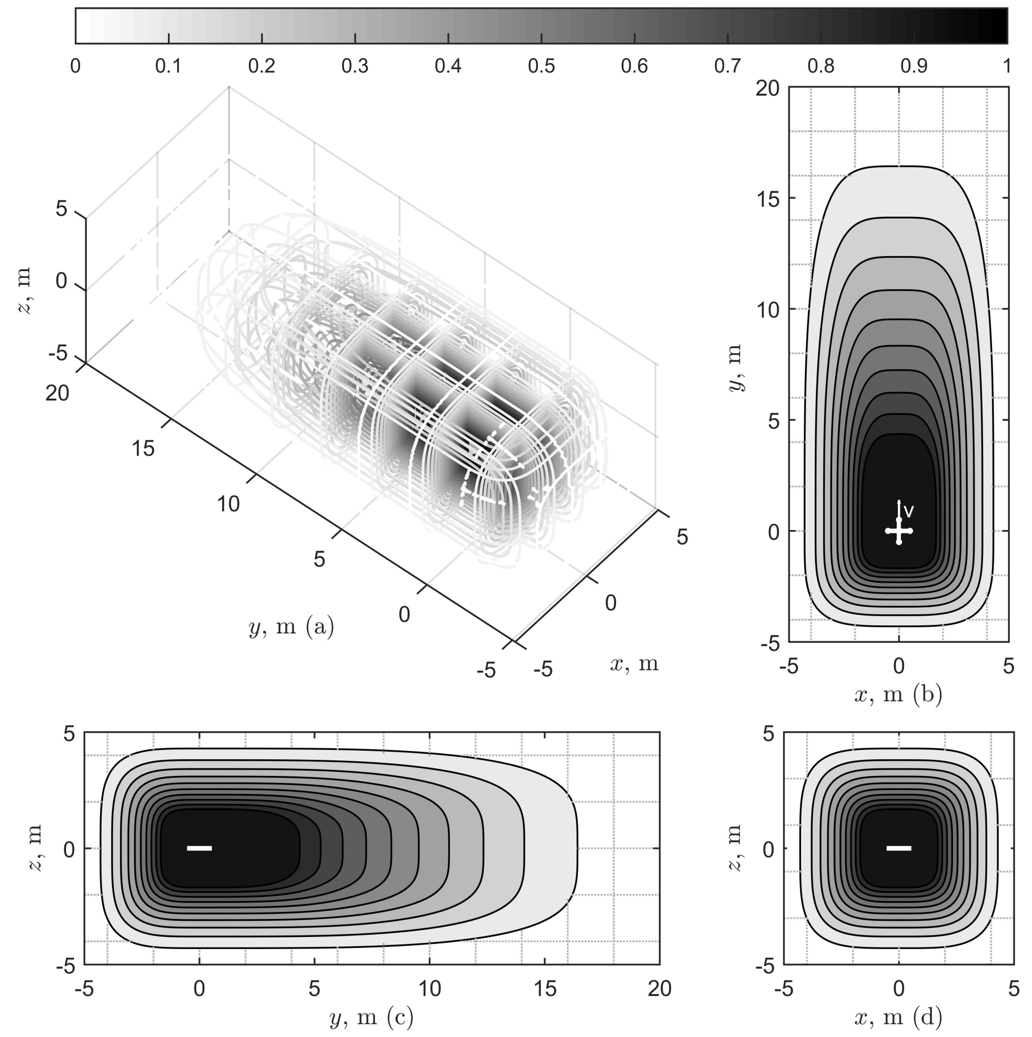


Figure 3.3: (a) Contour slice plot of the SEPF with **v**=2 m/s, $a_{fo} = 20$m, $a_{bo} = b_{fo} = b_{bo} = c_{fo} = c_{bo} = 5$m, $b_{fi} = 1.5 + v^2/2a_{max}$, $b_{fi} = c_{fi} = a_{bi} = b_{bi} = c_{bi} = 1.5$, $n = 4$, $\mu = 1$, $a_{max} = 1$. (b) one slice at $z = 0$, (c) one slice at $x = 0$, (c) one slice at $y = 0$.

where the continuous function $f(\cdot)$ is as defined in (3.1.4), $\bar{\mu} = 2\mu/(R_o - R_i)$ and $\mathbf{n}_{or} = \mathbf{p}_{or}/\|\mathbf{p}_{or}\|$. Note that the value of $((\|\mathbf{p}_{or}\| - R_i)/(R_o - R_i) - 1)$ is zero when an obstacle is at the boundary of the outer super-ellipsis and is negative one when an obstacle is on the boundary of inner one. Hence, to ensure the continuity of the repulsive motion vector on the boundary of the inner super-ellipsis, the repulsive collision avoidance motion vector is chosen to be -$\bar{\mu}\mathbf{n}_{or}$. Thus we finally have the following collision avoidance motion vectors in the case of static environment:

$$\mathbf{V}_{Sca} = \nabla_{\mathbf{p}_{or}} \mathbf{P}_{\mathbf{S}rep}(\mathbf{p}_{or}, \mathbf{v}_r)$$

$$\mathbf{V}_{Sca} = \begin{cases} 0, & \text{if } o \text{ is outside outer super-ellipse} \\ -\bar{\mu}\mathbf{n}_{or}, & \text{if } o \text{ is inside inner super-ellipse} \\ \bar{\mu}\Big(\frac{\|\mathbf{p}_{or}\| - R_i(\mathbf{v}_r)}{R_o - R_i(\mathbf{v}_r)} - 1\Big)\mathbf{n}_{or}, & \text{otherwise} \end{cases} \quad (3.1.8)$$

3.2 SEPF Based Collision Avoidance in Dynamic Environment

The design of the SEPF in Section 3.1 is for static environments. For the case of moving obstacles, we need to consider the relative velocity between the robot and the moving obstacle in addition to the relative position between them. The relative velocity (v_{ro}) between the robot and the point (o) in the direction from the robot to the point (o) in Fig. 3.1 is given by

$$v_{ro} = v_{rel}^T \mathbf{n}_{ro} \quad (3.2.1)$$

where $v_{rel} = \mathbf{v}_r - \mathbf{v}_o$, $\mathbf{v}_r$ is the vehicle's current velocity, $\mathbf{v}_o$ is the obstacle's current velocity and $\mathbf{n}_{ro} = \mathbf{p}_{ro}/\|\mathbf{p}_{ro}\|$ is a unit vector directing from the robot to the obstacle. Then d_{min} in (3.1.1) becomes

$$d_{min} = \frac{v_{ro}^2}{2a_{max}} \tag{3.2.2}$$

Note that when the robot is moving away from an obstacle, i.e., $v_{ro} \leq 0$, collision avoidance algorithm is not needed. However, when the robot is approaching an obstacle, i.e., $v_{ro} > 0$, collision avoidance algorithm is considered. Then (3.1.4) becomes

$$\mathbf{P}_{\mathbf{D}rep}(\mathbf{p}_{or}, v_{rel}) = \begin{cases} 0, & \text{if } o \text{ is outside outer super-ellipsoid} \\ 1, & \text{if } o \text{ is inside inner super-ellipsoid} \\ \mu\Big(\frac{\|\mathbf{p}_{or}\| - R_i(\mathbf{p}_{or}, v_{rel})}{R_o - R_i(\mathbf{p}_{or}, v_{rel})} - 1\Big)^2, & \text{otherwise} \end{cases} \tag{3.2.3}$$

In the sequel, we use $\mathbf{P}_{\mathbf{D}rep}$ to denote $\mathbf{P}_{\mathbf{D}rep}(\mathbf{p}_{or}, v_{rel})$ and R_i to denote $R_i(\mathbf{p}_{or}, v_{rel})$ for simplicity of notation. Note that R_i is a function of the relative position ($\mathbf{p}_{or}$) and relative velocity (v_{rel}) between the robot and the obstacle. We can see this if we substitute the equation of v_{rel} (3.2.2) which is also a function of the relative position and relative velocity between the robot and the obstacle in (3.1.2), and then (3.1.2) in (3.1.6).

3.2.1 Repulsive Collision Avoidance Vector

The gradient of an SEPF with respect to the relative position ($\mathbf{p}_{or}$) and relative velocity (v_{rel}) between the robot and point (o) in Fig. 3.1 gives the avoidance vector ($\mathbf{V}_{Dca}$). The repulsive collision avoidance motion vector of the SEPF when $\|\mathbf{p}_{ro}\| \in$

(R_i, R_o) and $v_{ro} > 0$ is given by

$$\mathbf{V}_{Dca} = \nabla_{\mathbf{p}_{or}} \mathbf{P}_{\mathbf{D}rep} + \nabla_{v_{rel}} \mathbf{P}_{\mathbf{D}rep} \tag{3.2.4}$$

the gradient of the third row of (3.2.3) with respect to the relative position ($\mathbf{p}_{or}$) and velocity (v_{rel}) is given by

$$\nabla_{\mathbf{p}_{or}} \mathbf{P}_{\mathbf{D}rep} = \frac{2\mu}{(R_o - R_i)^2} \Big(\frac{\|\mathbf{p}_{or}\| - R_i}{R_o - R_i} - 1 \Big) \Re 1 \tag{3.2.5}$$

$$\nabla_{v_{rel}} \mathbf{P}_{\mathbf{D}rep} = \frac{2\mu}{(R_o - R_i)^2} \Big(\frac{\|\mathbf{p}_{or}\| - R_i}{R_o - R_i} - 1 \Big) \Re 2 \tag{3.2.6}$$

where $\Re 1$ is

$$\Re 1 = (R_o - R_i)(\nabla_{\mathbf{p}_{or}} \|\mathbf{p}_{or}\| - \nabla_{\mathbf{p}_{or}} R_i) - (\|\mathbf{p}_{or}\| - R_i)(\nabla_{\mathbf{p}_{or}} R_o - \nabla_{\mathbf{p}_{or}} R_i),$$

and $\Re 2$ is

$$\Re 2 = (R_o - R_i)(\nabla_{v_{rel}} \|\mathbf{p}_{or}\| - \nabla_{v_{rel}} R_i) - (\|\mathbf{p}_{or}\| - R_i)(\nabla_{v_{rel}} R_o - \nabla_{v_{rel}} R_i).$$

If we substitute (3.2.2) in (3.1.2) and then (3.1.2) in (3.1.6), we get

$$R_i = \frac{1}{\sqrt[n]{\left| \frac{cos(\theta) sin(\phi)}{\frac{v_{ro}^2}{2a_{max}} + R_U + d_s} \right|^n + \left| \frac{sin(\theta) sin(\phi)}{b} \right|^n + \left| \frac{cos(\theta)}{c} \right|^n}}. \tag{3.2.7}$$

To calculate the gradient of R_i with respect to the relative position ($\mathbf{p}_{or}$) and velocity (v_{rel}), we need to calculate the gradient of v_{ro} with respect to the relative position

($\mathbf{p}_{or}$) which is given by

$$\nabla_{\mathbf{p}_{or}} v_{ro} = -\frac{v_{rel} + v_{ro}\mathbf{n}_{or}}{\|\mathbf{p}_{ro}\|}, \tag{3.2.8}$$

and the gradient of v_{ro} with respect to the relative velocity (v_{rel}) which is given by

$$\nabla_{v_{rel}} v_{ro} = \mathbf{n}_{ro} = -\mathbf{n}_{or}. \tag{3.2.9}$$

Now using (3.2.7), (3.2.8), and (3.2.9), we obtain the gradient of R_i with respect to relative position ($\mathbf{p}_{or}$) and relative velocity (v_{rel})

$$\nabla_{\mathbf{p}_{or}} R_i = -\xi v_{ro} \frac{v_{rel} + v_{ro}\mathbf{n}_{or}}{\|\mathbf{p}_{ro}\|} \tag{3.2.10}$$

$$\nabla_{v_{rel}} R_i = -\xi v_{ro}\mathbf{n}_{or} \tag{3.2.11}$$

where ξ is

$$\xi = \frac{\left|\frac{cos(\theta)sin(\phi)}{\frac{v_{ro}^2}{2a_{max}}+R_U+d_s}\right|^{n-1} sign(\frac{cos(\theta)sin(\phi)}{\frac{v_{ro}^2}{2a_{max}}+R_U+d_s})\frac{cos(\theta)sin(\phi)}{(\frac{v_{ro}^2}{2a_{max}}+R_U+d_s)^2 a_{max}}}{\sqrt[n]{\left(\left|\frac{cos(\theta)sin(\phi)}{\frac{v_{ro}^2}{2a_{max}}+R_U+d_s}\right|^n + \left|\frac{sin(\theta)sin(\phi)}{b}\right|^n + \left|\frac{cos(\theta)}{c}\right|^n\right)^{1+n}}} \tag{3.2.12}$$

We also need to know the following gradients to calculate (3.2.5) and (3.2.6)

$$\nabla_{\mathbf{p}_{or}} \|\mathbf{p}_{or}\| = \mathbf{n}_{or} \tag{3.2.13}$$

$$\nabla_{v_{rel}} \mathbf{p}_{or} = \nabla_{v_{rel}} R_o = \nabla_{\mathbf{p}_{or}} R_o = \mathbf{0}_{3\times1}. \tag{3.2.14}$$

Now, substituting (3.2.10), (3.2.11), (3.2.13), (3.2.14) in (3.2.5), (3.2.6) and then (3.2.5), (3.2.6) in (3.2.4) , we obtain the final value of the gradient of the third row of (3.2.3) which is given by

$$\mathbf{V}_{Dca} = \bar{\mu}_1 \rho \mathbf{n}_{or} - \bar{\mu}_2 \rho \frac{v_{rel} + v_{ro}\mathbf{n}_{or}}{\|\mathbf{p}_{ro}\|} \tag{3.2.15}$$

where $\bar{\mu}_1 = \frac{2\mu}{(R_o-R_i)^2}(R_o - R_i - (\|\mathbf{p}_{or}\| - R_o)\xi v_{ro})$, $\bar{\mu}_2 = \frac{2\mu}{(R_o-R_i)^2}(\|\mathbf{p}_{or}\| - R_o)\xi v_{ro}$ and $\rho = (\frac{\|\mathbf{p}_{or}\|-R_i}{R_o-R_i} - 1)$. The value of ρ is zero when point (o) is on the boundary of the outer super-ellipse, while it is negative one when point (o) is on the boundary of inner one. Hence, the gradient of the SEPF is chosen to be $-(\bar{\mu}_1 \mathbf{n}_{or} - \bar{\mu}_2 \frac{v_r el + v_{ro}\mathbf{n}_{or}}{\|\mathbf{p}_{ro}\|})$ when the point (o) is inside the inner ellipse to ensure the continuity of the repulsive avoidance vector.

Thus, we finally have the following collision avoidance motion vectors in case of *dynamic* obstacles:

$$\mathbf{V}_{Dca} = \begin{cases} 0, & \text{if } o \text{ is outside outer super-ellipsoid} \\ -\bar{\mu}_1 \mathbf{n}_{or} + \bar{\mu}_2 \frac{v_{rel}+v_{ro}\mathbf{n}_{or}}{\|\mathbf{p}_{ro}\|}, & \text{if } o \text{ is inside inner super-ellipsoid} \\ \bar{\mu}_1 \rho \mathbf{n}_{or} - \bar{\mu}_2 \rho \frac{v_{rel}+v_{ro}\mathbf{n}_{or}}{\|\mathbf{p}_{ro}\|}, & \text{otherwise} \end{cases} \tag{3.2.16}$$

It can be seen form (3.2.16) that the relative position and relative velocity are only needed to avoid collision with the moving obstacles.

3.3 Collision Avoidance under Teleoperation

3.3.1 Generation of The Reference Velocity Command for a UAV

In teleoperation, an operator sends control commands to a UAV through a teleoperation device which is an Xbox 360 controller in our case. The operator's command is considered as a velocity control input ($\mathbf{V}_{in}$) in the simulation and angle control input ($\mathbf{A}_{in}$) in the experimentation. Now, to generate the control reference input that a UAV should track in the case of static obstacles, the operator's control input and collision avoidance motion vector ($\mathbf{V}_{Sca}$) can be integrated into a single vector. This integration is a simple vector sum

$$\mathbf{V}_{ref} = \mathbf{V}_{in} - \mathbf{V}_{Sca} \tag{3.3.1}$$

in case of simulation where $\mathbf{V}_{ref}$ is the reference velocity that a UAV should track, and

$$\mathbf{A}_{ref} = \mathbf{A}_{in} - \mathbf{A}_{Sca} \tag{3.3.2}$$

in case of experimentation where $\mathbf{A}_{ref}$ is the reference angle that a UAV should track. However, to generate the control reference input that a UAV should track in the case of dynamic obstacles, the operator control input and collision avoidance motion vector from both static and dynamic obstacles can be integrated into a single vector.

$$\mathbf{V}_{ref} = \mathbf{V}_{in} - \mathbf{V}_{Sca} - \sum_{i=1}^{m} \mathbf{V}_{Dca_i} \tag{3.3.3}$$

in case of simulation where $\mathbf{V}_{Sca}$ and $\mathbf{V}_{Dca}$ are the velocity avoidance vectors in case of static and dynamic obstacles respectively, and

$$\mathbf{A}_{ref} = \mathbf{A}_{in} - \mathbf{A}_{Sca} - \sum_{i=1}^{m} \mathbf{A}_{Dca_i} \tag{3.3.4}$$

in case of experimentation where $\mathbf{A}_{Sca}$ and $\mathbf{A}_{Dca}$ are the angle avoidance vectors in case of static and dynamic obstacles respectively and m is the number of moving obstacles.

3.3.2 Repulsive Collision Avoidance Vector From Multiple Points

The discussion in Section 3.1.2 and 3.2.1 about the repulsive collision avoidance motion vector ($\mathbf{V}_{ca}$) is for one point on an obstacle. However, a 3D range sensor usually detects a point cloud on the obstacle. There are several ways to combine multiple repulsive collision avoidance motion vectors from multiple points on an obstacle into one final avoidance vector.

1. The sum of all the repulsive vectors generated from the point cloud. In general, this method results in a large avoidance vector which is undesirable because it restricts the UAV motion.

2. The mean of all repulsive vectors generated from the point cloud. This method sometimes leads to a small avoidance vector and a collision may occur because large avoidance vectors are averaged with small avoidance vectors.

3. The sum of maximum positive and minimum negative components of repulsive vectors. Let $\mathbf{V}_{fca} = [x_f, y_f, z_f]^T$ be the final avoidance vector, $\mathbf{x}_p$ be the

vector of positive components constructed from the positive component of each avoidance vector in x direction and $\mathbf{x}_n$ be the vector of negative components constructed from the negative component of each avoidance vector in x direction, then $x_f = \max(\mathbf{x}_p) + \min(\mathbf{x}_n)$. y_f and z_f can be calculated in the same way. In this method, the tangential repulsive vectors' components of a symmetric point cloud around an obstacle will cancel each other. For example, if the UAV is moving perpendicular to the wall, it will stop before hitting the wall because the tangential components to the wall will cancel each other while the normal ones force the vehicle to stop (see Fig. 4.2). However, if a UAV is moving towards the wall with an acute angle, it will deviate its path when approaching to the wall and then move parallel to the wall as shown in Fig. 4.4b.

We used the last method during the simulation and experimentation to validate the proposed method.

3.3.3 Directions of The SEPFs

In the case of static obstacles, the direction of a repulsive potential function is typically in the direction of current vehicle motion. However, in teleoperation case, the SEPF is assigned to both operator's control input and current vehicle's motion directions. If it is assigned to current vehicle's motion direction only, this will lead to a chattering behavior and the UAV will eventually collide with obstacles. For example, if a UAV is moving with an acute angle towards the wall, the vehicle first reaches the closest safety distance (d_s) to the wall and it continues to move along the wall and hence the SEPF direction is along the wall as well. However, the user control input is still towards the wall, and the robot follows it again because there is no SEPF towards the wall, but there is not enough time for the repulsive vector to prevent the collision this time, i.e., the wall entered the forbidden region of the SEPF. Further, if the SEPF is assigned to current operator's command direction only, this might lead to a collision with obstacles because the generated avoidance vector might steer the vehicle to a direction where there is no SEPF. In the case of dynamic obstacles, when obstacles enter the distance sensor's sensing range and $v_{ro} > 0$, a new SEPF is assigned to the direction of each moving obstacle in addition to the SEPFs from the static obstacles case.

3.3.4 Magnitude of The Repulsive Vector

In the case of static obstacles, the maximum magnitude of the repulsive vector resulting from the SEPF in the direction of an operator's control input is chosen to be equal to the magnitude of the operator control input, i.e., $\bar{\mu} = \|\mathbf{V}_{in}\|$. The reason for choosing $\bar{\mu}$ to be varying with an operator's control input is to limit the magnitude of the collision avoidance motion vector to the current magnitude of the operator's control input since a large repulsive avoidance vector is not needed when the UAV is commanded to move with very small velocities. For similar reasons, the maximum magnitude of the repulsive vector resulting from the SEPF in the direction of motion is also chosen to be equal to the magnitude of current vehicle velocity ($\mathbf{v}_r$), i.e., $\bar{\mu} = \|\mathbf{v}_r\|$. In the case of dynamic obstacles, an appropriate value for μ in (3.2.3) is chosen to scale $\mathbf{V}_{Dca}$ and make the collision avoidance process smooth.

Chapter 4

Simulation

A human-in-the-loop simulation for the teleoperation of a quadrotor UAV is implemented using the virtual robot experimentation platform (v-rep) program [39] in conjunction with Matlab program to validate the performance of the proposed autonomous collision avoidance framework using the SEPFs.

4.1 Simulation Setup

All the simulations and calculations were performed on a laptop computer with an Intel(R) Core(TM) i7-3632QM CPU and 8GB of RAM.

The simulation for quadrotor UAV dynamics is performed inside the v-rep. V-rep is a powerful simulator in which one can create, compose and simulate any robot. It has numerous built-in models which can be separately controlled by an embedded script, a plugin, a ROS node, a remote API client, or a custom solution [40]. It can be programmed using many programming languages such as Matlab, C/C++, and Lua. Attitude, velocity and position tracking controller were built, implemented using Lua language, inside v-rep to stabilize/control the quadrotor UAV. A built-in 3D sensor model inside v-rep was used to measure the relative distance to obstacles.

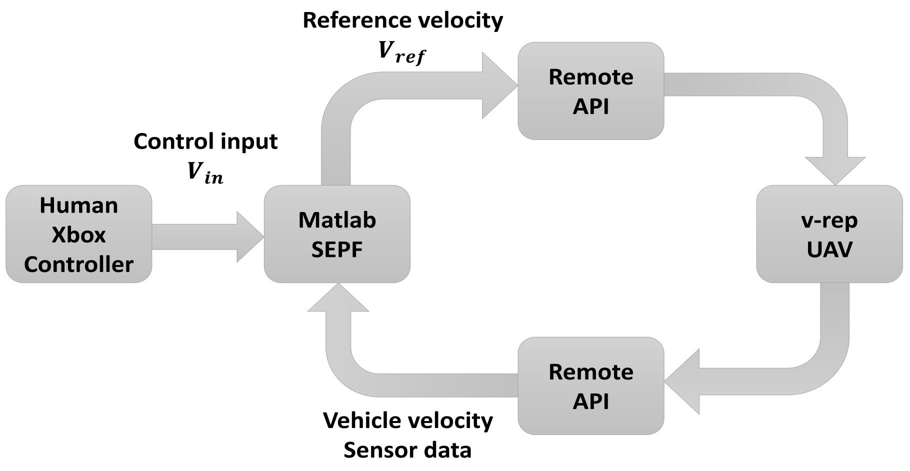


Figure 4.1: Simulation diagram showing data transfer between Matlab and v-rep and from human operator to Matlab.

As shown in Fig. 4.1, there is a human operator who drives the virtual quadrotor UAV inside v-rep. Also, there is a Matlab program in between the operator and v-rep that is implemented to

- take input commands from the operator to calculate $\mathbf{V}_{in}$,
- receive simulated velocity of the quadrotor UAV as well as sensor data from v-rep to calculate $\mathbf{V}_{Sca}$ and $\mathbf{V}_{Dca}$ (in the case of dynamic obstacles), and
- perform all necessary calculations to generate a reference velocity $\mathbf{V}_{ref}$ and send it to the v-rep quadrotor UAV model through a remote API connection.

4.2 Teleoperation of a Quadrotor UAV in Static Environment

In this section, simulation results for the teleoperation of a quadrotor UAV in an environment with static obstacles are presented. We found that the best way to express the results is the action sequence of images technique. In all simulation figures, the black curve, the red arrow, and the blue arrow represent the path of the quadrotor UAV, operator's command direction, and current UAV's motion direction respectively.

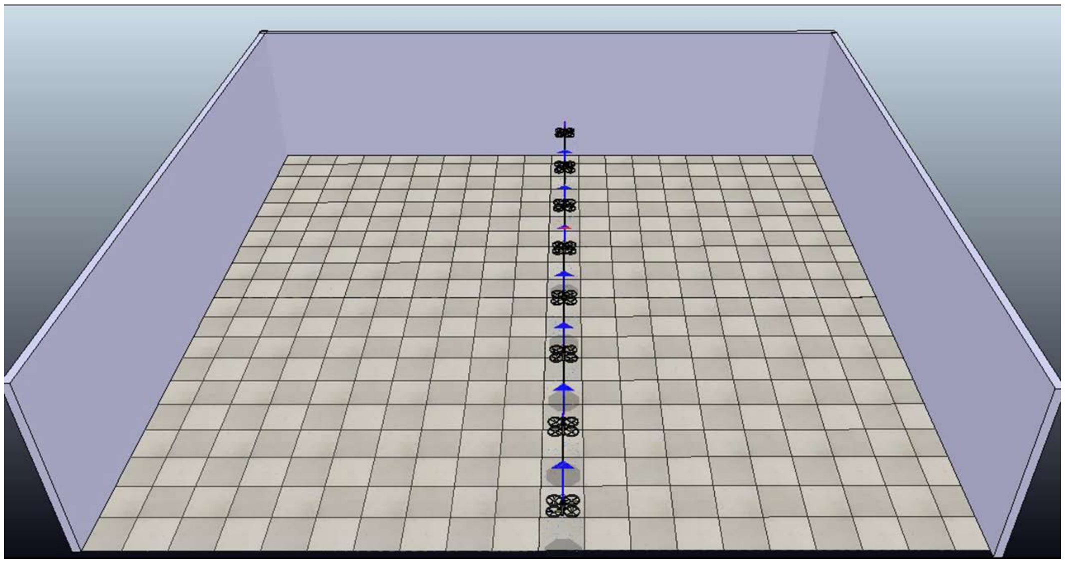


Figure 4.2: The quadrotor is commanded to move forward with its maximum velocity towards the wall in front

Fig. 4.2 shows the case where the operator steers the quadrotor UAV forward towards the wall in front with its maximum velocity. The quadrotor UAV gradually decreased its velocity and came to a full stop before the wall when it reached the minimum safety distance to the wall (d_s). The reason behind this behavior is that the component of the avoidance vector ($\mathbf{V}_{Sca}$) perpendicular to the wall cancels the operator command ($\mathbf{V}_{in}$) component which also perpendicular to the wall and has the same magnitude of the perpendicular component of $\mathbf{V}_{Sca}$ but opposite direction. Furthermore, the tangential components of the $\mathbf{V}_{Sca}$ to the wall cancel each other because the point cloud detected on the wall is symmetric.

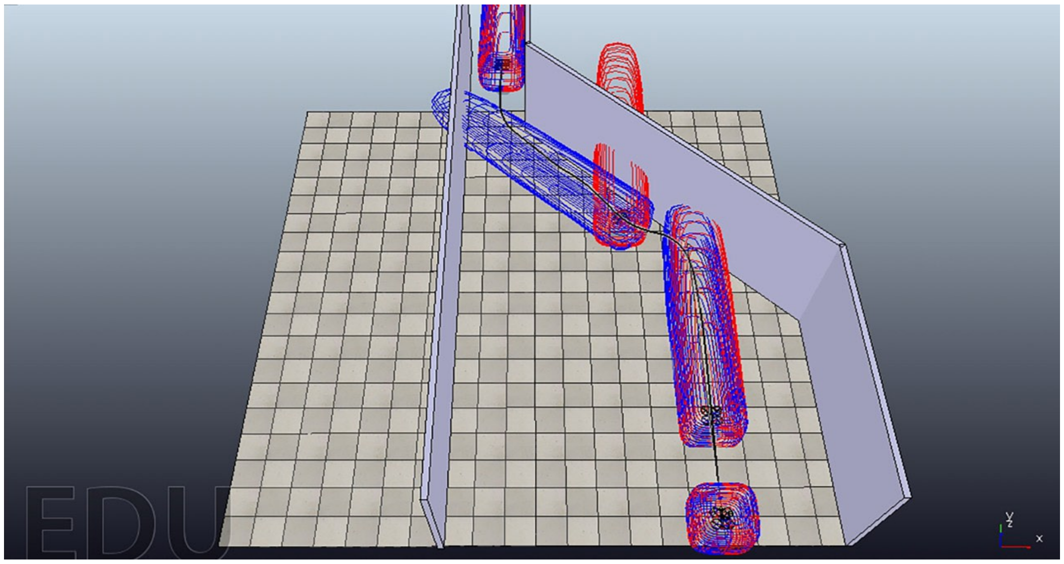
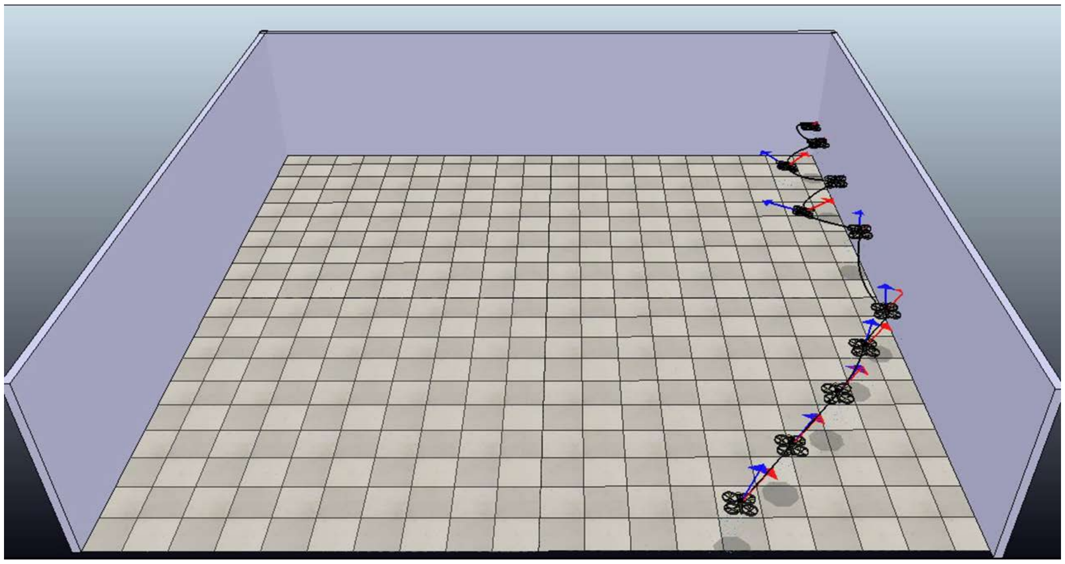


Figure 4.3: The quadrotor is moving towards a sloped wall in $(x-y)$ plane. The red SEPF represents operator's command direction and the blue one represents current UAV's motion direction.

Fig. 4.3 represents the case when an operator keeps commanding the quadrotor UAV to move straightforward with its maximum velocity to make the quadrotor UAV collide with the sloped wall in front. As shown in the figure, the quadrotor UAV does not collide with the wall despite the operator's continuous command in the forward direction. Instead, the vehicle continues to follow the operator's command while avoiding collision with the wall. The reason for this behavior is that when the quadrotor UAV comes close to the wall, the component in $\mathbf{V}_{in}$ which is perpendicular to the wall is canceled by the component in $\mathbf{V}_{Sca}$ which is also perpendicular and has the same magnitude but opposite direction, and the velocity vector component in $\mathbf{V}_{in}$ which is parallel to the wall is tracked by the quadrotor UAV. Further, when the quadrotor UAV continues to approach the left side wall, the blue SEPF constructed along the direction of the vehicle's motion prevents the collision this time, and the vehicle eventually enters the narrow passage without having any collision at all.

In Fig. 4.4, the quadrotor UAV is driven to approach the right side wall with an acute angle. Fig. 4.4a shows the case that the quadrotor UAV fails to avoid collision with the right side wall because the SEPF is assigned to the current UAV's motion direction only as discussed in Section 3.3.3. On the other hand, when the SEPF is assigned to *both* current UAV's motion and operator's command directions, the quadrotor UAV succeeds to avoid the collision with the wall. Then, it moves along the wall until it reaches the corner and stops there. The reason behind this behavior is that the component in the operator's command ($\mathbf{V}_{in}$) that is perpendicular to the wall is gradually decreased as the quadrotor UAV approaches the right side wall by the perpendicular component to the wall in $\mathbf{V}_{Sca}$ until these components cancel each others while the component in the operator's command ($\mathbf{V}_{in}$) that is parallel to the wall is followed by the quadrotor UAV. When the vehicle reaches the corner, it stops there because the two components in $\mathbf{V}_{in}$ and $\mathbf{V}_{Sca}$ are now perpendicular to the walls and have the same magnitudes with opposite directions as shown in Fig. 4.4b.

In Fig. 4.5, the quadrotor UAV is moving forward with its maximum velocity towards an obstacle protruded from the right side wall. Fig. 4.5a illustrates that the quadrotor UAV succeeds to bypass the protruded obstacle from the right side wall but collides with the left side wall where the SEPF is directed to the opera-


(a) The SEPF is in the direction of current UAV's motion direction only; collision occurred.

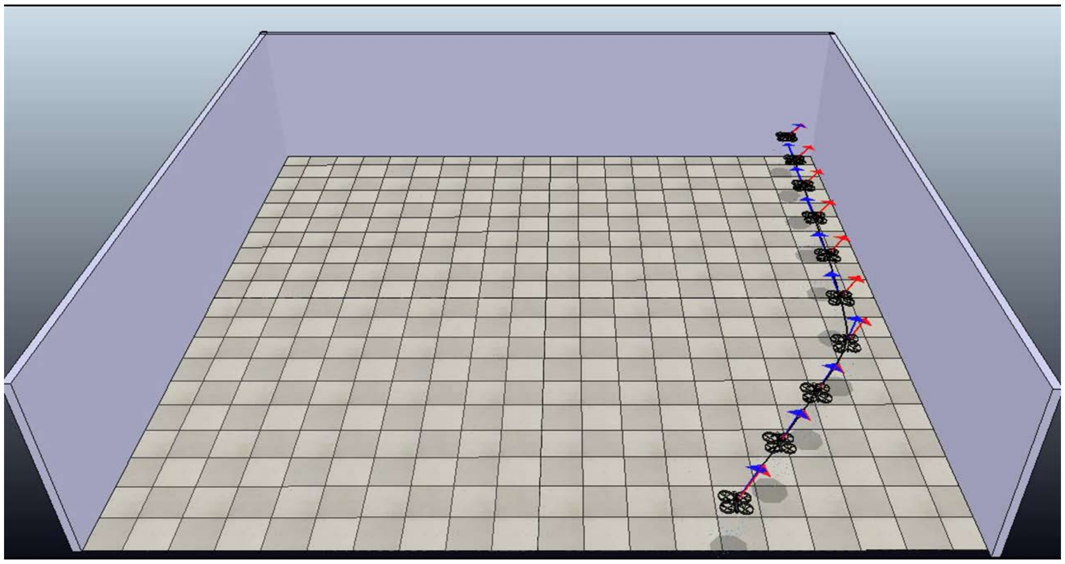
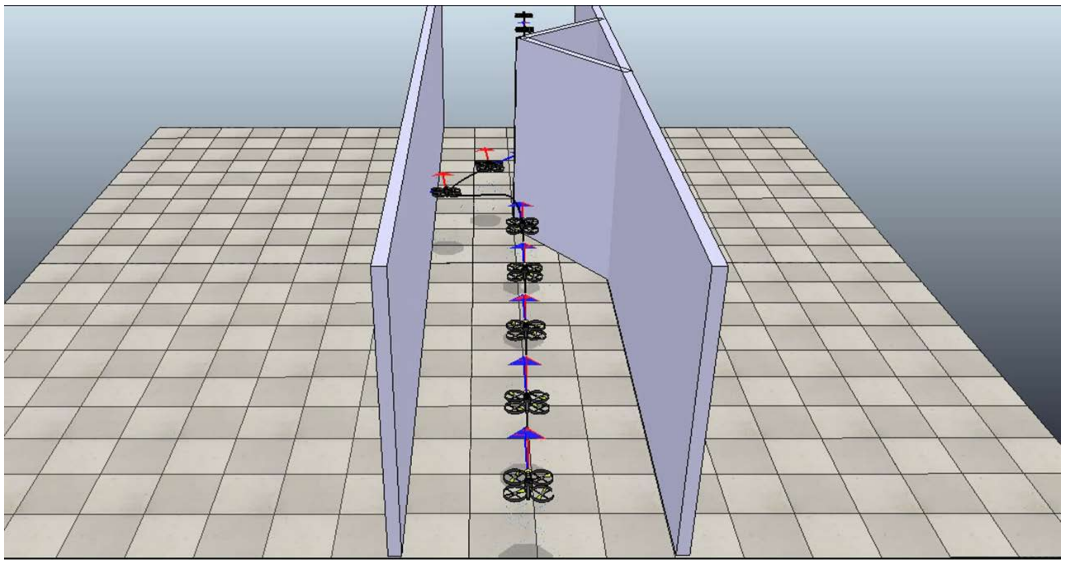

(b) The SEPF is in both current UAV's motion and operator's command direction; no collision occured

Figure 4.4: The quadrotor is moving with an acute angle towards the wall. Collision occurred in the top figure while no collision occurred in the bottom figure because of the direction of the SEPFs.

tor's command direction only and no SEPF is directed towards the left side wall to prevent the collision as discussed in Section 3.3.3. On the other hand, when the


(a) The SEPF is in the direction of operator's direction only; collision took place.

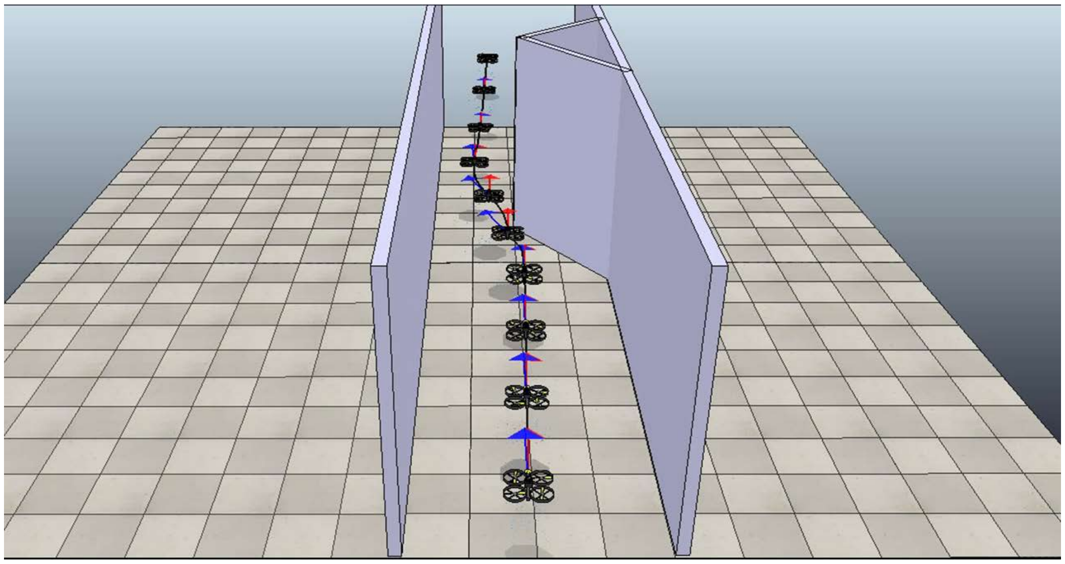

(b) The SEPF is in both current UAV's motion and operator's command direction; no collision took place

Figure 4.5: The quadrotor is steered towards an obstacle protruded from the right wall. Collision took place in the top figure while no collision took place in the bottom figure because of the direction of the SEPFs.

SEPF is directed to both current UAV's motion and operator command direction, the quadrotor UAV succeeds to avoid the collision with both the left side wall and protruded obstacle from the right side wall. The UAV first decreases it forward velocity and then moves parallel to the sloped obstacle. Now, the SEPF constructed in the current UAV's motion direction is facing the left side wall and hence prevents collision with it because the components of the ($\mathbf{V}_{Sca}$) from the left side wall and the protruded obstacle cancel each other and the UAV finally tracks the operator command which is still in forward direction as shown in Fig. 4.5b.

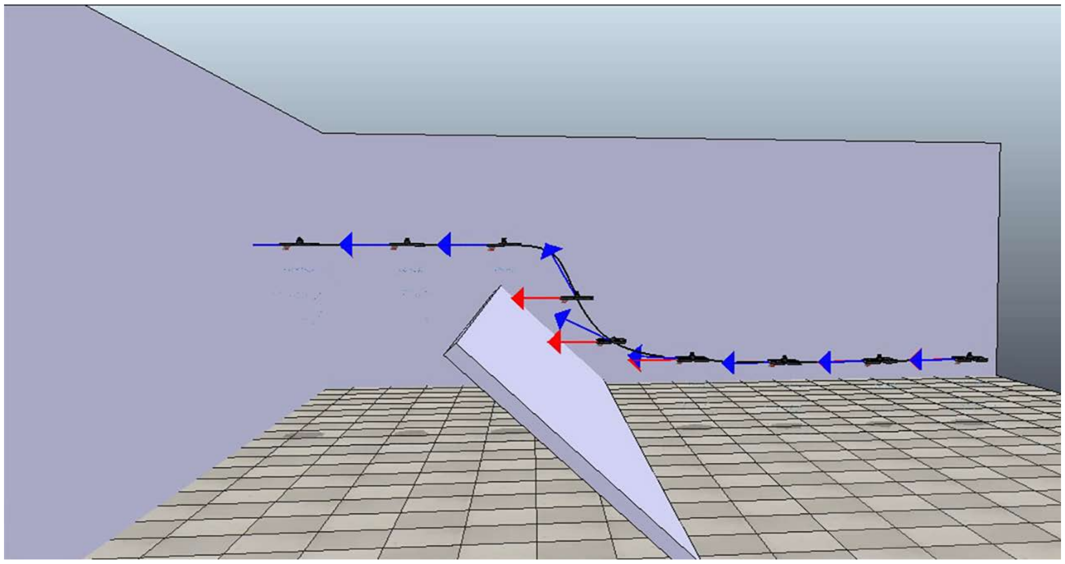


Figure 4.6: The quadrotor is moving towards a sloped wall in $(y - z)$ plane. The quadrotor avoided collision with sloped wall and stopped before the wall.

Fig. 4.6 represents the case when the quadrotor UAV is steered forward towards a sloped wall in $(y - z)$ plane. As shown in the figure, the quadrotor UAV slows its velocity and then deviates its path to move along the sloped surface. After avoiding the sloped surface, the quadrotor UAV continues to move forward and stops before the wall, similar to the case of Fig. 4.2.

4.3 Teleoperation of a Quadrotor UAV in Dynamic Environment

Simulation results for the teleoperation of a quadrotor UAV around moving objects are presented in this section. Note that, in all simulation figures presented in this section, when the quadrotor UAV is at position 1, the moving obstacle is at position 1 as well and when the quadrotor UAV is at position 2, the moving obstacle is at position 2 as well and so on.

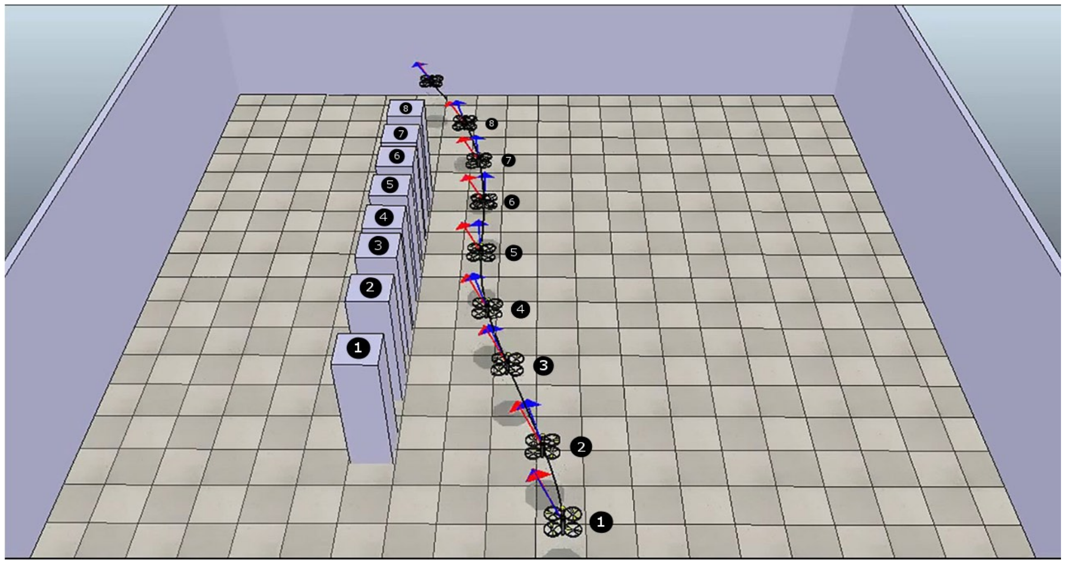


Figure 4.7: The quadrotor is commanded to move towards a vertically moving obstacle. The quadrotor UAV succeeded in bypassing the moving obstacle.

In Fig. 4.7, the operator commands the vehicle to move towards a vertically moving obstacle (with a speed of 1 m/s). The vehicle avoids collision with the obstacle, moves parallel to the obstacle, and then bypasses it. This happens because the components of the avoidance vector ($\mathbf{V}_{Dca}$) decrease the relative velocity

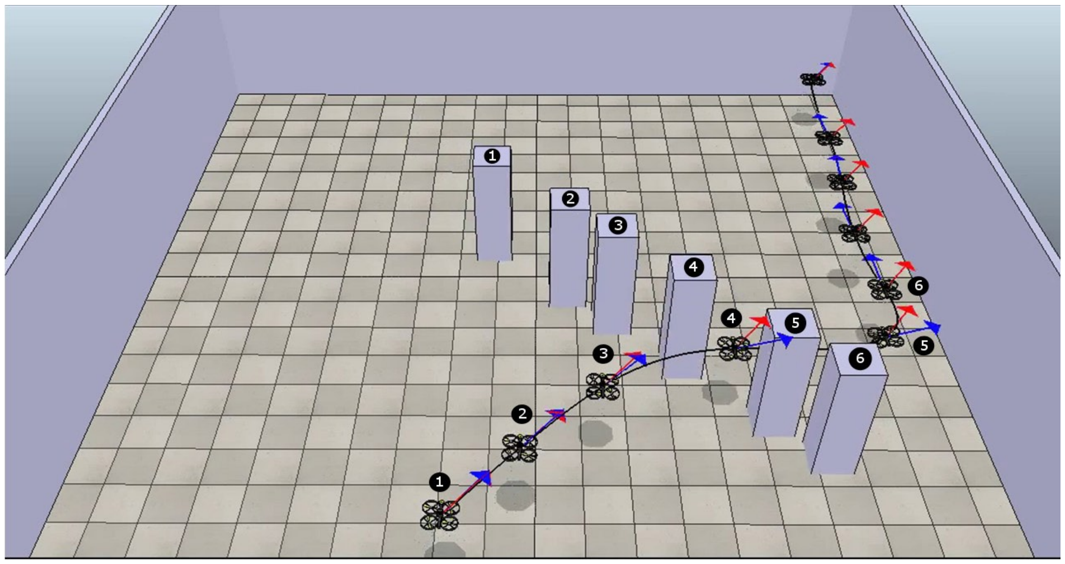


Figure 4.8: The quadrotor is commanded to fly towards a diagonally moving obstacle. The quadrotor UAV avoided the collision with the moving obstacle and moved parallel to the wall.

towards the obstacle (v_{ro}) to prevent the collision the obstacle and increase the relative velocity perpendicular to v_{ro} to drive the vehicle to bypass the obstacle.

Fig. 4.8 illustrates the case when the operator drives the vehicle towards a diagonally moving obstacle (with a speed of 1 m/s). At first, the quadrotor UAV decreases its relative velocity toward the moving obstacle, bypasses it at position 4, then moves parallel to the wall, and finally stops at the corner which is similar to the case of Fig. 4.4b.

The quadrotor UAV avoiding multiple moving obstacles (each with a speed of 1 m/s) is shown in Fig. 4.9. The quadrotor UAV first deviates its path due to the black obstacle to avoid colliding with it. Then, at position 5, it faces another moving obstacle (the blue one), deviates its path again away from the obstacle and tracks operator's command perfectly because there is no more obstacles until it faces a wall, and stops there because the perpendicular avoidance vector component

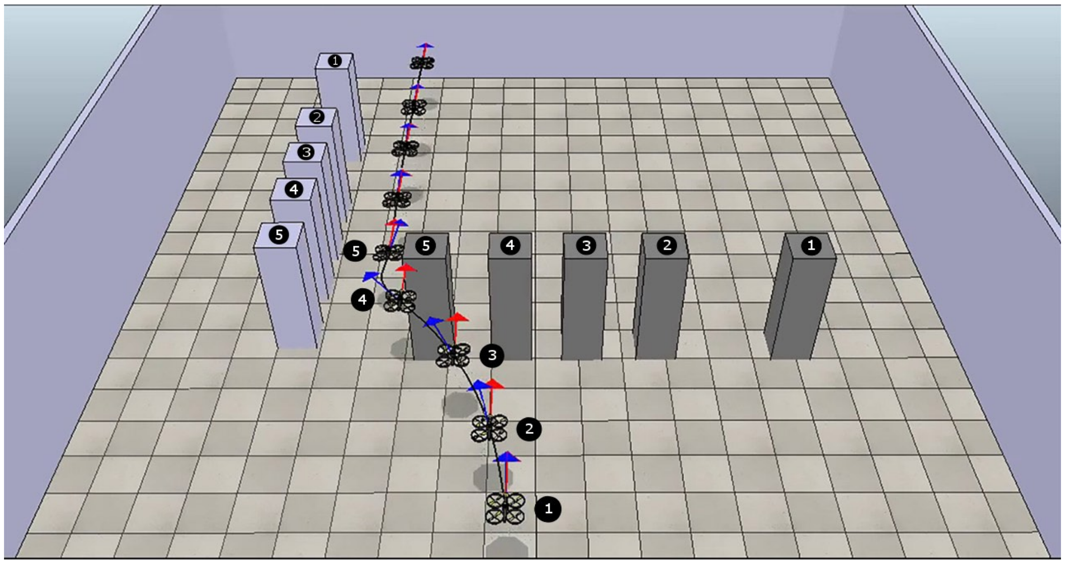


Figure 4.9: The quadrotor UAV is steered forward where there are two moving obstacles in the environment.

generated from the wall cancels command vector component perpendicular to the wall, and there is no command vector component parallel to wall to track by the vehicle.

To prove that our algorithm also works in a 3-dimensional space, a case of Fig. 4.10 is implemented. The vehicle moves towards an up and down moving sloped obstacle (with a speed of 1 m/s). The first rise in robot's path is because the obstacle is moving upward (not shown in the figure for the clarity of presentation). Then, the robot moves forward in $x - y$ plane, i.e., tracking operator command, because the obstacle is moving downward ($v_{ro} \leq 0$, i.e., $\mathbf{V}_{Dca} = \mathbf{0}$) and the robot sees no static obstacles ($\mathbf{V}_{Sca} = \mathbf{0}$). The second rise in robot's path is because the obstacle is now moving upward again, i.e., position 1,2 and 3. The robot moves upward to avoid collision with the moving obstacle. Then, the robot tracks operator's command and moves forward.

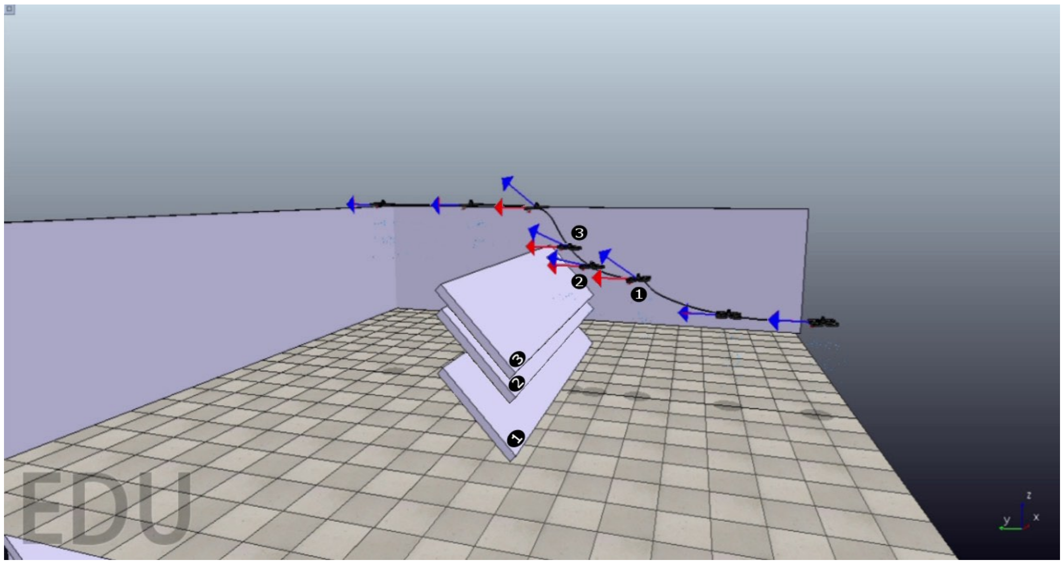


Figure 4.10: The quadrotor is moving towards an up and down moving sloped obstacle.

Chapter 5

Experimentation

A human-in-the-loop experiments for the teleoperation of a quadrotor UAV is implemented using a physical quadrotor UAV, AR.Drone 2.0 which is shown in Fig. 5.1, and a motion capture system in a laboratory environment of dimensions $4m \times 3m \times 2m$ with the help of Robot Operating System (ROS) [41] to validate the performance of the proposed autonomous collision avoidance framework using the SEPFs.

Figure 5.1: AR.Drone 2.0 photo [3].

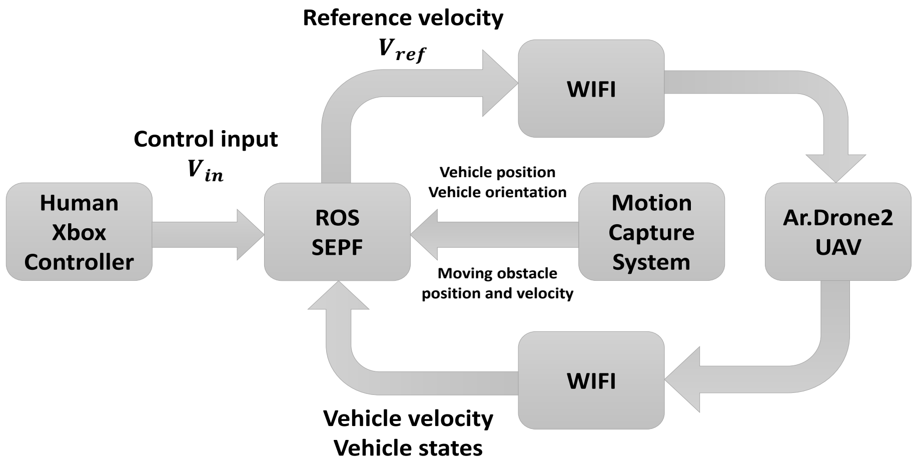


Figure 5.2: Experimentation diagram showing data transfer between ROS and AR.Drone 2.0 and from human operator to ROS.

5.1 Experimental Setup

All the experiments and calculations were performed on the same computer that is used in the simulation of the proposed method.

A virtual 3D range sensor is implemented using the data provided by the motion capture system. This virtual sensor supplies a point cloud of distance measurements. In all experiments, the obstacles are virtual and predefined. A position tracking controller is also implemented to improve the hovering performance of the AR.Drone 2.0 quadrotor. All the calculations during the experiments are performed within ROS framework. Fig. 5.2 illustrates the experimentation diagram. As shown in the figure, there is a human operator who drives the AR.Drone 2.0 quadrotor via an Xbox controller. Also, there is a program inside ROS written in C++. The function of this program is to

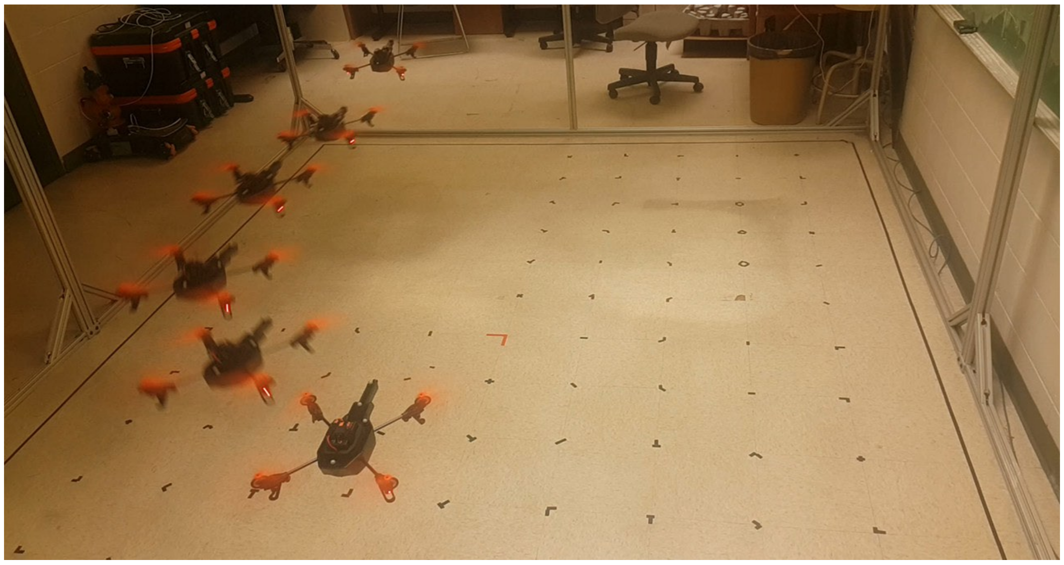
- take input commands from the operator to calculate $\mathbf{A}_{in}$,
- obtain the quadrotors position and orientation from the motion capture system to implement the position tracking controller for the quadrotor,
- receive the legacy navigation data from the quadrotor, and
- perform all necessary calculations to generate a reference angle $\mathbf{A}_{ref}$ and send it to the AR.Drone 2.0 quadrotor through a WIFI connection.

5.2 Teleoperation of a Quadrotor UAV in Static Environment

In this section, experimentation results for the teleoperation of a quadrotor UAV in an environment with static obstacles are presented. We found that the best way to express the results is to use the action sequence of images technique.

In all experiment figures, the white curve, the red arrow, and the blue arrow represent the path of the quadrotor UAV, operator's command direction, and current UAV's motion direction respectively. Since all the obstacles are virtual, we used rviz (3D visualization tool for ROS) package in ROS to help visualize the obstacles in the environment as well as the quadrotor position with respect to the virtual obstacles as shown Fig. 5.3 and Fig. 5.4. We performed two experiments which are similar to the simulation cases.

The first experiment is shown in Fig. 5.3. The quadrotor UAV is commanded to fly with an acute angle to the wall; it does not collide with the wall, as expected from the simulation in Fig. 4.4b, and moves parallel to the wall as shown in Fig. 5.3a.


(a) The quadrotor is moving with an acute angle towards the virtual wall on the left.

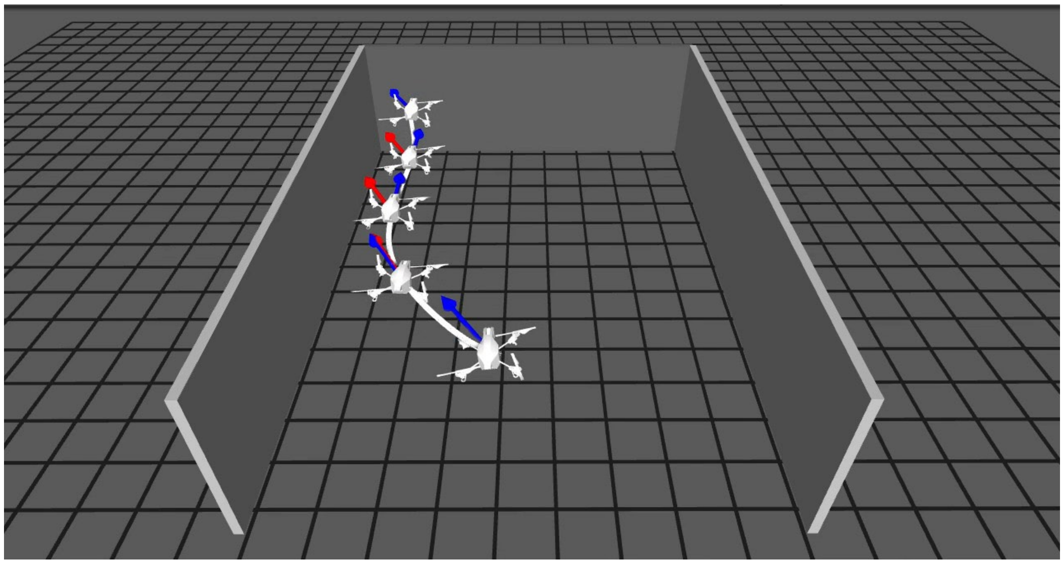
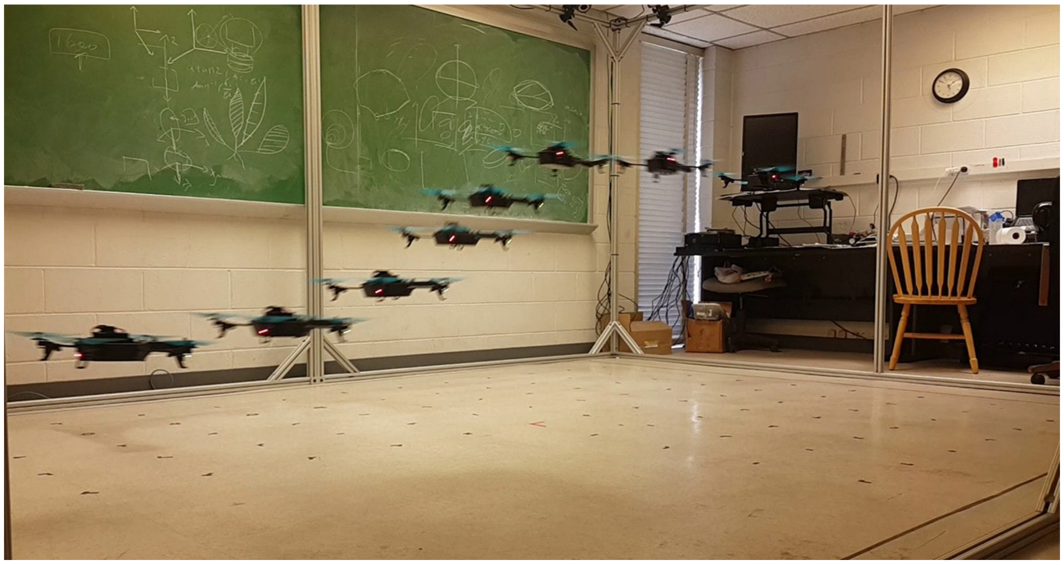
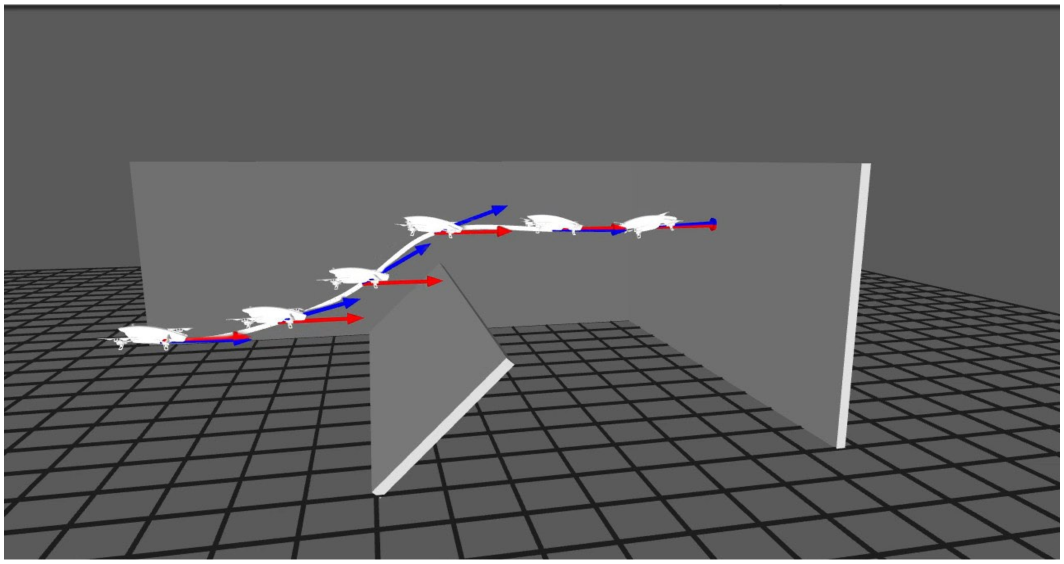

(b) Visualization of the above scene in rviz showing the virtual obstacles

Figure 5.3: The quadrotor is moving with an acute angle towards the wall on the left.

The virtual walls, UAV's path, UAV's current direction, and operator's command direction are shown in Fig. 5.3b.



(a) The quadrotor is moving towards a virtual sloped wall in $(y - z)$ plane.



(b) Visualization of the above scene in rviz showing the virtual obstacles

Figure 5.4: The quadrotor is moving towards a sloped wall in $(y - z)$ plane. The quadrotor avoided collision with sloped wall and stopped before the wall.

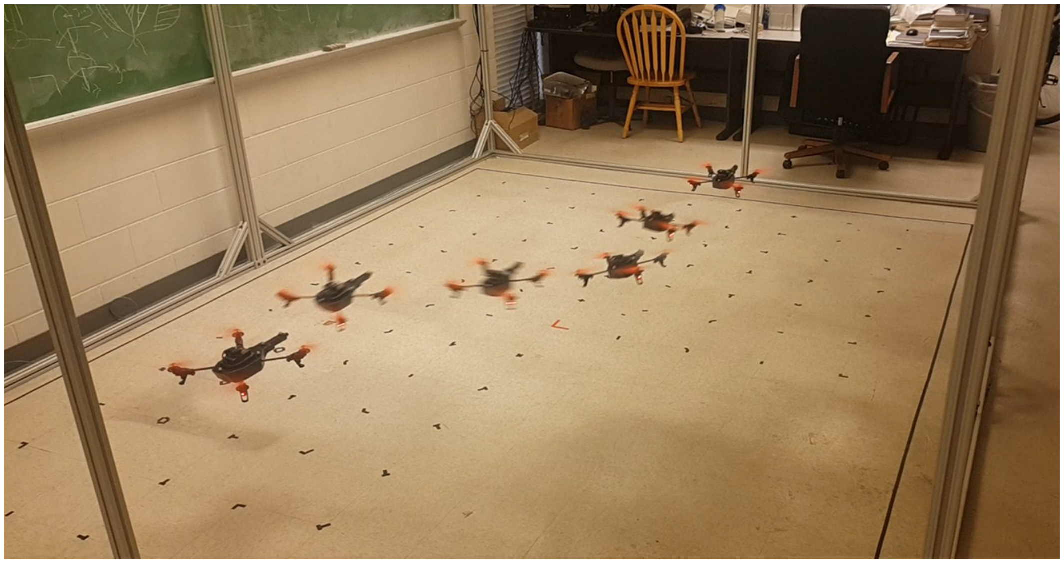
The second experiment is shown in Fig. 5.4. The quadrotor UAV is moving towards a sloped wall in $(y - z)$ plane; it deflects its path, as expected from the simulation in Fig. 4.6, and moves along the sloped wall, and then moves forward and stops before the wall as shown in Fig. 5.4a. The virtual walls, UAV's path, UAV's current direction, and operator's command direction are shown in Fig. 5.4b.

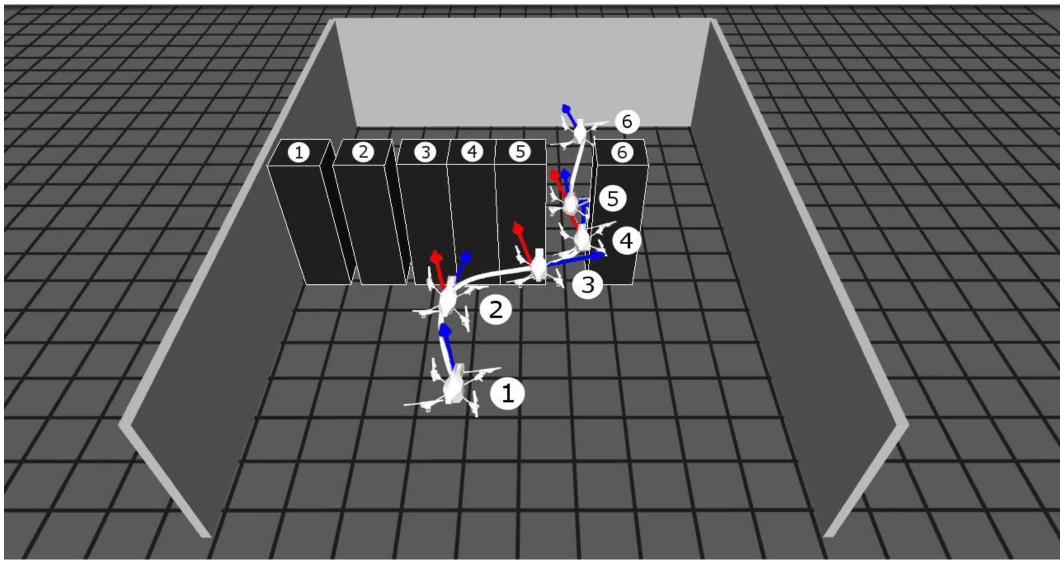
5.3 Teleoperation of a Quadrotor UAV in Dynamic Environment

Experimentation results for the teleoperation of a quadrotor UAV around moving objects are presented in this section. Note that, in all experiment figures presented in this section, when the quadrotor UAV is at position 1, the moving obstacle is at position 1 as well and when the quadrotor UAV is at position 2, the moving obstacle is at position 2 as well and so on. We performed three experiments which are similar to some extent to the simulation cases.

The first experiment is shown in Fig. 5.5. In Fig. 5.5a, the vehicle moves towards a horizontally moving obstacle (with a speed of 0.6 m/s). The vehicle avoids collision with the obstacle by detouring the obstacle for the reasons discussed in simulation case, Fig. 4.7. At position 5, the moving obstacle tries to collide with the quadrotor from its left side. The quadrotor, hence, moves to the right, and eventually, tracks operator command input. Fig. 5.5b shows the static and dynamic virtual obstacles, the path of the quadrotor, UAV's current direction, and operator's command direction.

The second experiment is shown in Fig. 5.6. In Fig. 5.6a, the vehicle moves towards a diagonally moving obstacle (with a speed of 0.5 m/s). The quadrotor does exactly the same as the simulation case shown in Fig. 4.8. the static and dynamic virtual obstacles, the path of the quadrotor, UAV's current direction, and operator's command direction are illustrated in Fig. 5.6b.

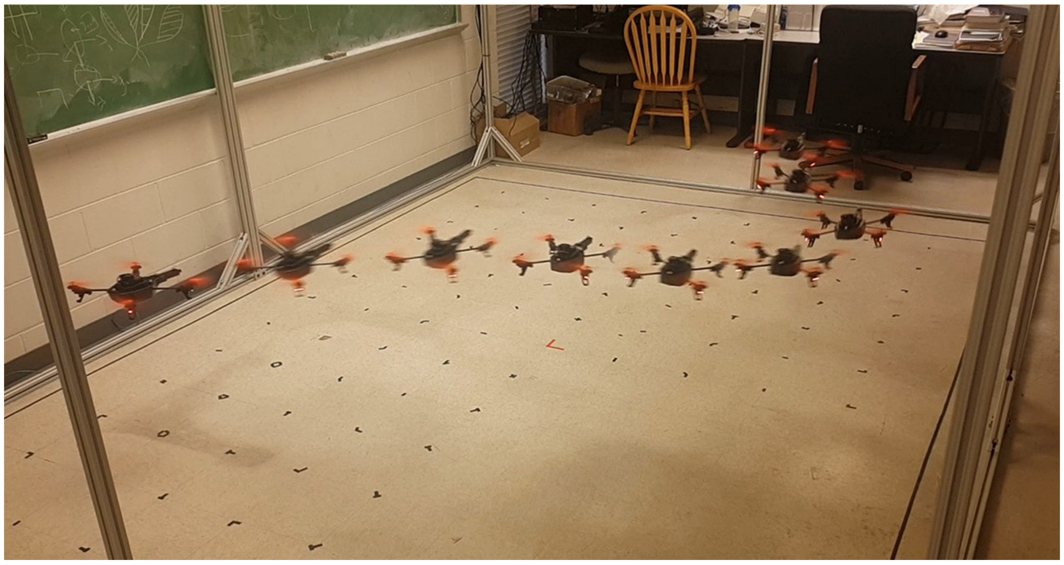

(a) The quadrotor is commanded to move towards a horizontally virtual moving obstacle.



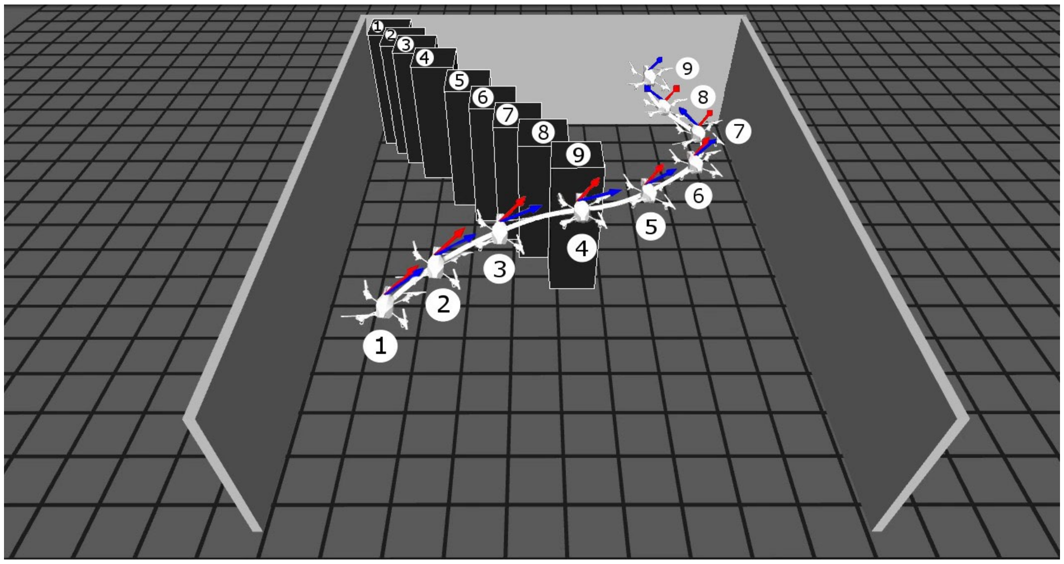
(b) Visualization of the above scene in rviz showing the virtual static and daynmic obstacles

Figure 5.5: The quadrotor is commanded to move towards a horizontally moving obstacle. The quadrotor UAV succeeded in bypassing the moving obstacle.

The third experiment is shown in Fig. 5.7. The quadrotor avoiding multiple moving obstacles (each with a speed of 0.5 m/s) is shown in Fig. 5.7a. Each obstacle moves back and forth from position1 to 6. This case is similar to the case shown



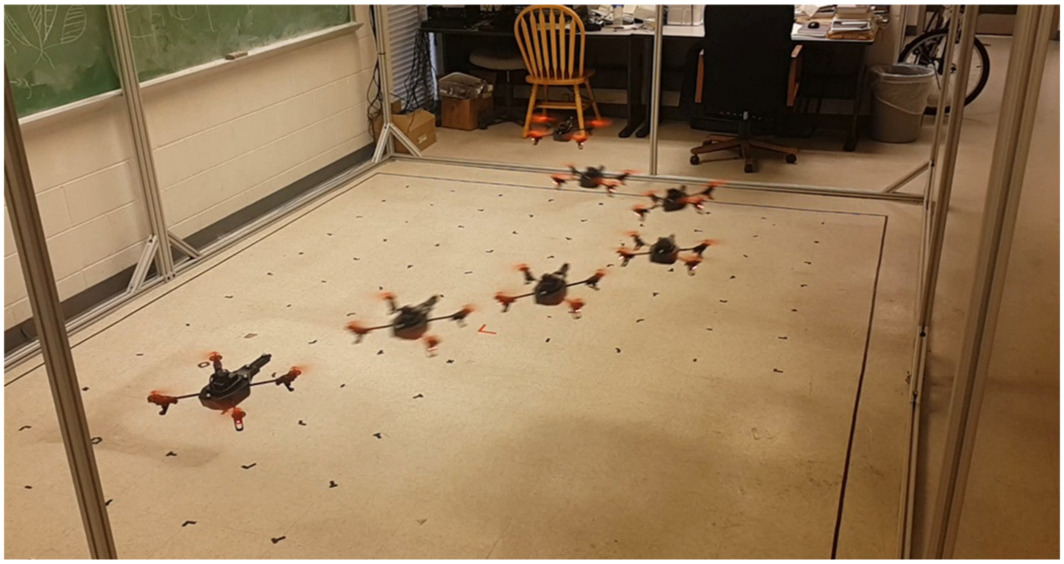
(a) The quadrotor is commanded to fly towards a diagonally virtual moving obstacle.



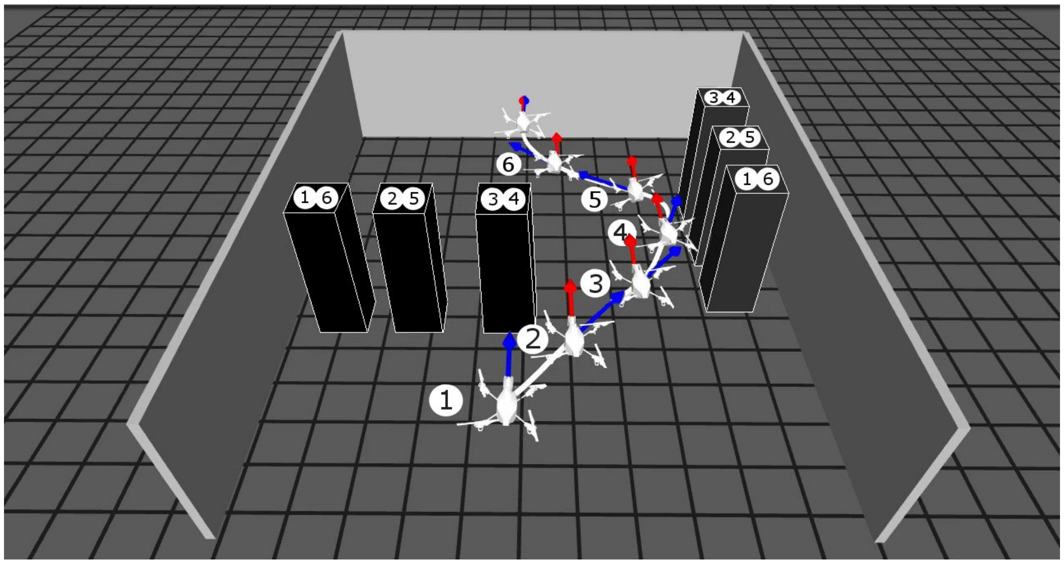
(b) Visualization of the above scene in rviz showing the virtual static and daynmic obstacles

Figure 5.6: The quadrotor is commanded to fly towards a diagonally moving obstacle. The quadrotor UAV avoided the collision with the moving obstacle and moved parallel to the wall.

in Fig. 4.9. Fig. 5.7b demonstrates the static and dynamic virtual obstacles, the path of the quadrotor, UAV's current direction, and operator's command direction.



(a) The quadrotor UAV is steered forward where there are two virtual moving obstacles in the environment.



(b) Visualization of the above scene in rviz showing the virtual static and daynmic obstacles

Figure 5.7: The quadrotor UAV is steered forward where there are two moving obstacles in the environment.

Chapter 6

Teleoperation Interface Program

In teleoperation, the operator drives a robot from a remote distance and visualizes the environment around the UAV via a camera mounted on the UAV. This camera usually has a limited field of view, and hence the operator cannot visualize the UAV's surrounding environment properly. This leads to a lack of situation awareness for the operator and decrease the efficiency and safety of the teleoperation process, i.e., collision might occur because the robot may move in a direction that is out of the camera's field of view, especially in the case of the quadrotor UAV which can move in any direction.

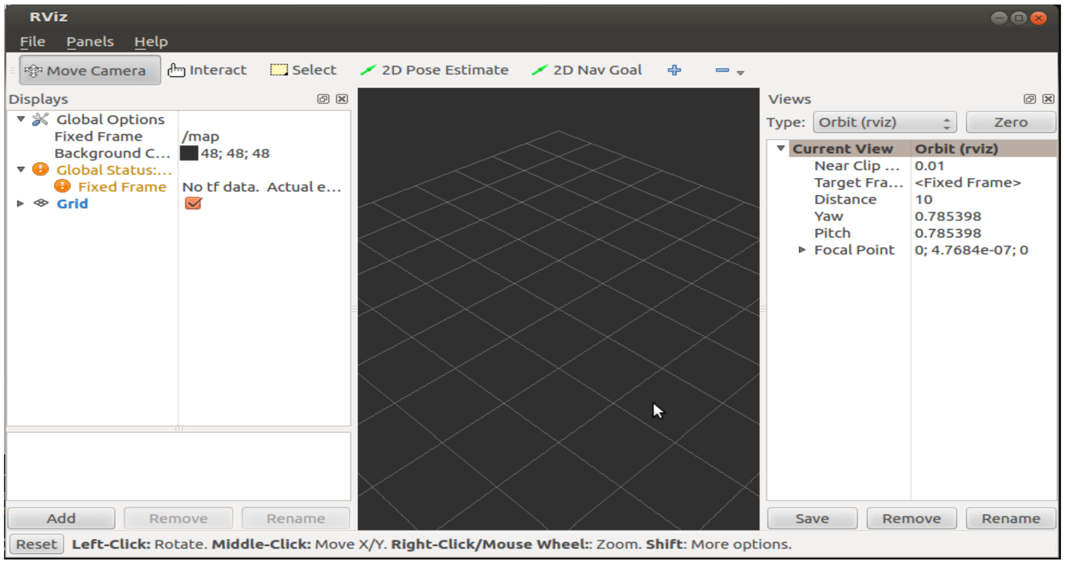
So far, we have developed an algorithm to assist the operator in autonomously avoiding collision with obstacles and to follow the operator's command as closely as possible. To further enhance the operator's situation awareness, the data, i.e., a point cloud, from the range sensor, e.g., vision sensor or LIDAR, can be used to visualize the surrounding environment, e.g., bulding a map.

In this chapter, we present a user interface program developed based on rviz package, a 3D visualization tool for ROS (Robot Operating System), to assist the operator during the teleoperation by building a 3D map of the environment or just visualizing the environment around the robot using the point cloud of measurements provided by the range sensor.

The user interface program was implemented in simulation only and not tested in the experiments using a physical quadrotor because the current quadrotor UAV does not have an on-board range sensor, and this work is currently in-progress to apply in real experiments.

6.1 Rviz

Rviz is a 3D visualization tool for ROS. It is a powerful ROS package that makes it easy to visualize point clouds, laser range sensor measurements, a video, etc. because of its many built-in display types. The main window of rviz when it starts is shown in Fig. 6.1



Global Options	
Fixed Frame	/map
Background C...	■ 48; 48; 48
Global Status:...	
Fixed Frame	No tf data. Actual e...
Grid	✓

Type: Orbit (rviz)

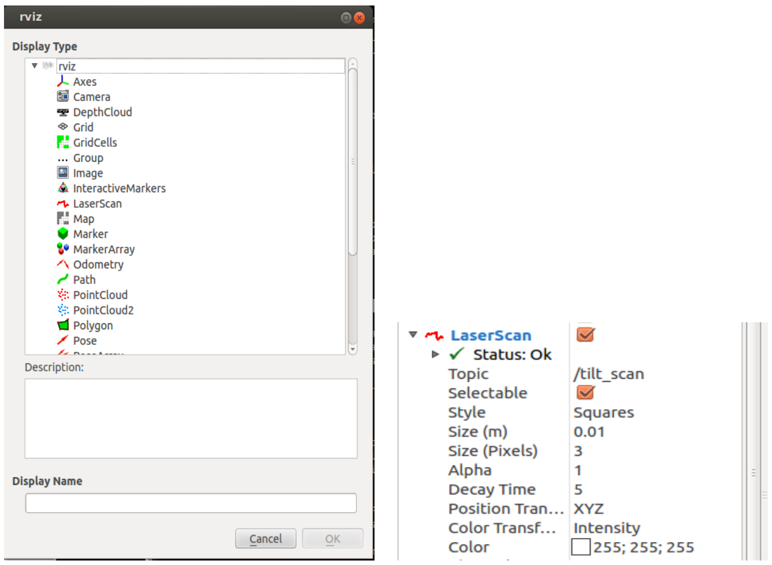
Current View	Orbit (rviz)
Near Clip ...	0.01
Target Fra...	<Fixed Frame>
Distance	10
Yaw	0.785398
Pitch	0.785398
Focal Point	0; 4.7684e-07; 0

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click/Mouse Wheel:**: Zoom. **Shift:** More options.

Figure 6.1: Rviz the 3D visualization tool for ROS [4].

When the Add button, on the left side Fig. 6.1, is clicked, A new display will appear with many display types as shown in Fig. 6.2a. Once the display is added, it will appear on the display panel of the main rviz window. Each display has many properties (see Fig. 6.2b) such as colors, size, the topic to subscribe to, etc. Besides the built-in display types, one can add its own customized display through plugins.

Each added display will appear in the 3D view in the middle of the main rviz window. One can interact with the 3D view using the mouse, e.g., zoom, rotate, etc., or from the view panel on right side of the main rviz window.



rviz
- Axes
- Camera
- DepthCloud
- Grid
- GridCells
- Group
- Image
- InteractiveMarkers
- LaserScan
- Map
- Marker
- MarkerArray
- Odometry
- Path
- PointCloud
- PointCloud2
- Polygon
- Pose

LaserScan	✓
Status: Ok	
Topic	/tilt_scan
Selectable	✓
Style	Squares
Size (m)	0.01
Size (Pixels)	3
Alpha	1
Decay Time	5
Position Tran...	XYZ
Color Transf...	Intensity
Color	255; 255; 255

(a) Display types. (b) Display properties

Figure 6.2: Rviz display types and properties [4].

6.2 3D Map Building

Building a map is very important for mobile robots, especially in teleoperation tasks because of its various advantages. For example, it increases the situation awareness for the operator which leads to easy and efficient teleportation; it can be used for robot localization in simultaneous localization and mapping (SLAM) process in places where there is no GPS signal; it is beneficial in collision avoidance process where the distance to obstacles can be estimated from the built map.

The map building process is tested through simulation, and an octomap ROS package [42] is used to generate the 3D map of the environment and display it using rviz. The diagram for the 3D map building using v-rep and ROS is shown in Fig. 6.3. As can be seen from the figure, the point cloud from the range sensor and video are published as a ROS topics by v-rep. The octomap package then subscribes to the point cloud topic to generate the map and publish it. Finally, rviz subscribes to the map and video topics and displays them for the operator.

3D map building process using the above discussed method was tested in three different scenes. The first scene is shown in Fig. 6.4 where the quadrotor UAV is moving in a corridor and facing a protruded obstacle from the right wall. The top figure (Fig. 6.4a) is the simulation scene while the three figures (Fig. 6.4b, Fig. 6.4c and Fig. 6.4d) in the middle demonstrate the stages of the map building. The bottom figure (Fig. 6.4e) shows the complete 3D map of the scene. The other two simulation scenes are shown in Fig. 6.5 and Fig. 6.6. It should be noted that the 3D map building is suitable for a static environment. In a dynamic environment, it is better to just visualize the point cloud of the range sensor measurements as discussed in the next section.

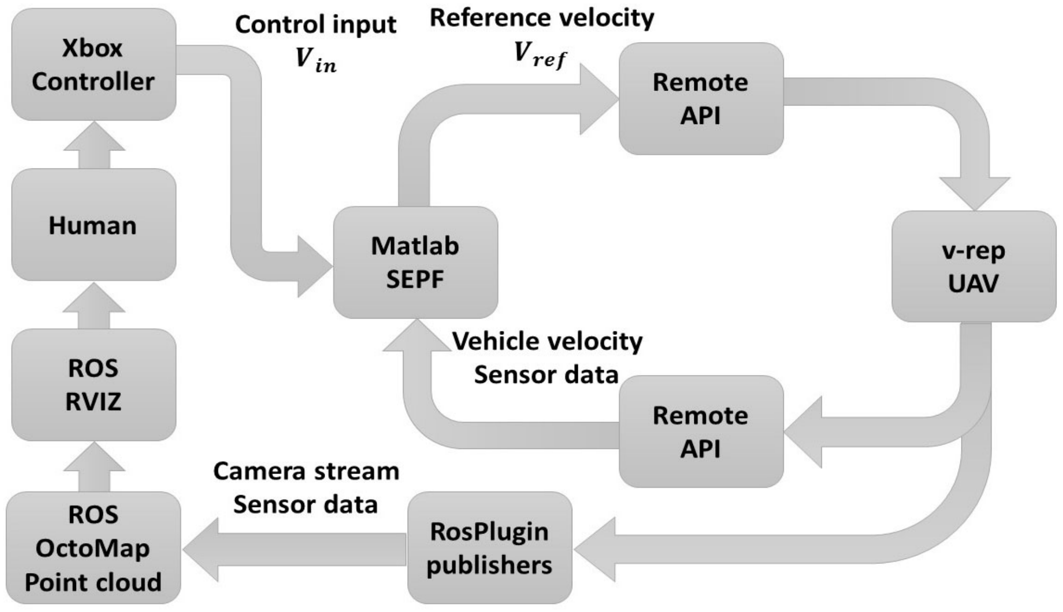
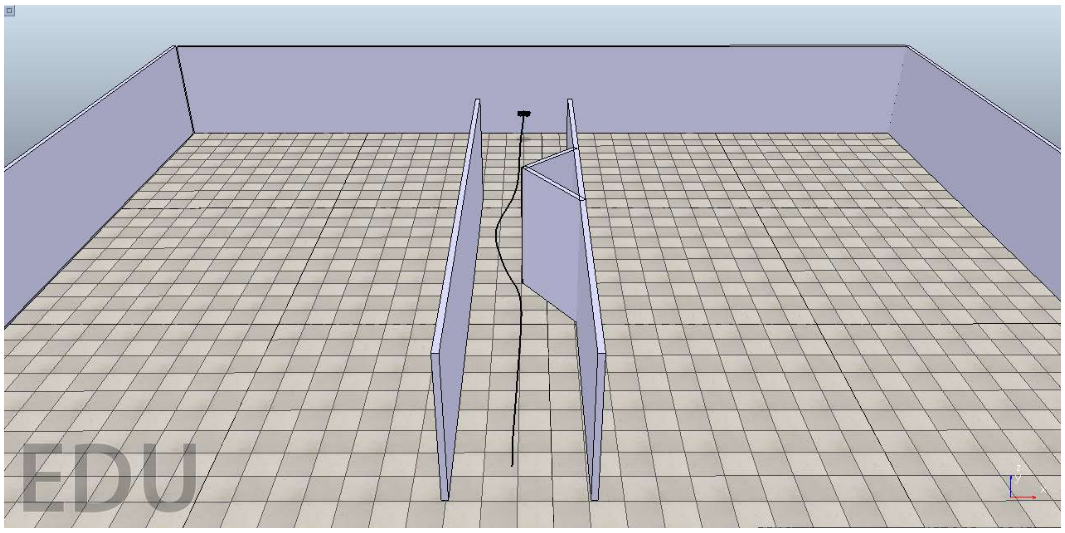
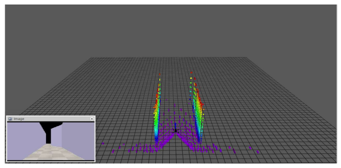


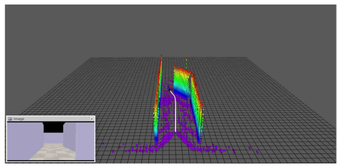
Figure 6.3: Diagram for 3D map building using v-rep and ROS.



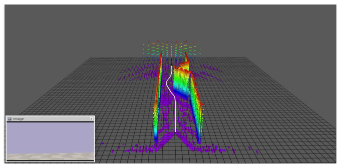
(a) The simulation scene for the case shown below



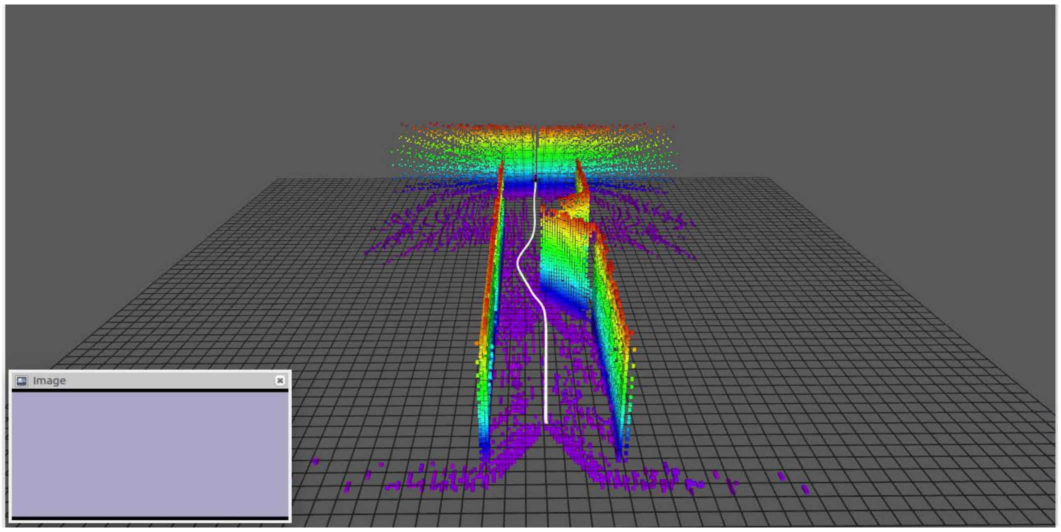
(b)



(c)

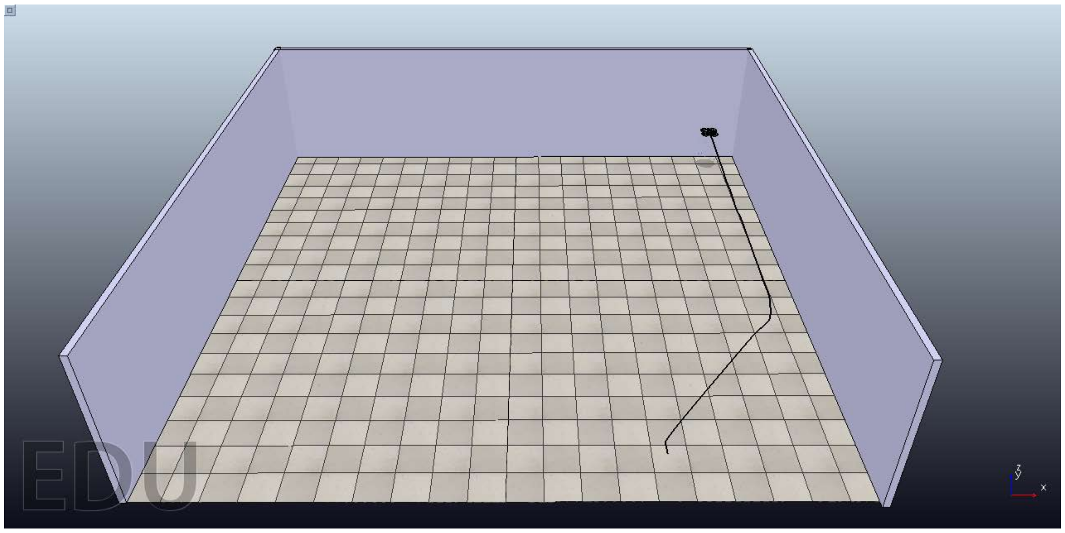


(d)

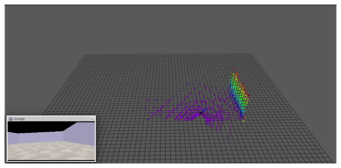


(e) The UAV stopped before the wall.

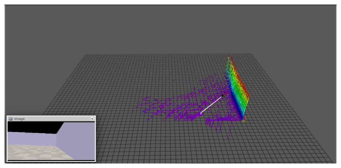
Figure 6.4: 3D map building for the case where the quadrotor UAV is moving in a passage and facing a protruded obstacle from the right wall.



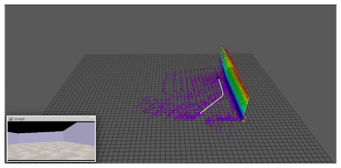
(a) The simulation scene for the case shown below



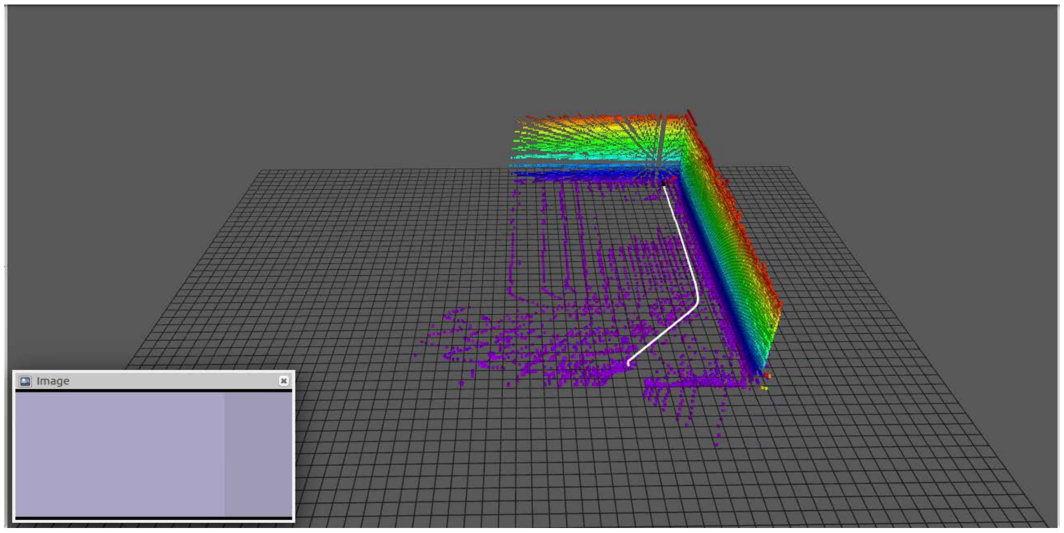
(b)



(c)

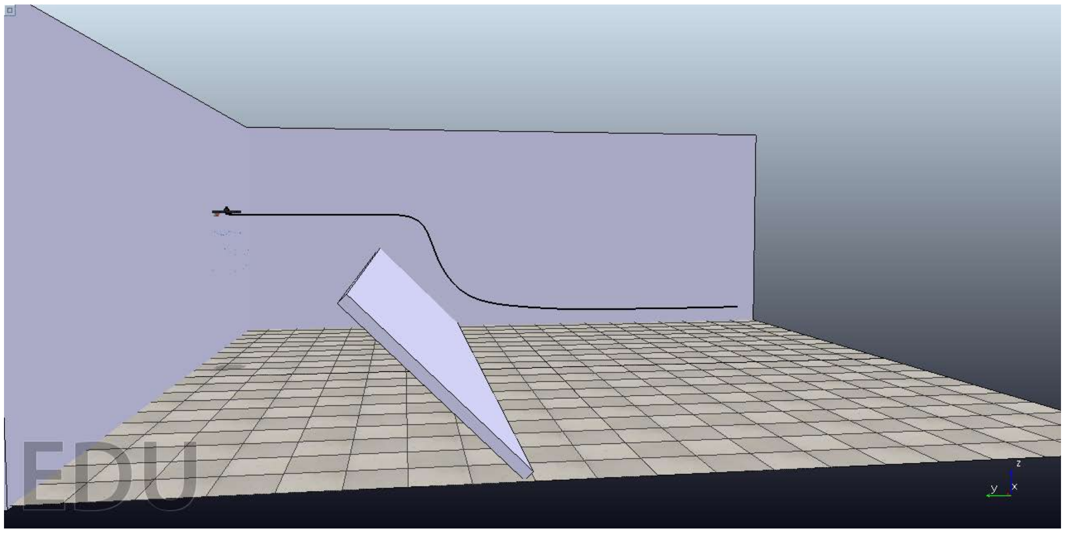


(d)

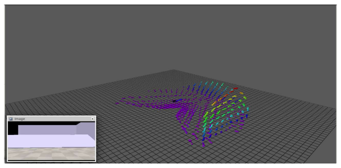


(e) The UAV stopped at the corner

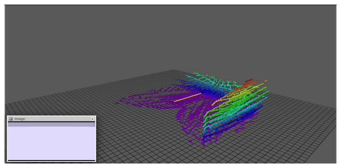
Figure 6.5: 3D map building for the case where the UAV is steered with an acute angle toward a wall on the right side.



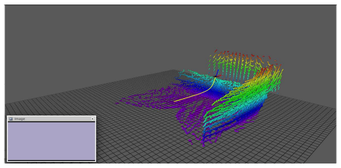
(a) The simulation scene for the case shown below



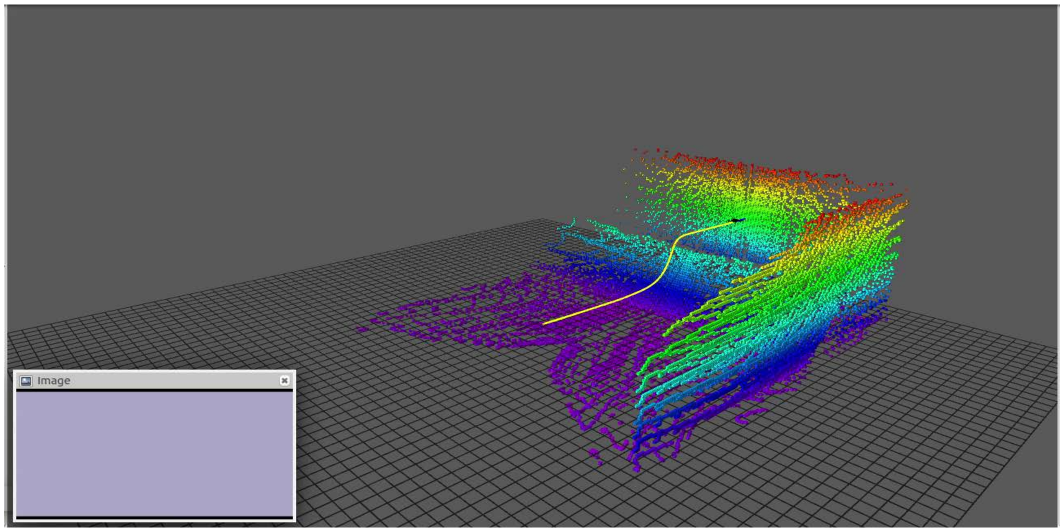
(b)



(c)



(d)



(e) The UAV stopped before the wall

Figure 6.6: 3D map building for the case where the UAV is moving toward a sloped wall in $(y - z)$ plane.

6.3 Point Cloud Visualization

Sometimes map building is computationally expensive or even misleading, particularly in a dynamic environment where objects are moving, and they will be mapped on several locations on the map which is undesirable. To address these problems, the point cloud from the distance sensor can be visualized online without any map building. In this case, the operator can visualize the environment around the robot and can recognize if there are moving obstacles. The point cloud visualization for the scenes in Fig. 6.4a, Fig. 6.5a and Fig. 6.6a are shown in Fig. 6.7, Fig. 6.8 and Fig. 6.9 respectively. Fig. 6.10 demonstrates the case of moving obstacles where the operator can easily distinguish the moving obstacle from the static objects even if it is not seen by the front camera as can be seen in Fig. 6.10b, Fig. 6.10c and Fig. 6.10d. The diagram for the point cloud visualization using v-rep and ROS is shown in Fig. 6.3.

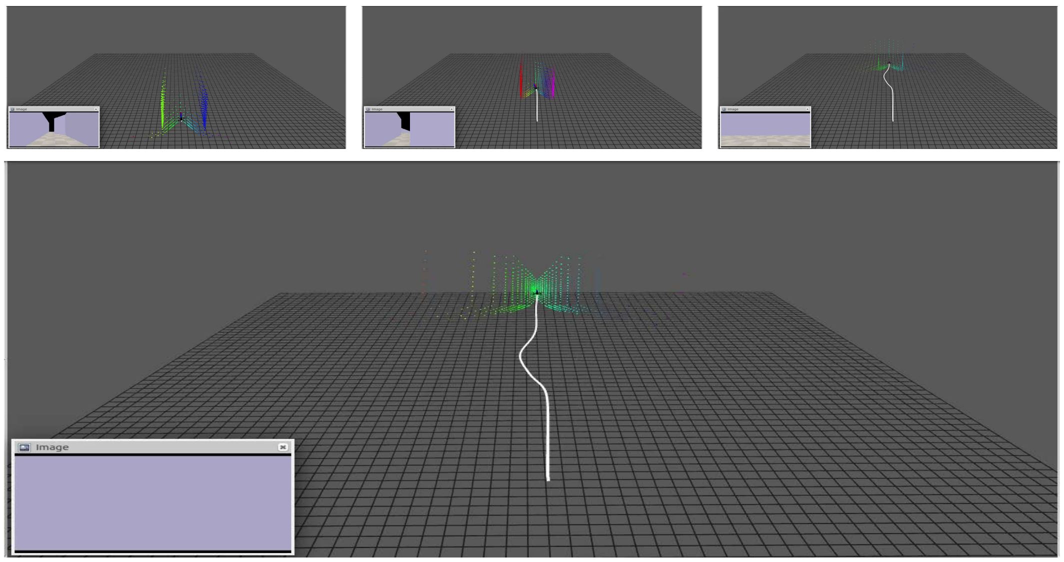


Figure 6.7: Point cloud visualization for the case where the quadrotor UAV is moving in a passage and facing a protruded obstacle from the right wall.

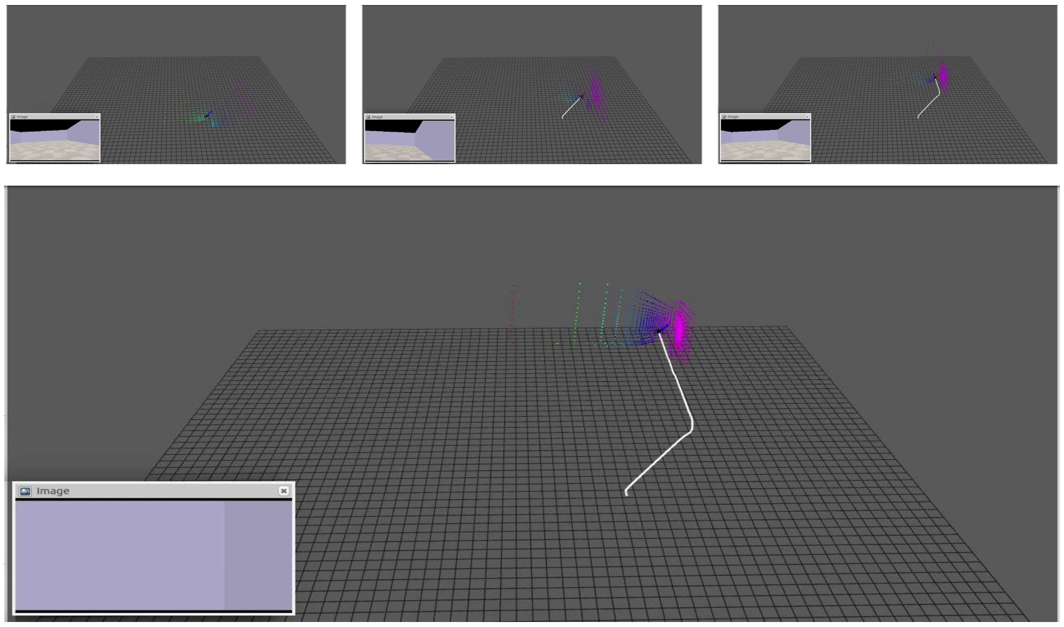


Figure 6.8: Point cloud visualization for the case where the UAV is steered with an acute angle toward a wall on the right side.

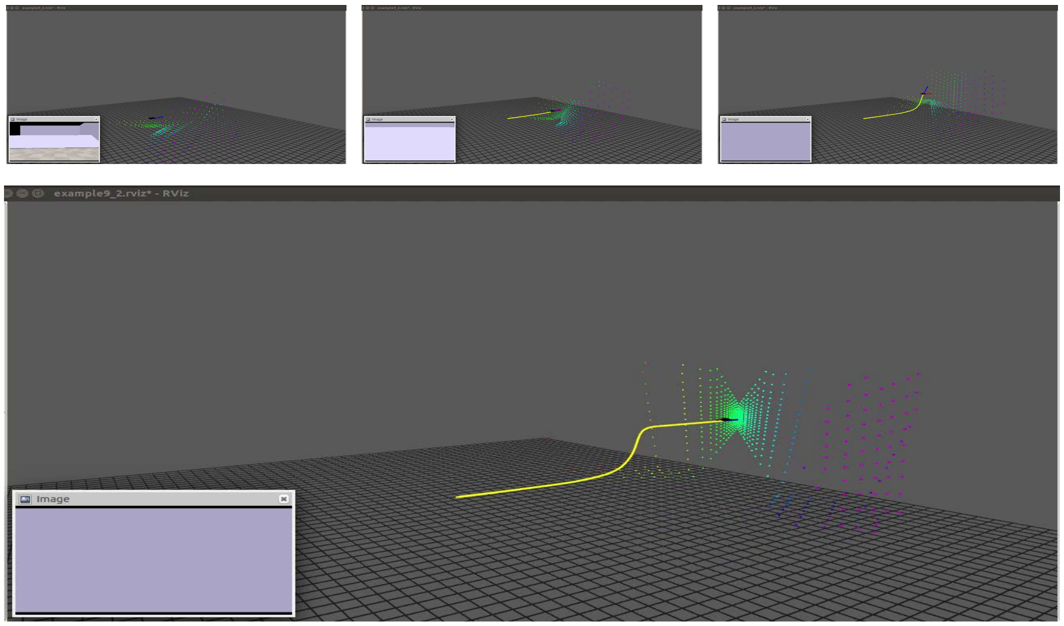
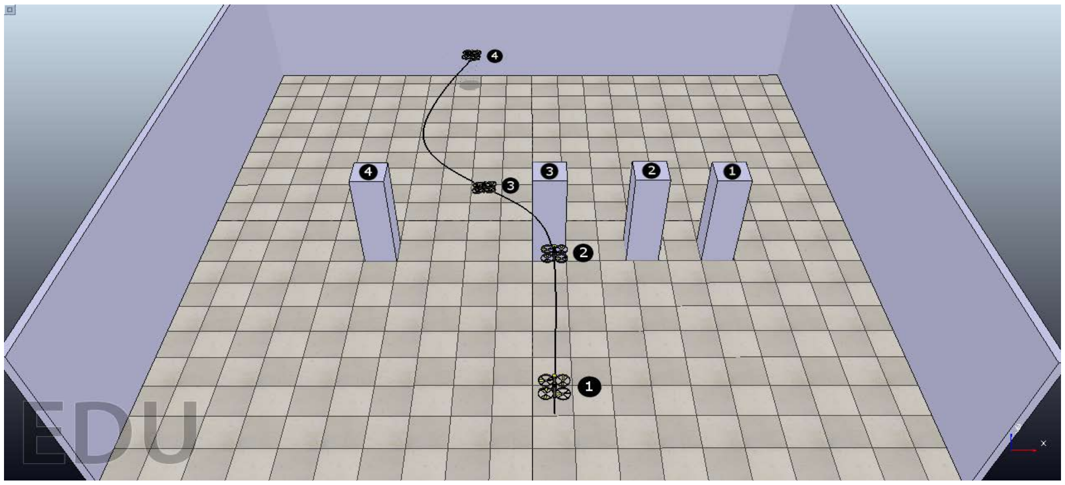
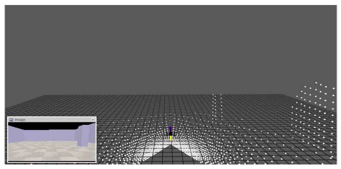


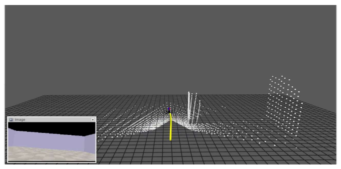
Figure 6.9: Point cloud visualization for the case where the UAV is moving toward a sloped wall in $(y - z)$ plane.



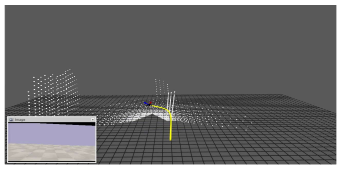
(a) The simulation scene for the moving obstacle case shown below. When the UAV is at position 1 the moving obstacle is at position 1 as well and so on.



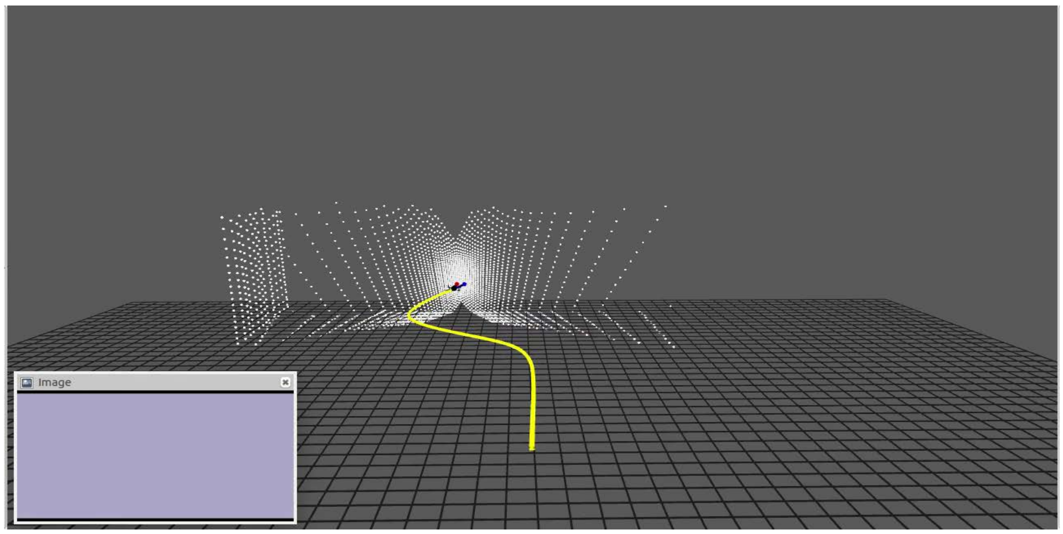
(b)



(c)



(d)



(e) The quadrotor stopped before the wall

Figure 6.10: Point cloud visualization for the case of moving obstacles in which the operator can distinguish the moving obstacles from the static ones.

Chapter 7

Conclusions and Future Research

This thesis proposed a 3-dimensional autonomous collision avoidance algorithm in static and dynamic environments based on the idea of super-ellipsoidal potential function (SEPF.) In the design of the SEPF, we have a full control over the shape and size of the potential function. In particular, we can adjust the length, width, height, and the amount of flattening at the tips of the potential function so that the collision avoidance motion vector generated from the potential function can be adjusted accordingly. The proposed method is computationally inexpensive and requires the relative distance to the obstacle in order to work in the case of static obstacles and the relative distance and velocity to the obstacle in the case of moving obstacles.

In the proposed algorithm, operator's commands that are only in the direction of the obstacle are overridden and others that are in the obstacle-free path are tracked. In this way, our algorithm ensures that: i) the UAV autonomously avoids obstacles in its path; ii) the UAV chooses the obstacle-free path if there is a control input in that path direction; and iii) the operator is always in control of the vehicle. Thus, our algorithm enables easy and safe teleoperation.

The results show the effectiveness of the proposed algorithm where the algorithm was first validated using the v-rep simulation program in conjunction with Matlab program. Then, it was validated using a real quadrotor UAV, AR.Drone 2.0 and a motion capture system. In all simulation and experiment cases, the operator failed to collide the quadrotor UAV with the obstacles and the quadrotor UAV tracked the operator's command as closely as possible.

Recommendations for future research include:

- **On-board Collision Avoidance:** Since the ultimate goal of this research is to assist the UAV operator in real life situations and not just inside a laboratory environment where the obstacles are programmed and the location of the UAV is known using the motion capture system, we need to eliminate the dependency on the motion capture system by performing the following in order to enable the proposed method to work outside the motion capture system:

  - **On-board Sensing:** Even though we implemented our method on a physical quadrotor UAV, the obstacles were virtual and programmed for each experiment and a motion capture system was used to obtain the UAV position. In an unknown real world environment, an on-board 3D range sensor is needed so that the obstacles can be detected. This can be achieved by using a method similar to [25] or using vision sensors that provide point clouds of measurements.

  - **Position Tracking Controller Using On-board Sensors Only:** Currently, we are using the data provided by the motion capture system to implement a position tracking controller. If we go outside the motion capture system, we need a method to implement a position tracking controller using on-board sensors only, especially in indoor environments where there is no GPS signal.

- **Haptic Feedback:** The use haptic feedback in conjunction with the proposed method is beneficial because it increases the operator's situation awareness.
- **Multiple UAVs:** It is worthy to test the proposed algorithm in multiple UAVs teleoperation.

Bibliography

[1] H. Bolandi, M. Rezaei, R. Mohsenipour, H. Nemati, and S. M. Smailzadeh, "Attitude control of a quadrotor with optimized pid controller," 2013.

[2] G.Niemeyer, C. Preusche, and G. Hirzinger, "Telerobotics," in *Springer handbook of robotics*. Springer, 2008, pp. 741–757.

[3] Parrot, "gallary for ar.drone 2.0," 2016. [Online]. Available: http://www.parrot.com/usa/gallery/ardrone2/?page=2

[4] Rviz, "Rviz users guide," 2016. [Online]. Available: http://docs.ros.org/jade/api/rviz/html/user_guide

[5] S. R. B. dos Santos, C. L. Nascimento, and S. N. Givigi, "Design of attitude and path tracking controllers for quad-rotor robots using reinforcement learning," in *Aerospace Conference, 2012 IEEE*, March 2012, pp. 1–16.

[6] R. He, S. Prentice, and N. Roy, "Planning in information space for a quadrotor helicopter in a gps-denied environment," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, May 2008, pp. 1814–1820.

[7] L. Garcia-Delgado, A. Dzul, V. Santibanez, and M. Llama, "Quad-rotors formation based on potential functions with obstacle avoidance," *IET Control Theory Applications*, vol. 6, no. 12, pp. 1787–1802, Aug 2012.

[8] H. Bai, S. Shao, and H. Wang, "A vtol quadrotor platform for multi-uav path planning," in *Electronic and Mechanical Engineering and Information Technology (EMEIT), 2011 International Conference on*, vol. 6, Aug 2011, pp. 3079–3081.

[9] C. Masone, A. Franchi, H. H. Blthoff, and P. R. Giordano, "Interactive planning of persistent trajectories for human-assisted navigation of mobile robots," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2012, pp. 2641–2648.

[10] T. M. Lam, H. W. Boschloo, M. Mulder, and M. M. van Paassen, "Artificial force field for haptic feedback in uav teleoperation," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 39, no. 6, pp. 1316–1330, Nov 2009.

[11] N. Diolaiti and C. Melchiorri, "Teleoperation of a mobile robot through haptic feedback," in *Haptic Virtual Environments and Their Applications, IEEE International Workshop 2002 HAVE*, 2002, pp. 67–72.

[12] M. Qasim and K. D. Kim, "Super-ellipsoidal potential function for autonomous collision avoidance of a teleoperated uav," *World Academy of Science, Engineering and Technology, International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering*, vol. 10, no. 1, pp. 164–169, 2016.

[13] K. Dalamagkidis, "Classification of uavs," in *Handbook of Unmanned Aerial Vehicles*. Springer, 2015, pp. 83–91.

[14] M. Arjomandi, S. A. andM. MammoneandM. Nelson, and T. Zhou, "Classification of unmanned aerial vehicles," *Report for Mechanical Engineering class, University of Adelaide, Adelaide, Australia*, 2006.

[15] J. Israelsen, M. Beall, D. Bareiss, D. Stuart, E. Keeney, and J. van den Berg, "Automatic collision avoidance for manually tele-operated unmanned aerial vehicles," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 6638–6643.

[16] J. Cui, S. Tosunoglu, R. Roberts, C. Moore, and D. W. Repperger, "A review of teleoperation system control," in *Proceedings of the Florida Conference on Recent Advances in Robotics*. Florida Atlantic University Boca Raton, FL, 2003.

[17] M. Ferre, R. Araci, C. Balaguer, M. Buss, and C. Melchiorri, *Advances in telerobotics*. Springer, 2007, vol. 31.

[18] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The international journal of robotics research*, vol. 5, no. 1, pp. 90–98, 1986.

[19] K. M. Choset, *Principles of robot motion: theory, algorithms, and implementation*. MIT press, 2005.

[20] J. O. Kim and P. K. Khosla, "Real-time obstacle avoidance using harmonic potential functions," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 3, pp. 338–349, Jun 1992.

[21] D. H. Kim and S. Shin, "Local path planning using a new artificial potential function composition and its analytical design guidelines," *Advanced Robotics*, vol. 20, no. 1, pp. 115–135, 2006.

[22] S. S. Ge. and Y. C. J, "Dynamic motion planning for mobile robots using potential field method," *Autonomous Robots*, vol. 13, no. 3, pp. 207–222, 2002.

[23] P. Stegagno, M. Basile, H. H. Blthoff, and A. Franchi, "A semi-autonomous uav platform for indoor remote operation with visual and haptic feedback," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 3862–3869.

[24] F. Rehmatullah and J. Kelly, "Vision-based collision avoidance for personal aerial vehicles using dynamic potential fields," in *Computer and Robot Vision (CRV), 2015 12th Conference on*, June 2015, pp. 297–304.

[25] M. Nieuwenhuisen, D. Droeschel, J. Schneider, D. Holz, T. Lbe, and S. Behnke, "Multimodal obstacle detection and collision avoidance for micro aerial vehicles," in *Mobile Robots (ECMR), 2013 European Conference on*, Sept 2013, pp. 7–12.

[26] C. Masone, P. R. Giordano, H. H. Blthoff, and A. Franchi, "Semi-autonomous trajectory generation for mobile robots with integral haptic shared control," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 6468–6475.

[27] S. Stramigioli, R. Mahony, and P. Corke, "A novel approach to haptic teleoperation of aerial robot vehicles," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, May 2010, pp. 5302–5308.

[28] A. Y. Mersha, S. Stramigioli, and R. Carloni, "Switching-based mapping and control for haptic teleoperation of aerial robots," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2012, pp. 2629–2634.

[29] S. Omari, M. D. Hua, G. Ducard, and T. Hamel, "Bilateral haptic teleoperation of vtol uavs," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, May 2013, pp. 2393–2399.

[30] D. Lee, A. Franchi, H. I. Son, C. Ha, H. H. Blthoff, and P. R. Giordano, "Semi-autonomous haptic teleoperation control architecture of multiple unmanned aerial vehicles," *IEEE/ASME Transactions on Mechatronics*, vol. 18, no. 4, pp. 1334–1345, Aug 2013.

[31] H. I. Son, A. Franchi, L. L. Chuang, J. Kim, H. H. Bulthoff, and P. R. Giordano, "Human-centered design and evaluation of haptic cueing for teleoperation of multiple mobile robots," *IEEE Transactions on Cybernetics*, vol. 43, no. 2, pp. 597–609, April 2013.

[32] E. J. Rodriguez-Seda, J. J. Troy, C. A. Erignac, P. Murray, D. M. Stipanovic, and M. W. Spong, "Bilateral teleoperation of multiple mobile agents: Coordinated motion and collision avoidance," *IEEE Transactions on Control Systems Technology*, vol. 18, no. 4, pp. 984–992, July 2010.

[33] A. M. Brandt and M. B. Colton, "Haptic collision avoidance for a remotely operated quadrotor uav in indoor environments," in *Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on*, Oct 2010, pp. 2724–2731.

[34] J. Mendes and R. Ventura, "Safe teleoperation of a quadrotor using fastslam," in *Safety, Security, and Rescue Robotics (SSRR), 2012 IEEE International Symposium on*, Nov 2012, pp. 1–6.

[35] D. Bareiss, J. van den Berg, and K. K. Leang, "Stochastic automatic collision avoidance for tele-operated unmanned aerial vehicles," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, Sept 2015, pp. 4818–4825.

[36] C. Fulgenzi, A. Spalanzani, and C. Laugier, "Probabilistic rapidly-exploring random trees for autonomous navigation among moving obstacles," in *Workshop on safe navigation, IEEE International Conference on Robotics and Automation (ICRA)*. Citeseer, 2009.

[37] J. D. Lawrence, *A catalog of special plane curves*. Courier Corporation, 2013.

[38] E. W. Weisstein, "Ellipsoid," *moment*, vol. 30, p. 31, 2005.

[39] E. Rohmer, S. P. N. Singh, and M. Freese, "V-rep: A versatile and scalable robot simulation framework," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov 2013, pp. 1321–1326.

[40] V-rep, "virtual robot experimentation platform," 2016. [Online]. Available: http://www.coppeliarobotics.com

[41] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and N. Andrew, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

[42] A. Hornung, K. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, “Octomap: An efficient probabilistic 3d mapping framework based on octrees,” *Autonomous Robots*, vol. 34, no. 3, pp. 189–206, 2013.