EXAMENSARBETE INOM DATALOGI OCH DATATEKNIK, AVANCERAD NIVÅ, 30 HP

STOCKHOLM, *SVERIGE 2018*

Safe Navigation of a Tele-operated Unmanned Aerial Vehicle

DANIEL DUBERG

KTH
SKOLAN FÖR DATAVETENSKAP OCH KOMMUNIKATION

Safe Navigation of a Tele-operated Unmanned Aerial Vehicle

DANIEL DUBERG

Master in Computer Science
Date: January 22, 2018
Supervisor: Patric Jensfelt
Examiner: Joakim Gustafson
Swedish title: Säker teleoperativ navigering av en obemannad luftfarkost
School of Computer Science and Communication

Abstract

Unmanned Aerial Vehicles (UAVs) can navigate in indoor environments and through environments that are hazardous or hard to reach for humans. This makes them suitable for use in search and rescue missions and by emergency response and law enforcement to increase situational awareness. However, even for an experienced UAV tele-operator controlling the UAV in these situations without colliding into obstacles is a demanding and difficult task.

This thesis presents a human-UAV interface along with a collision avoidance method, both optimized for a human tele-operator. The objective is to simplify the task of navigating a UAV in indoor environments. Evaluation of the system is done by testing it against a number of use cases and a user study. The results of this thesis is a collision avoidance method that is successful in protecting the UAV from obstacles while at the same time acknowledges the operator's intentions.

Sammanfattning

Obemannad luftfarkoster (UAV:er) kan navigera i inomhusmiljöer och genom miljöer som är farliga eller svåra att nå för människor. Detta gör dem lämpliga för användning i sök- och räddningsuppdrag och av akutmottagning och rättsväsende genom ökad situationsmedvetenhet. Dock är det även för en erfaren UAV-teleoperatör krävande och svårt att kontrollera en UAV i dessa situationer utan att kollidera med hinder.

Denna avhandling presenterar ett människa-UAV-gränssnitt tillsammans med en kollisionsundvikande metod, båda optimerade för en mänsklig teleoperatör. Målet är att förenkla uppgiften att navigera en UAV i inomhusmiljöer. Utvärdering av systemet görs genom att testa det mot ett antal användningsfall och en användarstudie. Resultatet av denna avhandling är en kollisionsundvikande metod som lyckas skydda UAV från hinder och samtidigt tar hänsyn till operatörens avsikter.

Contents

Chapter 1

Introduction

Unmanned aerial vehicles (UAVs) are getting increasingly more popular in the civil, military, and commercial sector. They are being used for a range of different applications, such as precision farming [72], protecting wildlife [58], mapping [53], road traffic monitoring [30], and environmental monitoring [26].

With their high manoeuvrability and potential for a small form factor, UAVs can navigate in indoor environments and through environments that are hazardous or hard to reach for humans. These abilities make UAVs suitable for use in search and rescue missions and by emergency response and law enforcement to increase situational awareness [67, 23, 33]. However, even for an experienced UAV teleoperator, which is someone who remotely controls the UAV from a distance without necessary having direct line of sight of the UAV, controlling the UAV in these situations without colliding into obstacles is a demanding and difficult task. Tele-operation in these situations is difficult even for, simpler to control, ground vehicles as demonstrated from the search and rescue mission at the World Trade Center [11].

Tele-operating a UAV in an indoor environment, such as a collapsed building, is challenging for several reasons. There are obstacles close in all directions, which means that the operator needs to pay attention to not collide into any of the obstacles. Situational awareness is another challenge for the operator, since the operator usually only sees through a forward looking camera. This makes it difficult for the operator to know what is to the sides of the UAV and how to move the UAV to be able to go from one place to another. It would be advantageous if the operator only had to focus on going from point A to point

B without having to worrying about colliding into obstacles.

1.1 Problem Statement

This thesis aims to answer the following question: how to design a system for a UAV such that an operator is aware of the surroundings and can safely and quickly navigate from one position to another in an indoor environment without being in line of sight with the UAV and without training?

1.2 Scope and Limitations

For this project the UAV is limited to three degrees of freedom: forward/back, left/right, and rotation around the vertical axis. It is therefore assumed that the UAV can maintain a desired altitude. The presence of a high level control system is another assumption. This means that the UAV is controlled by the operator by giving it a direction, velocity, and a rotation around the vertical axis instead of directly controlling the roll, pitch, and yaw.

This thesis is restricted to holonomic UAVs, meaning that it has control over all of its degrees of freedom, that can hold its position in the air, such as: multirotor UAVs and helicopters. We focus on the design of the collision avoidance method and the human-UAV interface by having real-world constraints and conditions in mind when the experiments are performed in the simulator.

1.3 Outline

The remainder of this report is organized as follows. In Chapter 2 related work in the area of collision avoidance and human-robot interfaces is presented. Based on the information and ideas in Chapter 2, a complete system that assists UAV tele-operators is presented in Chapter 3. The system is tested against a number of use cases and a user study is performed in a simulated environment which the conditions for is laid out in Chapter 4. In Chapter 5, the results from the experiments are presented. The results are then discussed in Chapter 6. Lastly, Chapter 7 states the conclusions that can be derived and

what can be expanded upon in future work.

Chapter 2

Related Work

In this chapter, different collision avoidance methods will be presented together with ideas and research on how to design a user interface for tele-operating a robot.

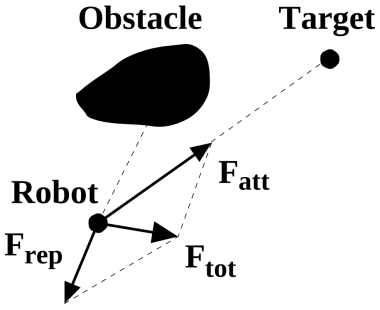
2.1 Collision Avoidance

Collision avoidance is a well researched area. In this section, some of the most popular and noteworthy collision avoidance methods will be presented and their strengths and weaknesses will be explained. Most of the methods covered in this section does not take into account the kinematic and dynamic constraints of the robot, therefore towards the end of this section research that address this will be presented.

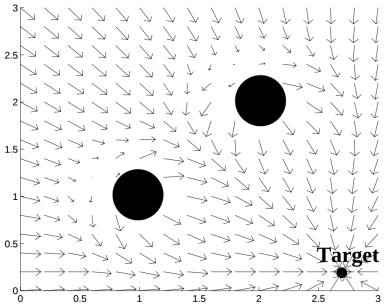
Early on in robotics, collision avoidance was regarded as a high level path planning problem. This means that the path planner generates a path that is completely collision free and then a low level control system executes the path. The low level control system's only concern is to follow the generated path. This approach has a number of drawbacks. One drawback is that it is computationally heavy to compute a collision free path in a cluttered environment. Another drawback is that it makes the robot not able to react to changes in the environment.

2.1.1 Potential Field Methods

Potential field methods (PFMs) [29] places the robot in a vector field of artificial forces, where obstacles are repelling forces, F_{rep}, and the desired position is an attractive force, F_{att}. The direction and velocity



(a) Shows the computation of the direction and velocity with PFM. The obstacle is repelling the robot while the target is attracting the robot, and combined they result in the force F_{tot}.



(b) Shows the motion direction at any location of the space when using potential field methods.

Figure 2.1: Motion computation with potential field methods. Figures copied from [42].

at a position x is determined by the sum of all the forces acting on the robot in a given moment (see Figure 2.1):

$$F_{tot}(x) = F_{att}(x) + F_{rep}(x) \tag{2.1}$$

with

$$F_{att}(x) = -\nabla \frac{1}{2} k_{att} \left(x - x_d\right)^2 \tag{2.2}$$

$$F_{rep}(x) = -\nabla \begin{cases} \frac{1}{2} k_{rep} \left(\frac{1}{p(x)} - \frac{1}{p_0}\right)^2 & \text{if } p(x) \leq p_0 \\ 0 & \text{if } p(x) > p_0 \end{cases} \tag{2.3}$$

where $p(x)$ is the shortest distance to the obstacle, p_0 is the maximum distance at which an obstacle has influence. k_{att} and k_{rep} determines the strength of the attractive and repelling force respectively.

Potential field methods were introduced to improve the real-time performance of manipulators and mobile robots. This is accomplished by transferring some of the responsibilities of collision avoidance to the low level control system. By doing this the path planner does not need to generate a fully collision free path, which makes it less computationally heavy. By letting the low level control system get data from sensors it is possible to make the robot more reactive to the environment.

Early implementations of potential field methods were used for global path planning, it was assumed that there was global knowledge of the environment and that the obstacles could be described as simple geometric shapes. In [4] and [7], Borenstein and Koren presents the virtual force field (VFF) algorithm that has adapted PFM such that it is reactive, meaning it will work in environments with unknown obstacles, and can be run in real-time on mobile robots. VFF uses a two dimensional certainty grid [51, 50] C to represent the environment. The value of a cell in C is a measure of how certain we are that there is an obstacle in that position. The greater the value the more certain we are that there is an obstacle in the cell. This approach of representing the environment is well suited for filtering out misreadings that is caused by the sensor used to gather data. To calculate F_{rep}, the repelling forces, in VFF a subset of cells C^* are selected from C with the robot in the center. Each cell in C^* applies a repelling force that is proportional to the contents of the cell and the squared distance between the robot and the cell. The attractive force, F_{att}, towards the target location is always present, it varies depending on the distance between the robot and the target location. The authors made experiments with a differential drive robot, which means it has two independent wheels, one on each side, and can rotate on the spot, that had an ultrasonic sensor. The experiment showed that the method was superior to other methods at that time.

Borenstein and Koren discovered fundamental problems with PFMs when they worked on VFF, which they addressed in [32]. The problems that they discovered where:

- The robot could get trapped because of local minima. One example of a local minima is a U-shaped obstacle.
- The robot might not be able to pass through two obstacles that are close to each other. This is because the sum of the repelling forces from the two obstacles will be greater than the attractive force.
- Oscillations can occur when the robot moves close to obstacles and in narrow passages.

Because of these problems the authors went on to developed vector field histogram, which will be presented next.

2.1.2 Vector Field Histogram

Vector field histogram (VFH), like VFF, was developed for the purpose of real-time collision avoidance on mobile robots. It was first presented in [5] and later expanded upon in [6]. Like VFF, VFH also uses a certainty grid C for representing the environment, updates it and creates the subset of cells C^* in the same way. However, the similarities between the two methods ends here. The problem with VFF lies in the fact that the data from the certainty grid is directly reduced to a single force when all repelling forces are summed. To remedy this, VFH does data-reduction in two steps. The first step is to create a one dimensional polar histogram H around the current robot location. H consists of a number of angular sectors. Each angular sector has a value that represents the obstacle density in C^* that corresponds to that sector. The obstacle density value in a sector is the sum of all obstacle vector's magnitudes in that sector, which depends on the certainty value and the distance from the robot.

The second data-reduction step is to calculate the next steering direction. This is done by searching for the sector which is the closest to the desired direction and has an obstacle density value that is below some threshold. The threshold has to be manually tuned. Too large threshold results in the robot being less aware of obstacles. However, if the threshold is too low it results in missing potential direction of motion and not being able to pass through narrow passages. The last step is to calculate the speed, V, as:

$$V = V_{max}\left(1 - \frac{\min\left(h_c, h_m\right)}{h_m}\right)\left(1 - \frac{\Omega}{\Omega_{max}}\right) + V_{min} \tag{2.4}$$

where V_{max} is the robots maximum speed, V_{min} lower limit for V preventing the speed from being zero. Ω is the current steering rate and Ω_{max} is the maximum steering rate. h_c represents the obstacle density in the current direction of travel. $h_c > 0$ means that an obstacle is in front of the robot, a large value indicates that there are either a large obstacle in front of the robot or that there are obstacles close to the robot. Lastly, h_m is a constant that is empirically determined such that a sufficient reduction in speed is achieved.

An enhanced method called VFH+ was introduced by Ulrich and Borenstein [63]. VFH+ improves upon several of the shortcomings of VFH. In VFH+ the width of the robot is directly taken into account, which is not the case in the original VFH, this in turn reduces the

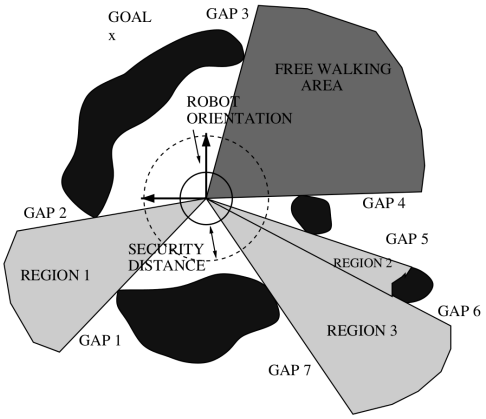


Figure 2.2: Top down view of environment, with gaps, regions and free walking area. Obstacles are in black. Figure copied from [44].

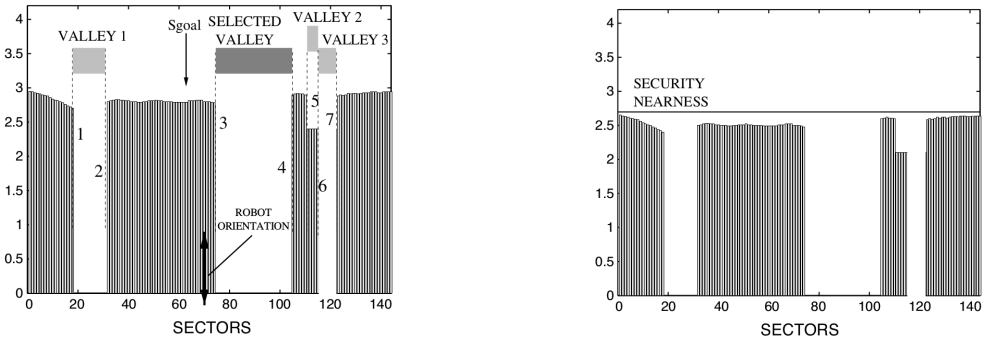
amount of parameter tuning that has to be done. Other improvements includes smoother and more reliable robot trajectories that prevents the robot from directing its motion towards obstacles.

VFH* was presented in [64] two years after VFH+ by the same authors. VFH+ only considers the current state of directions when computing the next steering command, this makes it possible for VFH+ to get stuck in local minimas. VFH* aims to solve this by also taking into account future configurations. For this to be possible global knowledge of the environment is required, where VFH* can use the A* search algorithm [25] to find a path to the goal. VFH* is therefore not a pure local obstacle avoidance method. Another requirement for VFH* to be successful is a good motion model of the robot.

2.1.3 Nearness Diagram

Nearness diagram (ND) is another reactive collision avoidance method for mobile robots. Minguez and Montano first introduced the method in [44] and it was later expanded upon in [45] by the same authors. ND uses two diagrams both of which are divided into sectors, δ, around the robot to represent the environment, in a similar fashion as in VFH. However, in ND these diagrams are constructed directly from sensor measurements and uses the actual distance to the obstacles as a metric.

The two diagrams are called PND and RND. They represent how close the obstacles are to the robot's center and to the robot's boundary



(a) PND. **(b)** RND.

Figure 2.3: The two diagrams that are used in ND based on the environment in Figure 2.2. Figures copied from [44].

respectively. For each sector i PND and RND are defined as:

$$PND_i = \begin{cases} d_{max} + 2R - \delta_i & \text{if } \delta_i > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.5}$$

$$RND_i = \begin{cases} d_{max} + E_i - \delta_i & \text{if } \delta_i > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.6}$$

where d_{max} is the sensors maximum range, R the radius of the robot, E_i is the length from the robot center to the robot boundary in the direction of sector i (R for circular robots). δ_i is the minimum allowed distance to an obstacle in sector i, if that sector does not contain an obstacle then $\delta_i = 0$.

The PND diagram is first searched for *discontinuities*. Discontinuities are when $|PND_i - PND_j| > 2R$ for two adjacent sectors i and j, and are noted as *gaps* in Figure 2.2 and as numbers inside Figure 2.3a. Next valleys are formed between two discontinuities as seen in Figure 2.3a and in Figure 2.2 as *regions*. The valley, or region, closest to the goal is then selected as the *selected valley*, or *free walking area*, it is in the direction of this valley that the robot will attempt to move next.

The RND diagram is used when a selected valley has been found, Figure 2.3b displays an example of a RND diagram. By inspecting the sectors in the RND diagram that corresponds to the sectors in the PND diagram for the selected valley, it is possible to determined which out of five possible situations the robot is in when moving towards the selected valley. The five situations decide the robots velocity and

direction inside the selected valley and they are represented as a binary tree and are therefore exclusive and complete. The five situations are:

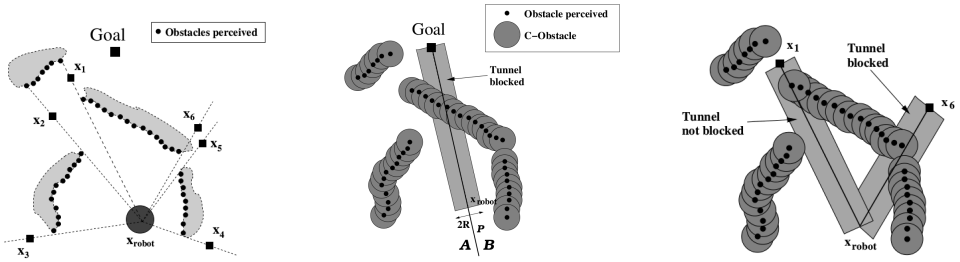
1. There are obstacles close on one side of the selected valley.
2. There are obstacles close on both sides of the selected valley.
3. There are no obstacles close in the selected valley and the selected valley is wide.
4. There are no obstacles close in the selected valley and the selected valley is narrow.
5. The goal is inside the selected valley.

Experiments were performed and the method was compared against PFM and VFH among other. The authors states that the method does not suffer from the fundamental problems of PFMs. There is no parameter that has to be tuned and altered when moving from an open area to a cluttered, as there is with VFH. ND takes into account the width of the robot, something VFH does not, which means that ND does not cut corners while VFH can do that. The author also explains that VFH+ solves most of the problems with VFH but that it instead makes it impossible to direct motion towards an obstacle, meaning it is not well suited for moving in cluttered environments.

2.1.4 Obstacle-Restriction Method

Minguez, one of the authors of ND, presented the obstacle-restriction method (ORM) in [41]. Something that is special about ORM compared to the previously mentioned methods is that ORM uses all available obstacle information in all the parts of the method, i.e., there is no data-reduction. The method assumes that the obstacle information is in form of points, hereafter called *obstacle points*, and gathered continuously by a distance sensor.

ORM consists of two parts. The first part is about selecting a direction of motion, since it is not always possible to move directly towards the goal. Therefore the first part starts with creating a list of potential sub-goals. A potential sub-goal is a location which is either:



(a) x_1 to x_6 denotes sub-goals in ORM.

(b) Shows how the rectangle is split into two rectangles, A and B.

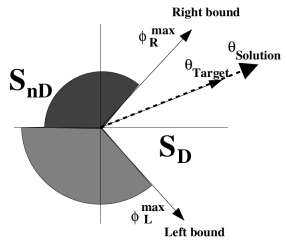
(c) Shows when a sub-goal can be reached and not reached.

Figure 2.4: Shows the first part of ORM. Figures copied from [41].

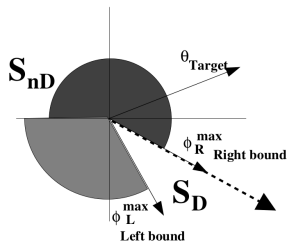
- In the gap between two obstacle points which are angular continuous, meaning that the resolution of the distance sensor used makes it impossible for an obstacle point to be registered between them. The distance between the obstacle points have to be greater than the robot diameter. If that is fulfilled a sub-goal is located between the two obstacle points. x_1 and x_2 in Figure 2.4a are two such sub-goals.
- In the direction of an edge of an obstacle at a distance further than the robot diameter. See x_3, x_4, x_5, and x_6 in Figure 2.4a.

After the list of sub-goals has been compiled it is time to determine if the goal or one of the sub-goals should be used in the second part of the method. If the goal location can be reached from the current location then it is used. Otherwise the sub-goal with the minimum angular difference relative to the goal and that can be reached is used. To determine if a location, x_a, can be reached from another location, x_b, a rectangle between the two locations with a width equal to the robots diameter is constructed. This rectangle is split along the line from x_a to x_b into two rectangles, A and B, which can be seen in fiugre 2.4b. The robot can reach x_a from x_b if all of the obstacle points, expanded with the radius of the robot, inside A is at a greater distance than the robots diameter from all of the obstacle points, also expanded with the radius of the robot, inside B. An example of sub-goals that can and cannot be reached can be seen in Figure 2.4c.

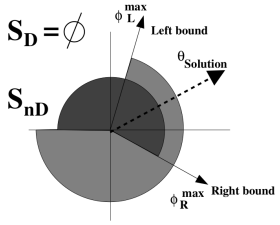
The second part of the method is to compute a direction of motion that moves the robot towards the goal or sub-goal, hereafter referred to as *target*, selected from the previous part while simultaneous avoiding collisions with obstacles. This is done by first calculating a set of



(a) First motion direction, the target direction is inside S_D.



(b) Second motion direction, the target is not inside S_D. However, S_D does contain directions.



(c) Third motion direction, S_D does not contain directions.

Figure 2.5: Shows the three cases that can occur when selecting motion direction in ORM. Figures copied from [41].

motion constraints, S_1 and S_2, for each obstacle. Relative to the robot each obstacle has two sides. S_1 is the set of directions on the side of the obstacle that is opposite to the *target*, these directions are not suitable to do the avoidance. S_2 is the set of directions that, if moved towards, would put the obstacle inside the robots security zone. The robots security zone is an area around the robot bounds and is therefore defined by the radius of the robot and a security distance, D_s. The complete motion constraint for an obstacle is the union of S_1 and S_2, $S_{nD} = S_1 \cup S_2$.

When S_{nD} has been calculated for each obstacle a left and right bound is computed. The left bound, ϕ_L, is the leftmost direction of all motion constraints for the obstacles to the right of the *target*. The right bound, ϕ_R, is the rightmost direction of all motion constraints for the obstacles to the left of the *target*. The set of desired directions, S_D, is the complementary of the union of every obstacle's S_{nD}.

There are three cases of motion direction to selected from depending on S_D:

1. If S_D contains the target direction then that direction of motion is selected, as can be seen in Figure 2.5a.

2. If S_D does not contain the target direction but contains other directions then the direction that is closest to the target direction is selected, as shown in Figure 2.5b.

3. If S_D contains no direction then the selected direction of motion will be $\frac{\phi_L + \phi_R}{2}$, displayed in Figure 2.5c.

Experiments with ORM were performed, by the authors, with a differential drive wheelchair and the results were compared to ND. The conclusion from the experiments was that ORM performs better in open spaces and equal in cluttered locations but with smoother movement.

2.1.5 Kinematic and Dynamic Constraints

None of the above mentioned collision avoidance methods take into account the kinematics and dynamics of the robot (at least not in their original form). All of them more or less assumes a holonomic robot. There are however collision avoidance methods that do take the kinematics and dynamics of the robot into account.

The Dynamic window approach (DWA) [16] is an example of a method that do take into account the kinematics and dynamics of the robot, since it is derived directly from the motion dynamics of the robot. However, in this thesis such methods are ignored in favor of having the collision avoidance and the kinematic and dynamic constraints in separate modules. By doing this it is simpler to altered the proposed method in this thesis if new insights into any of the two areas is presented later on. Next will therefore research on how to take into account the kinematic and dynamic constraints without having to alter the collision avoidance methods mentioned above.

Minguez, Montano, and Santos-Victor presented, in 2002, an approach that makes the collision avoidance methods, without having to be modified, implicitly take into account the robot's kinematic constraints [49]. This is to solve the problem of applying collision avoidance methods to non-holonomic robots. The idea is to use something called Ego-Kinematic Transformation (EKT) that maps each point (in the Euclidean space) in the robot's frame of reference to a new space, called Ego-Kinematic Space (EKS). In EKS the motion constraints of the robot are embedded.

The same year, Minguez, Montano, and Khatib [47] addressed the problem of applying collision avoidance methods to robots with dynamic constraints that cannot be ignored. There are two parts, where the idea of the first part is similar to the one in [49] described previously. To take the dynamic constraints into account the space is again altered. This is done by transforming the obstacle distances into distances that depends on the sampling time, which is the time between

each motion command, and the deceleration capabilities of the robot. The new space is called Ego-Dynamic Space (EDS) and it is obtained by applying the Ego-Dynamic Transformation (EDT):

$$d_{obs} \rightarrow d_{eff} = a_b T^2 \left(\sqrt{1 + \frac{2 d_{obs}}{a_b T^2}} - 1 \right) \tag{2.7}$$

where d_{obs} is the real measured distance to the obstacle from the robot, T is the sampling time, and a_b the maximum deceleration. d_{eff} is the maximum distance the robot can travel in the direction of the obstacle before having to apply maximum deceleration to prevent collision. The collision avoidance method is then applied in the EDS.

The second part consists of selecting the closest collision-free position in a *spatial window* (SW) to the position the collision avoidance method suggested. SW is the set of all possible locations that can be reached within the sample time T assuming constant velocity, and is defined by the corners:

$$\begin{aligned} X_{min} &= (v_x - \Delta v)T \\ X_{max} &= (v_x + \Delta v)T \\ Y_{min} &= (v_y - \Delta v)T \\ Y_{max} &= (v_y + \Delta v)T \end{aligned} \tag{2.8}$$

where $[v_x, vy]$ is the current velocity. Figure 2.6 shows an example of a SW. Bipin, Duggal, and Krishna [3] used EDT and SW for autonomous navigation with a quadcopter, in 2015. They compared the performance of the autonomous navigation without and with EDT. The success rate, how often it avoided obstacles, increased from 62.5% to 79.16%. The authors stated that the narrow field of view (93°) of the obstacle detection sensor was the main culprit to failure.

In 2003, Minguez and Montano [46] combined the work in [49] and [47] to create the Ego-KinoDynamic space (EKDS) and the Ego-KinoDynamic transformation (EKDT). EKDS is obtained by applying EKDT, which in turn is the result of first applying a modified EDT and then EKT. EDT is modified for robots that move in arcs (non-holonomic, like a car or differential drive). In EKDS the robot's shape, kinematics, and dynamics is implicitly taken into account. Minguez and Montano [43] presented, in 2009, another approach for taking into

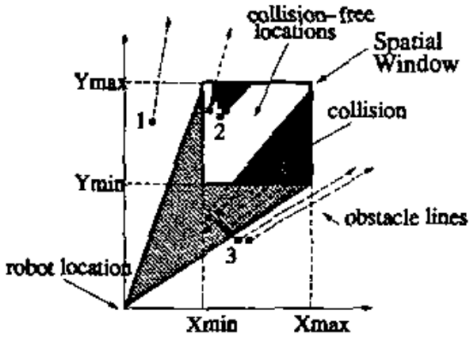


Figure 2.6: Spatial Window. Figure copied from [47].

account the robot's shape, kinematics, dynamics in a unified framework. This work was an extension of Minguez, Montano, and Santos-Victor [48] work from 2006, which did not take into account the dynamics of the robot. In the three papers the main focus has been on ground based different drive (or similar, such as car) robots.

2.2 Human-Robot Interface

Tele-operation requires an operator to operate a robot based on the information that the operator is provided. In this section we will go through and look at research to get a better understanding of how to make the human-robot interface such that the operator's performance is maximized.

2.2.1 Use of Cameras

One of the most common ways of presenting the environment for a tele-operator is through one or more cameras mounted on the robot. For most commercial UAVs, like the ones from DJI [17] and Parrot [55], a video feed from a forward facing camera is the only feedback the operator is provided about the environment. The video feed from the cameras makes it so that the operator can see the environment in a similar fashion as if physically being there. In [71] a couple of problems with seeing the world through a camera is brought up. Scale is one of these problems which makes it hard for the operator to know where it is possible to go. An example of this is in the search and rescue mission at the World Trade Center (WTC) [11]. The tele-operators of the robots there only had a video feed from the robot as feedback. That resulted

in the robots getting stuck multiple times because the operators could not perceive whether the robot would fit through openings.

Estimating depth and the rate of motion are two other problems, which are brought up in [71]. One key observation about the depth problem is that we normally see the world in stereo with our two eyes. The video feed is presented on a flat monitor and this creates a conflict since everything is at the same depth from our eyes. A contributing factor to why there is a problem with estimating the rate of motion is our vestibular system.

In [71] it is stated that people do not always move and gaze in the same direction, instead they gaze where they want to go in the future or at something of interest. When tele-operating a robot the camera is most often mounted statically in the front of the robot. This causes a conflict since the operator will always gaze in the same direction relative to the robot. Where the cameras are mounted on the robot effects how the human operator's perceives the environment. When the camera is mounted low towards the ground it presents unnatural viewing angles for the operator which may result in degraded perception [52].

Field of view (FOV), frame rate, and resolution characterizes the video feed, we will next look at what impact these have on the tele-operator's performance.

Field of View

Limited FOV can cause something called the keyhole effect [71]. The keyhole effect makes noticing new things, navigating in the environment and modelling the world more difficult.

Witmer and Sadowski [70] conducted an experiment, with 24 participants, where they compared the human performance of judging distances in a real world environment compared to a virtual environment. In the virtual environment the participants had 140° horizontal and 90° vertical FOV compared to 200° horizontal and 135° vertical FOV of the human eyes combined [27]. The result from the experiment showed that humans judge distances better in a real world environment. The authors states that one of the reasons for the degraded depth perception in the virtual environment may be due to the limited FOV.

In [40], McGovern conclude that with a wider FOV (three 40° cameras for a 120° FOV horizontally) operators found it "easier" to oper-

ate in a restricted environment compared to a narrow FOV (one 40° camera). With the narrow FOV operators were not comfortable with turning corners. In [59] a similar conclusion was stated with reference to two other papers. The first reference used stated that a wide FOV (three 60° FOV cameras) made operation "easier" and that it was useful when turning and operating in close quarters. It was also stated that a wide FOV is useful for maintaining spatial orientation. The second reference found that the number of collisions with obstacles was significant lower when using a wide FOV compared to a narrow FOV.

Smyth [61] conducted an experiment on the effects of camera FOV when driving a military vehicle. In the experiment 150°, 205° and 257° FOV camera lenses were used and compared. The video feed from the camera was presented on three flat panel displays mounted side by side creating a 110° view. A conclusion from the experiment was that increased FOV lead to reduced speed of travel. A reason for this was that the scene compression with higher FOV made the speed to be perceived as increased. In contrast, Van Erp and Padmos [65] compared 50° FOV with 100° FOV and wrote that with increased FOV operators found it simpler to estimate the speed. With wider FOV estimations of time to contact, location of obstacles and when to start turning into a sharp turn were in most cases improved. The latter can be explained by the use of the tangent point when turning [34], with a wider FOV the tangent point is always visible.

Frame Rate and Resolution

Frame rate and resolution are grouped together since these constitutes most of the bandwidth of the video feed. It is therefore important to determine how to trade-off the two such that the operators performance can be maximized given a limited bandwidth.

McGovern [40] performed experiments about tele-operating land vehicles on roads and off-road. From the experiments they concluded that resolution does not have any major impact on performance if there are no obstacles or the obstacles are of different sizes and types.

Massimino and Sheridan [39] conducted an experiment in 1994 where they tested how various forms of visual and force feedback effect the human performance on different tele-manipulation tasks. One part of this was to test three different frame rates: 3, 5 and 30 frames per second (FPS) with and without force feedback. Force feedback

made it so that the operator could feel what was happening. The average time for completing the tele-manipulation tasks without force feedback at 3 FPS was 5.36 seconds, at 5 FPS it took 4.48 seconds and at 30 FPS it took 2.56 seconds. With force feedback 3 FPS and 5 FPS it took were not significant different from 30 FPS without force feedback. 30 FPS with force feedback was significantly faster than all other. In 1996 a similar experiment was done but in a virtual environment [56]. Here 28, 14, 7, 3, 2 and 1 FPS was compared. The result were that from 28 to 14 FPS the performance was similar with no statistically significant difference. From 7 FPS and below the performance decreased drastically.

In 1997 the effects of frame time variation was studied [69]. By using head-mounted display they measured the performance of 10 participants doing different tasks in a virtual environment presented in different frame rate and varied frame time. The conclusions from the study was that when the FPS is high enough (20 FPS) then fluctuation in frame times have little or no effect on the operators performance. When the FPS is low (around 10 FPS) then fluctuations in frame times have noticeable effect on performance. In general at least 10 frames per second is recommended for operating in virtual environments [68].

In [65], different combinations of frame rate and resolution were tested when teleoperating a vehicle. A baseline of 30 FPS at a resolution of 512 by 484 pixels were used. There were five tasks that the operators performed: turning sharp curves, estimating longitudinal distances, braking, lane change and estimate target speed. The conclusion was that both the frame rate and the resolution could be lowered to 10 FPS and 256 by 242 pixels respectively without any significant difference to the baseline.

In 2006 Thropp and Chen reviewed over 50 studies on the effects of frame rate on human performance [62]. They summarized them into four areas: perceptual performance, psychomotor performance, subjective perception and behavioral effects. frame rates at around 16 FPS is recommended for teleoperation, where navigation and target tracking is important. However, 10 FPS may be sufficient if bandwidth is a problem. In general the authors state that frame rates at 10 FPS and below can cause stress and general performance decrements. For most tasks it seems that 15 FPS is the minimum for no significant performance degradation. At 10 FPS accaptable performance can be achieved for many tasks. In the same year another study of previ-

ous work was published [12]. In an experiment where they teleoperated ground vehicles no performance degradation was found when the frame rate lowered from 30 FPS to 7.5 FPS. In another experiment they found that 2 FPS and 4 FPS degraded the performance but there were no significant difference between 8 FPS and 16 FPS.

2.2.2 Latency

Latency is the delay between an input action and the output response. The source of latency can be a number of things: the sampling rate of input devices, the update rate of output devices, intermediate computations, or that information is transmitted over a communication network [38]. It has been demonstrated that humans can detect latency as low as 10-20 ms [18]. It has been shown that at about one second of latency humans tend to use the *move and wait* strategy, which means that they input a command and wait for the response before inputting a new command [20]. One of the earliest experiments on the effects of latency on human tele-manipulation performance were performed by Sheridan and Ferrell [60], in 1963. They found that the time for completing the tasks increased significantly when the latency increased. It should be noted that the latency were in order of seconds in the experiments.

In 1993, MacKenzie and Ware [38] performed an experiment to determine how the human performance is affected by latency. Eight volunteers performed the task of moving the mouse cursor into a region of the screen under different amount of latency. All of the volunteers had prior experience of operating a mouse. 8.3, 25, 75, and 225 ms of latency were tested by buffering mouse samples and delaying the update of the screen. The results show that movement times increased by 63.9% (1493 ms compared to 911 ms) and error rate by 214% (3.6% to 11.3%) when latency was increased from 8.3 ms to 225 ms. 8.3 ms and 25 ms showed similar results, while both the movement time and error rate were noticeable worse at 75 ms. In contrast, Lane et al. [35] found no statistical significant performance degradation with latencies from one second and below, in 2002. However, in [35] the latency increased with each successive test, meaning that the test subject had more practise before doing the test with increased latency. In 1988, Frank, Casali, and Wierwille [21] found that the operators performance was significantly degraded in simulated driving task with a latency of 170 ms.

Latency and frame rate are two topics that are closely related. A change to either of them can affect the other. Both Arthur, Booth, and Ware [2], in 1993, and Ellis et al. [19], in 1999, found that latency has a greater effect on human performance than frame rate. In [2], they compared 3D tracing task performance with latency between 50 ms to 550 ms and frame rate of 10, 15, and 30 FPS. While in [19] they examined human 3D tracking performance with frame rates of 6, 12, and 20 FPS and 480, 320, 230, 130, and 80 ms latency.

The effects of variable compared to constant latency is another aspect that has been investigated. DePiero, Noell, and Gee [15] wrote "Driving experience using the ORNL system has demonstrated the significance of latency (image age) on driver performance. Both a low and a constant value of latency is very important.". In [35] it is stated that a low by variable latency can have a more severe effect on performance than a higher but fixed latency. Watson et al. [68] found that in grasping and placement tasks with a standard deviation of latency at 82 ms or less there were no significant degradation of performance. Luck et al. [37] performed a tele-operation experiment where the participants controlled a robot through different courses with a mean latency of 1, 2, and 4 seconds that were both fixed and variable (50% around the mean latency). It took significant longer to complete the courses with variable latency. However, with constant latency an increased amount of errors occurred. The authors propose that the increase in errors is the result of the operator feeling more confident with fixed latency.

2.2.3 Map and Orientation

At the WTC search and rescue mission, as mentioned earlier, the robots got stuck since the operators could not determine where the robot would fit based on the video feed from the robot, which was the only feedback they had [11]. There were also reports that they had a hard time orienting themselves in there and that they got lost. If the operators had had a map their performance might have increased.

There have been a number of studies that have compared track-up maps and north-up maps. Tack-up maps are ego-referenced and are always rotated such that what is in front of the robot is always up. North-up maps are world-referenced and are therefore fixed. For local navigation it has been shown that track-up maps are better and that

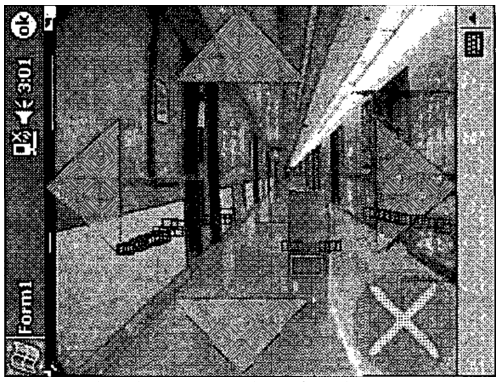


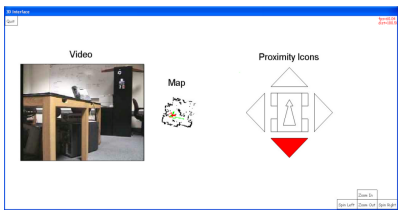
Figure 2.7: Map overlaid onto video feed. Figure copied from [28].

north-up maps are better when global awareness is of importance [1, 10, 13, 36, 14]. Campbell, Carney, and Kantowitz [9] recommends that both track-up and north-up maps should be available. When navigating the default should be the track-up map and when planning the default should be the north-up map.

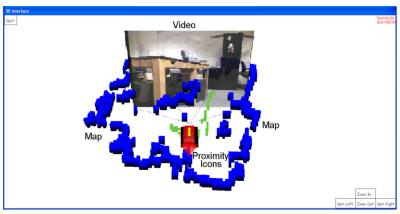
2.2.4 Combining Camera and Map View

Keskinpala and Adams [28] conducted an experiment where 30 participants operated a mobile robot using three different interfaces. The three interfaces were: video feed only, map (constructed from sensor data) only and the last were both combined with the map overlaid on top of the video feed, as seen in figure 2.7. The combination of video feed and map resulted in the worst performance. However, the authors stated that the reason for this was because of the processing delay that occurred when putting the map onto the video feed.

Nielsen and Goodrich [54] did a similar experiment where they compared: 2D map only, 2D map and video feed, video feed only, 3D map only and lastly 3D map and video feed. 2D map and video feed meant that the 2D map and video feed were side by side, as seen in figure 2.8a. The video feed was integrated into the 3D map in the 3D map and video feed view, figure 2.8b. When the map and video feed were side by side they competed for the operator's attention. With the video feed integrated into the 3D map performance was overall the best.



(a) 2D map and video feed side by side.



(b) 3D map and video feed combined.

Figure 2.8: Combining map and video feed. Figures copied from [54].

2.2.5 Haptic Feedback

Everything mentioned earlier in this section has been relying on the same sense to give feedback about the environment to the operator, namely vision. Haptic feedback on the other hand utilizes the sense of touch to provide feedback to the operator. An experiment in 1994 [39] demonstrated that haptic feedback is useful and increases the performance in telemanipulation tasks.

In an effort to aid people with handicaps an omnidirectional wheelchair, which means that it can move in any direction at any time, with a built in collision avoidance system, among other assisting systems, was developed [8]. The operator would control the wheelchair with a joystick. To prevent collisions the collision avoidance system would automatically modify the operators control input if necessary. Kitagawa et al. [31] stated that "automatic collision avoidance is uncomfortable for the wheelchair operator". They in turn developed a collision avoidance system, also for a omnidirectional wheelchair, that did not automatically modify the operators control input. Instead the system would dynamically alter the stiffness of the joystick based on the closest obstacle in the input direction and the velocity of the wheelchair. The closer an obstacles comes and the higher the velocity is the stiffness will increase. The result from this was that the operator found the obstacles intuitively and successfully avoided collisions according to the operator's intentions.

Han and Lee did something similar in [24]. From the environment information they created a force vector that was sent to the joystick. The forces in the force vector were proportional to the velocity of the robot and the distance to the obstacles. The operator could then understand the environment based on the amount of force needed to move the joystick in a certain direction. Experiments were conducted

where two operators tele-operated a robot from a starting position to a goal position with and without the joystick feedback. A camera was mounted on the robot and the operators used the video stream to see where to go. The experiments were performed in a dark environment. They also let the robot drive fully autonomously to compare the performance. It took 58 seconds for the fully autonomous robot, 46-48 seconds when tele-operated with joystick feedback, and one minute and 20 seconds without joystick feedback. Without joystick feedback the operator collide with obstacles multiple times and followed a less optimal path towards the goal. Therefore, best results were obtained when tele-operated with joystick feedback.

Chapter 3

Method

In this chapter, a complete tele-operation system for UAVs will be presented. First a collision avoidance method that has been optimized for use with a human operator is presented. Secondly a human-UAV interface that provides the operator with feedback about the environment to increase the situational awareness of the operator is presented.

3.1 Collision Avoidance Method

In Section 2.1 a number of collision avoidance methods were discussed. Out of these ORM (Section 2.1.4) was chosen as the most appropriate for this project. ORM does not have the mentioned problems that PFMs have: oscillations, passing through two obstacle that are close to each other and local minima. Local minima can be a problem with ORM depending on the sensor used for detecting obstacles, if the sensor does not see far enough. Minguez, the author of ORM and one of the authors of ND, compared the two methods and came to the conclusion that both perform equally well in cluttered locations and that ORM performs better in open spaces [41].

Both PFMs and VFH uses certainty grids, which requires constant tracking of the current location to be updated correctly. With ORM there is no need to track the current location, instead the most recent obstacle information is used. PFMs, VFH and ND all does some form of data reduction, this makes it hard to know exactly how far away the obstacles are. In VFH, for example, it is difficult to know if there are multiple obstacles far away in a certain area or if a single obstacle is close, since these situations can give similar response. It is therefore

hard, or impossible, to know if it is safe to move in a certain direction.

Next implementation details and how ORM was adapted for teleoperation will be presented.

3.1.1 Sensor Data Processing

Most parts of ORM benefits from having the obstacle information sorted based on the angle to the obstacle from the UAV frame of reference. Therefore the data from the distance sensors are processed and combined into a single array, d, of length n. An element at index i in d corresponds to the nearest distance measurement to an obstacle at angle $\frac{i}{n}2\pi$ relative to the UAV.

Algorithm 1 presents the full procedure. Each element in d is first initialized to $min\_distance$ which is a minimum distance that an obstacle can be from the UAV. This is to prevent the UAV from moving in directions where there is no sensor information. The algorithm assumes that the sensor data is in the form of a matrix, where each element in the same column is at the same azimuth angle relative to the UAV. It also assumes that the azimuth angle always increases or decreases monotonous when moving from one column to the next in the same direction. From line 8 the minimum distance to an obstacle is found for a specific column, and later used as the nearest distance for that angle. This is to simplify the environment, by reducing it from 3D to 2D.

In the pseudo code, the lines after after 19 makes the algorithm compatible with lower resolution sensors. If using a sensor such that when moving from column l to $l + 1$ the corresponding indices in the array d increases/decreases with more than one, it would set the elements between the indices to -1. A value of -1 indicates that the resolution of the array d is higher than that of the data from the distance sensors, and therefore that value should be ignored.

input : An array S containing matrices of size $rows \times columns$. Each matrix contains distance sensor data from one of the distance sensors
output: An array d where the data at index, i, is the distance measurement to the nearest obstacle at the angle $\frac{i}{\texttt{Size}(d)} 2\pi$ relative to the UAV

Initialize each element in d to $min\_distance$;
Initialize each element in array u (of same size as d) to $false$;
// Each element in u indicates if the corresponding element in d has been updated
for $s \in S$ **do**
  prev_angle $\leftarrow -1$;
  for $x \leftarrow 1$ **to** NumColumns(s) **do**
    min_column_distance $\leftarrow \infty$;
    for $y \leftarrow 1$ **to** NumRows(s) **do**
      distance $\leftarrow$ GetDistance($s[x, y]$);
      if distance $<$ min_column_distance **then**
        min_column_distance $\leftarrow$ distance;
      end
    end
    angle $\leftarrow$ GetAngleTo(s, x);
    index $\leftarrow \frac{\text{angle}}{2\pi}$ Size(d) ; // Rounded to the closest integer
    if $\neg u$[index] ***or*** min_column_distance < d[index] **then**
      u[index] $\leftarrow true$;
      d[index] $\leftarrow$ min_column_distance;
    end
    if prev_angle $\neq -1$ **then**
      prev_index $\leftarrow \frac{\text{prev\_angle}}{2\pi}$ Size(d);
      min_index $\leftarrow$ Min(index, prev_index);
      max_index $\leftarrow$ Max(index, prev_index);
      if max_index $-$ min_index $> \frac{\texttt{Size}(d)}{2}$ **then**
        $temp \leftarrow$ min_index $+$ Size(d);
        min_index $\leftarrow$ max_index;
        max_index $\leftarrow temp$;
      end
      for $j \leftarrow$ min_index $+1$ **to** max_index -1 **do**
        if $\neg u[j \mod \texttt{Size}(d)]$ **then**
          $d[j \mod \texttt{Size}(d)] \leftarrow -1$;
        end
      end
    end
    prev_angle $\leftarrow$ angle;
  end
end

Algorithm 1: Angular Distance Measurements

3.1.2 Obstacle-Restriction Method for Tele-operation in The Plane

ORM was designed for navigating a robot from its current location to a goal location. When tele-operating, the goal is often in the form of a direction, θ, and a magnitude[1], m. It should be noted that there are other ways of tele-operating a UAV. An example would be to place waypoints on a map, for this to be efficient a global map of the environment have to be available or simultaneous localization and mapping (SLAM) would have to be done. For this project we want to be able to explore unknown environments, which means there is no global map, and SLAM is out of the scope for this project. Therefore the first mentioned approach of tele-operating is used. With the UAV as frame of reference (meaning the current location defines the origin), a goal location is obtained by:

$$goal\_location = (m\cos(\theta), m\sin(\theta)) \tag{3.1}$$

In Section 2.2.5 we learned that a problem with automatic collision avoidance for tele-operation is that the operator might feel uncomfortable when the robot moves in another direction than intended. To solve this problem two parameters, dir_{diff} and opp_{diff}, are introduced to the first part of ORM, where it looks for a sub-goal.

dir_{diff} is used to restrict the area where a sub-goal can be located. This area is defined by the two angles $\theta - dir_{diff}$ and $\theta + dir_{diff}$, meaning that a sub-goal has to be located where the angular difference between the direction towards the sub-goal and the direction inputted by the operator, θ, is a maximum of dir_{diff}. This restriction prevents the UAV from making drastically different movements than expected, while still being able to correct the operators input such that the UAV moves where the operator intends and avoids obstacles.

opp_{diff} is introduced to deal with the situations where the operator's intentions are ambiguous. Imagine a situation where the $goal\_location$ is blocked and there are suitable sub-goals on both the left side, s_L, and right side, s_R, of the $goal\_location$, with the UAV as frame of reference. We define δ_L and δ_R as the absolute angular differences between the goal-location and s_L and s_R respectively. When δ_L and δ_R are close to similar it is difficult to determine the operator's intentions and thus

[1]If the input device is a joystick the direction could be the direction the stick is pointed and the magnitude could be how far the stick is moved.

decide if s_L or s_R should be chosen. opp_{diff} defines how different δ_L and δ_R has to be for one of the sub-goals to be chosen:

$$target = \begin{cases} s_L & \text{if } |\delta_L - \delta_R| \geq opp_{diff} \text{ and } \delta_L < \delta_R \\ s_R & \text{if } |\delta_L - \delta_R| \geq opp_{diff} \text{ and } \delta_R < \delta_L \\ goal\_location' & \text{otherwise} \end{cases} \quad (3.2)$$

where $goal\_location'$ is the $goal\_location$ moved closer to the UAV such that it is no longer blocked.

Both dir_{diff} and opp_{diff} determine how accurate the operator has to be and how much control the operator has. With a larger dir_{diff} and a smaller opp_{diff} the operator does not have to be as accurate when moving the UAV through small openings and similar situations. However, if the operator would like to move closer to an obstacle to get a better view of it the larger dir_{diff} and smaller opp_{diff} would make this difficult since the collision avoidance system would try to move the UAV around the obstacle. A smaller dir_{diff} and a larger opp_{diff} would have the opposite effect, it would give the operator more control but demand better accuracy.

Situation Dependent Parameters

The best values for dir_{diff} and opp_{diff} are typically situation dependent. Therefore, it is proposed that these parameters are defined as a function of m:

$$dir_{diff} = (dir_{diff\_max} - dir_{diff\_min})\frac{m}{m_{max}} + dir_{diff\_min} \quad (3.3)$$

$$opp_{diff} = (opp_{diff\_max} - opp_{diff\_min})\left(1 - \frac{m}{m_{max}}\right) + opp_{diff\_min} \quad (3.4)$$

where dir_{diff_max} and dir_{diff_min} are the maximum and minimum wanted value of dir_{diff}, opp_{diff_max} and opp_{diff_min} are the maximum and minimum wanted value of opp_{diff}, and m_{max} is the maximum value that m can take. When the operator inputs a smaller magnitude, m, the proposed collision avoidance system will only be able to do small adjustments meaning that the operator has more control over where the UAV moves but at the same time requires higher accuracy. This makes it easier for the operator to move closer to obstacles to get a better view

of them. If the operator instead inputs a larger magnitude the collision avoidance system can make bigger adjustments, which means that the operator does not have to be as accurate. The latter is well suited for when the operator wants to move and explore the environment more quickly.

The second part of the collision avoidance method is similar to the second part of ORM. The only difference is that instead of calculating the motion constraints S_1 and S_2, the obstacle points are divided into two sets, L_{obs} and R_{obs}. L_{obs} contains all obstacle points that are to the left of the target location, with respect to the UAV. R_{obs} contains all obstacle points that are to the right of the target location, with respect to the UAV. For each obstacle point, obs, in L_{obs} or R_{obs} a right bound, $\phi_R^{'}$, and left bound, $\phi_L^{'}$, is calculated respectively:

$$\phi_R^{'} = obs_{dir} + (\alpha + \beta) \tag{3.5}$$

$$\phi_L^{'} = obs_{dir} - (\alpha + \beta) \tag{3.6}$$

where:

$$\alpha = \left| atan\left(\frac{R + D_s}{obs_{dist}}\right)\right| \tag{3.7}$$

$$\beta = \begin{cases} (\pi - \alpha)\left(1 - \frac{obs_{dist} - R}{D_s}\right) & \text{if } obs_{dist} \leq D_s + R \\ 0 & \text{otherwise} \end{cases} \tag{3.8}$$

where obs_{dir} and obs_{dist} are the direction and distance towards the obstacle from the UAV, R is the radius of the UAV, and D_s is the security distance from ORM. The $\phi_R^{'}$ that is furthest to the right compared to the target direction, or the one closest to the left of the target direction if there are none to the right, is selected as ϕ_R. Similarly, the $\phi_L^{'}$ that is furthest to the left compared to the target direction, or the one closest to the right of the target direction if there are none to the left, is selected as ϕ_L.

With the UAV as frame of reference, there are now three cases:

1. ϕ_R is to the left of the target and ϕ_L is the right of the target. In this case the UAV will move directly towards the target.
2. Either ϕ_R is to the right of the target or ϕ_L is to the left of the target. In this case the UAV will move towards either ϕ_R or ϕ_L depending on which is closest to the target direction.

3. ϕ_R is to the right of the target and ϕ_L is to the left of the target. In this case the UAV will move towards the direction given by: $\frac{\phi_R+\phi_L}{2}$.

The direction that should be moved towards next have now been calculated. What is left is to decide the target velocity, vel, which has been formulated with inspiration from VFH (see Equation 2.4):

$$vel = vel_{max} \cdot \min\left(\frac{\min\left(h_m, obs^*\right)}{h_m}, \frac{m}{m_{max}}\right) \tag{3.9}$$

where vel_{max} is the maximum velocity, obs^* is the distance to the closest obstacle, and h_m is a constant that is empirically determined such that a sufficient reduction in speed is achieved, as in Equation 2.4. m is the magnitude of the input command, as motioned earlier, and m_{max} is the maximum value that m can obtain. The velocity is dependent on the magnitude, m, to make it possible for the operator to move slower if so desired. m influence both the velocity of the UAV and how sub-goals are selected.

3.1.3 Situations With No Input

Situations where no input is given to the UAV is likely to occur when a human operator is controlling the UAV. These situations are often not considered when designing a collision avoidance system. This is because it is assumed that there is always a target location until the UAV has reached the last location where it simply stops. It is important to define what the collision avoidance system should do when no input is given.

In a real world environment the conditions changes constantly. It is therefore not sustainable to assume that the UAV will be able to be completely still and hold its position while in the air. Perhaps there is a sudden wind that the UAV will not be able to compensate for, or that a moving obstacle approaches it.

To remedy these issues, this thesis proposes that the first course of action when there is no input is to stop the UAV from moving by actively braking. This is to prevent the UAV from colliding into an obstacle in its current heading. The next course of action is to continuously check that the UAV is at a safe distance from surrounding obstacles. A safe distance is in this case the security distance, D_s, from ORM. It

is done by actively moving towards the midpoint with respect to the obstacle points that are within the security distance of the UAV:

$$target = \left(\frac{x_{min} + x_{max}}{2}, \frac{y_{min} + y_{max}}{2} \right) \tag{3.10}$$

where x_{min} and x_{max} are the minimum and maximum x-coordinates of the obstacle points, rotated 180° around the UAV, within the security distance of the UAV. y_{min} and y_{max} is the same except for the y-coordinates. The obstacle points are rotated 180° around the UAV to ensure that the UAV moves away from them instead of towards them. By constantly moving towards $target$ the UAV will avoid obstacles even if they are moving towards the UAV. One exception is if there are obstacles in opposite directions inside the security distance approaching the UAV. In this situation the UAV will move towards the middle of the obstacles as long as possible but will not be able to avoid a collision if the obstacles move close enough.

3.1.4 Kinematic and Dynamic Constraints

The collision avoidance method presented in Section 3.1.2 does not explicitly take the kinematic and dynamic constraints of the UAV into account, just like the original ORM. Instead, the dynamics are implicitly taken into account by applying the EDT, from Section 2.1.5, to the distances in the array d, mentioned in Section 3.1.1, to create the EDS. The collision avoidance method is then applied in the EDS.

The UAVs are holonomic, thus their is no need for using EKT and EKS. All the other works presented in Section 2.1.5 mostly focused on ground robots with differential drive (or similar), therefore EDS and EDT seemed most appropriate for this project. In [66] it was showed that EDT is well suited for use with UAVs.

3.2 Human-UAV Interface

In this section a human-UAV interface will be presented based on the ideas and experimental results presented in Section 2.2. A number of different ideas on how to give feedback to a tele-operator was presented in Section 2.2, this will be combined to one solution in this section.

3.2.1 Cameras

The use of cameras is one of the most common approaches for giving feedback about the environment to a tele-operator, as mentioned in Section 2.2.1. This might not come as a surprise since a camera resembles the human eye, which is the main way humans gather information about the environment. It therefore seems fitting to use cameras for this project.

Based on what was mentioned in Section 2.2.1 a wide horizontal FOV, around 150° to 200°, results in better all-around tele-operation performance and will be used for this project. With a wide FOV the keyhole effect is avoided, navigation is improved with respect to less collisions, taking corners, locating in the environment, and estimating distances and speed. The upper limit of 200° FOV is due to the fact that in one experiment with higher FOV, 205° and 257°, the speed of travel was reduced. Since this was the only experiment with FOV over 200° and the results were negative an upper limit of 200° FOV was set, which also resembles the FOV of the human eye.

When it comes to frame rate, a minimum of 10 FPS is needed to not be affected by significant performance degradation based on the information in Section 2.2.1. However, if possible a frame rate of around 15-30 FPS is recommended for better performance, less stress, and fluctuations in frame time does have a smaller impact on performance at these frame rate. The update rate of the screen is also important to take into consideration when choosing the frame rate, to prevent variation in frame time. If the frame rate is lower than the update rate, the frame rate should be a multiple of the update rate and the other way around if the frame rate is higher. This ensures that the time between each frame is constant.

The resolution of the video feed does not seem to have a significant impact on the tele-operator's performance. This thesis therefore proposes that the resolution is adjusted such that the targeted FOV, frame rate, and latency is reached, since these parameters have a reported more substantial impact on performance.

In Section 2.2.1 it is brought up that if the cameras are placed low towards the ground it presents unnatural viewing angles. It is therefore proposed that the UAV will fly at a height that is equal to the height of the operator.

The use of the tangent point of a curve was brought up in Sec-

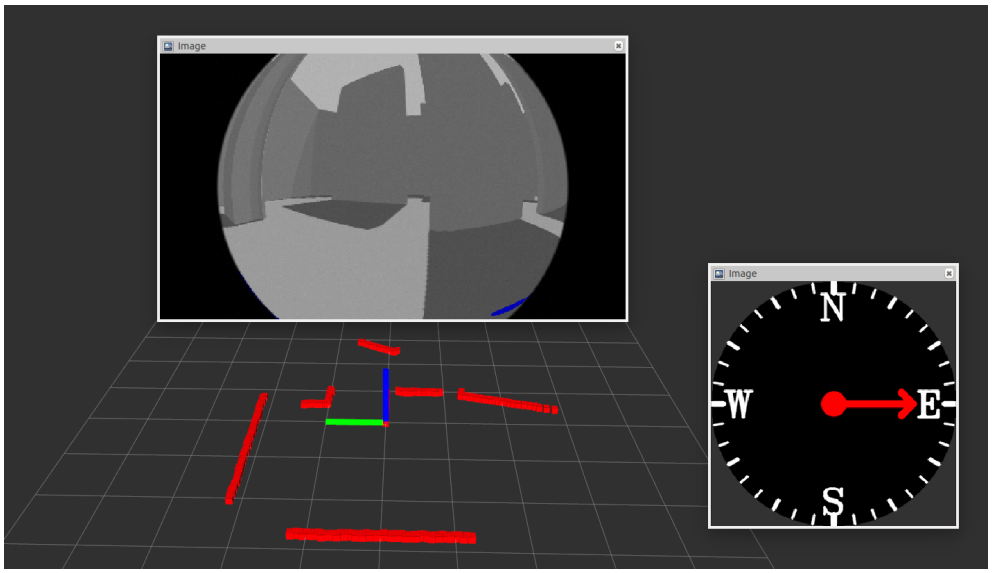


Figure 3.1: The interface presented to the tele-operator.

tion 2.2.1. If only a forward looking camera is present on the UAV it is not possible to make sure that the tangent point of a curve is always in view and at the same time utilize the fact that the UAV is holonomic. To solve this it is proposed that multiple cameras are used to create a 360° view, by stitching the images from these cameras together, or by using a single 360° camera. A region of this view is presented to the operator at a time, such that it complies with the FOV mentioned earlier. The region of the 360° view to display is controlled in two ways. The first is automatic and makes it so that the direction the operator wants to move is always in the center of the view. This is to make it more efficient and simpler to move around since the UAV does not actually have to turn and at the same time the tanget point of a curve is in view. The second is to have the operator control which region of this 360° view to be displayed, this is to increase the operator's awareness of the surroundings since it is possible to quickly look around in the environment.

3.2.2 Map and Compass

As mentioned in Section 2.2.3 maps can be used to give more information about the environment to the tele-operator and help the operator orient in the environment. Since this project assumes that there is no global map of the environment available, only a local map, generated

from the distance sensor on the UAV, is constructed. This map is a track-up map, such that it is rotated so the direction the operator is looking at is always up. The map and camera view is combined into one, since it was learned in Section 2.2.4 that combining them led to best overall performance and they did not have to compete for the operator's attention.

To compensate for the lack of a global map, which is well suited for planning, a compass is used. With the help of the compass the operator can better orient herself/himself when there are a lot of obstacles and plenty of turns has to be made to reach the desired location.

Figure 3.1 displays the complete interface that is presented for the tele-operator.

3.2.3 Haptic Feedback

Haptic feedback is another approach of giving feedback to a tele-operator and has been explored in Section 2.2.5. The main reason for why haptic feedback is used in this project is to make the tele-operator better understand why the UAV is behaving as it does when giving input. In Section 3.1.2 we restricted how much the collision avoidance system could alter the input direction by changing the first part of ORM, based on what was learned in Section 2.2.5. There are however cases where the collision avoidance system presented in this project still will alter the direction with up to 180°. This is the case when the UAV is moving too close to an obstacle. Haptic feedback is used to make the operator aware the UAV gets the input but is unable to move in the wanted direction. The haptic feedback is activated when the output direction from the collision avoidance system differs from the inputted direction with more than ω degrees.

3.2.4 Latency and Processing

From Section 2.2.2 it is clear that latency has a considerable impact on the performance of a tele-operator. To ensure low as possible latency, from a given input to an output, most of the interface will be computed on the computer that the operator is using instead of the computer that is on the UAV. The result of this is that less data has to be sent from the UAV to the operator's computer. For instance the data for the compass can be a single number (instead of an image of a compass) which is

used to create the image of the compass on the operator's computer. The images from the cameras are compressed before being transferred to decrease the size which result in faster transfer.

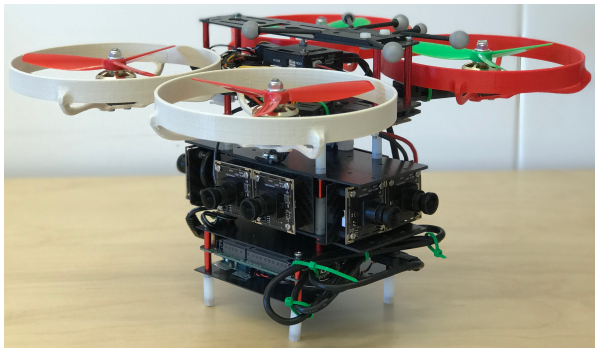
By having the operator's computer generate most of the interface more resources are free on the UAV computer. Stitching images can be a particularly demanding task, by doing it on the operator's computer instead would mean that there are resources left on the UAV computer.

3.2.5 Design of UAV

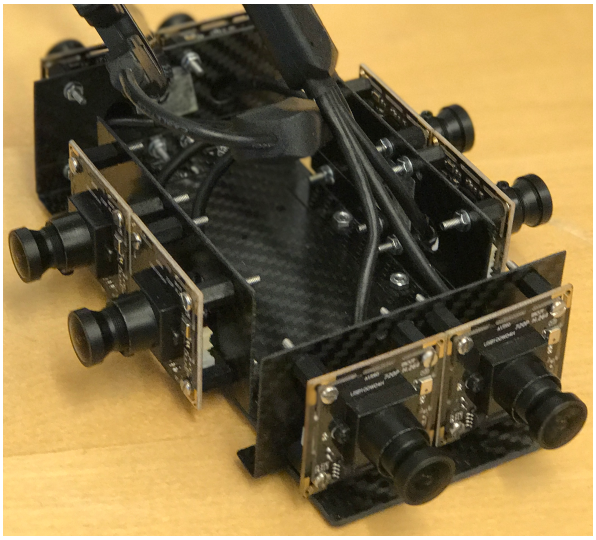
A real UAV, which can be seen in Figure 3.2a, was built that satisfies everything mentioned in this chapter. It has four pairs of cameras pointing at different directions, which can be seen in Figure 3.2b, that are used for both detecting and measuring distances to obstacles and for giving vision feedback to the operator. Detecting and measuring distances to obstacles is done by finding disparities between the images from a pair of cameras.

Each camera has a 170° horizontal FOV. The images from the left camera of each camera pair is stitched together to create the 360° view, such that the operator can look in any direction at any time without having to rotate the UAV.

The reason for this setup is that it provides all information needed and at the same time holds back the weight and size of the UAV. The cameras makes it possible to detect obstacles at different heights. With, for example, a 2D laser range finder it is only possible to detect obstacles in a plane.



(a) The UAV.



(b) The cameras.

Figure 3.2: Illustrates the UAV and camera setup designed for this thesis.

Chapter 4

Experimental Setup

This chapter details the experimental setup.

4.1 Testing Environment

Simulations were performed using Gazebo [22], a robotics simulator, together with Robot Operating System (ROS) [57]. Different testing environments were constructed with simple primitive shapes. The tests were performed using a simulated multirotor UAV.

4.1.1 Sensor Simulation

For the collision avoidance method proposed in this project to function one or multiple sensors to gather distance measurements to obstacle surrounding the UAV is needed. A simulated laser range finder was used for this in the simulation instead of stereo cameras that was mentioned in Section 3.2.5. The laser range finder will be covered in detail below.

The interface also requires a couple of sensors. To generate the map the same laser range finder used for the collision avoidance method is utilized. A simulated inertial measurement unit (IMU) is used for getting the direction the UAV is facing for the purpose of the compass. Simulated cameras are also present and will be covered in detail below.

For positioning the UAV uses simulated Global Positioning System (GPS) and the IMU.

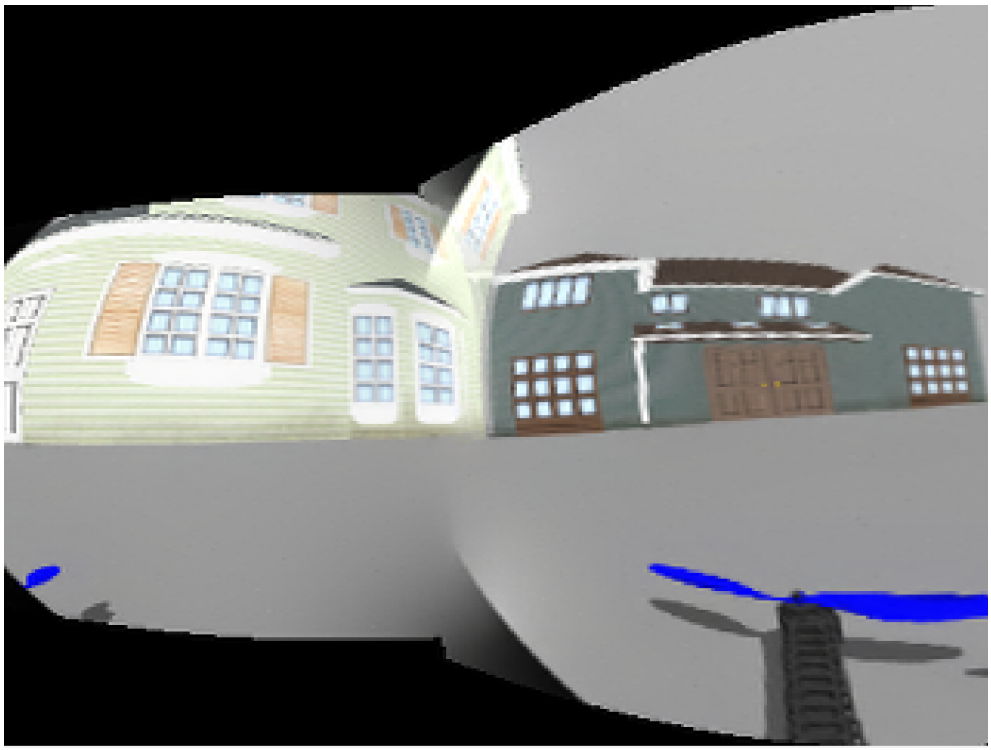


Figure 4.1: Displays artifacts caused by stitching two images.

Cameras

Four cameras, with a 170° horizontal FOV each, was used for this project. The cameras was placed such that there were one camera facing each direction: forward, back, left, and right. This configuration was chosen such that the 360° view mentioned in Section 3.2.1 could be created by stitching the images from the cameras. The stitching processes can cause artifacts as can be seen in Figure 4.1. The cameras had a resolution of 320 by 240 pixels and a update rate of 30 hz.

The reason for the above configuration for the simulation was to make it as similar as possible to the real UAV presented in Section 3.2.5. The cameras were not used for detecting obstacles because there were not enough disparities in the simulated environment for that to function.

Laser Range Finder

A simulated Hokuyo laser range finder, which is popular for use on UAVs, is used for detecting obstacles in the simulator instead of using

cameras as on the real UAV, presented in Section 3.2.5. The laser range finder is configured to give similar obstacle information as the cameras on the real UAV. It is placed above and horizontally in the center of the UAV. The horizontal FOV of the sensor is set to 360° with a resolution of one degree, meaning that 360 simulated laser rays are sent out to measure distances. The minimum and maximum effective range of the sensor is 0.1 meter and 2.5 meters respectively. The update rate was set to 30 hz.

Gaussian noise, with a mean of 0 meter and standard deviation of 0.01 meter, is added to better simulate the real sensor. This means that a 99.7% of the distances registered by the sensor will be within 0.03 meter of the true reading.

4.2 Evaluation

To evaluate the proposed system, it is tested against a set of use cases and in a user study.

4.2.1 Use Cases

The use cases are used to determine if the system is behaving as expected in different situations. Each use case is set in an environment with an initial position and orientation (also known as the pose of the UAV) of the UAV and a constant input command is given to the UAV. From this a desired behaviour, i.e., how the UAV moves and what feedback the operator gets, is defined and tested against. An operator is not needed for the uses cases since the input commands are predefined and does not depend on the feedback that is received from the system. Therefore these use cases are mainly to evaluate the proposed collision avoidance method and that haptic feedback is received when it is supposed to.

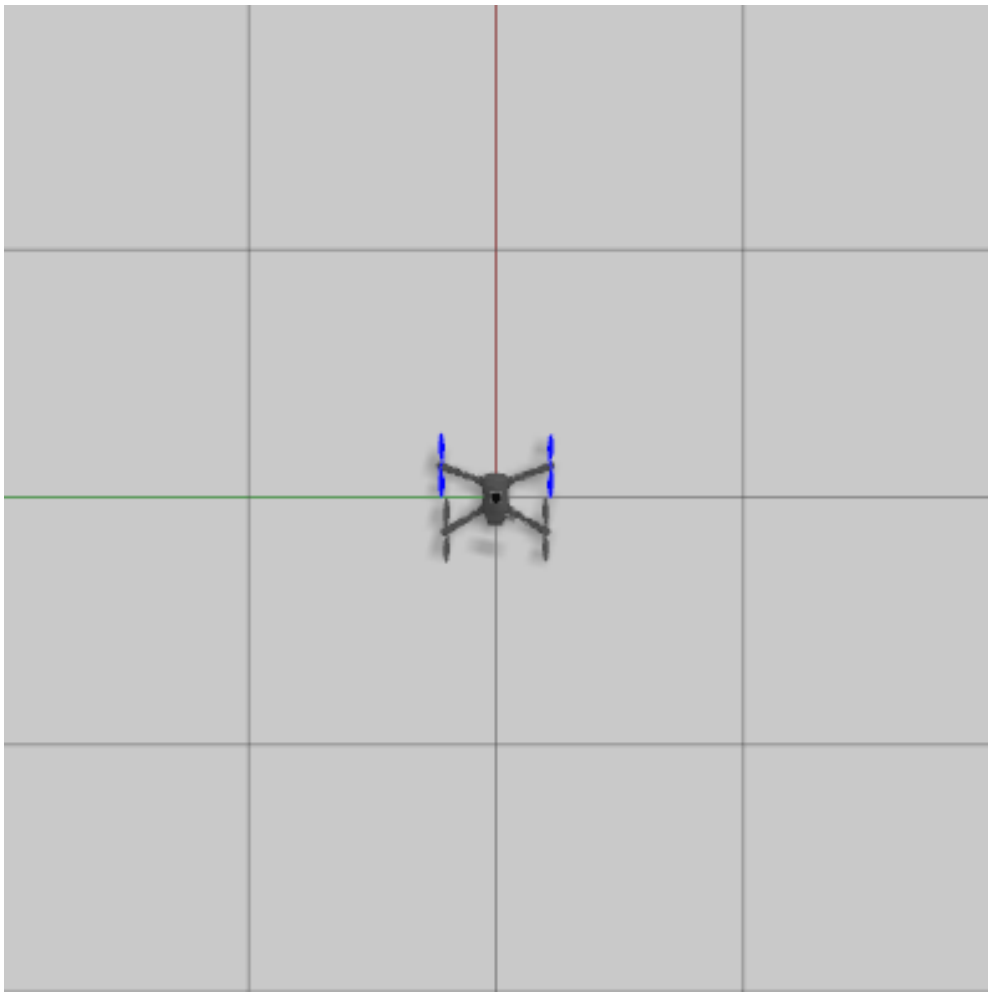


Figure 4.2: The environment in use case 1. There are no obstacles.

Use Case 1: No Obstacles

In this use case, there are no obstacles (see Figure 4.2). The UAV is free to move in any given direction without colliding. Distance sensors have a maximum working range, which means that the situation simulated here can occur even if there are obstacles surrounding the UAV. It is therefore of interest to evaluate the collision avoidance system when there are no sensor readings and to make sure that there are no false positives from the haptic feedback.

The UAV will attempt to move forward, back, left, and right.

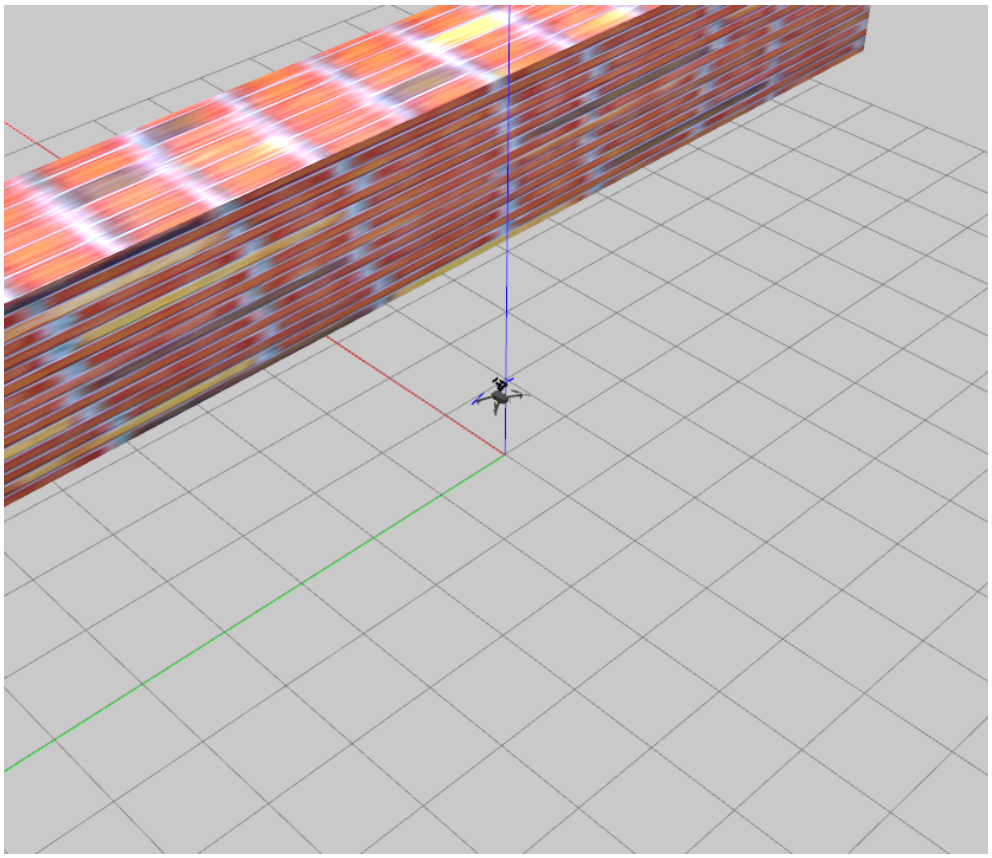


Figure 4.3: The environment in use case 2. The UAV approaches a wide wall and has to stop to avoid collision.

Use Case 2: Move Straight Towards a Wall

In this use case, the UAV is moving straight towards a wide wall (see Figure 4.3). The purpose of this use case is to verify that the collision avoidance system prevents the UAV from directly moving into obstacles. It is also to confirm that the operator receives the appropriate feedback, in form of haptic feedback, when the collision avoidance system prevents the UAV from moving further.

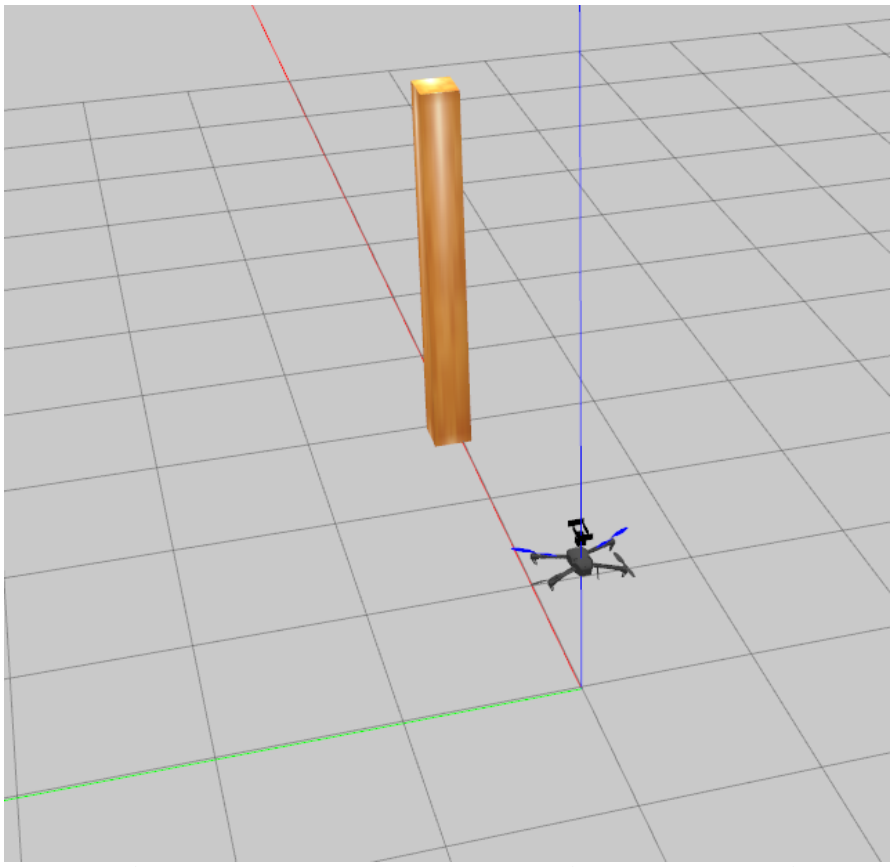


Figure 4.4: The environment in use case 3. The UAV approaches a small obstacle and should either move around it or stop in front of it depending on the inputted magnitude, m.

Use Case 3: Move Towards Small Obstacle

In this use case, there is a small obstacle slightly to the left in front of the UAV (see Figure 4.4). The UAV is commanded to move straight forward towards the obstacle, two times. The first time the UAV is given a magnitude, m, that corresponds to 100% of m_{max}. The second time with a magnitude, m, that corresponds to 25% of m_{max}. The UAV should move around the obstacle the first time, and stop in front of it the second time. The effects of the two parameters, dir_{diff} and opp_{diff}, is primary evaluated in this use case since they are defined as a function of m. 25% of m_{max} was chosen for the second run because it fixes dir_{diff} at 0 with the values for dir_{diff_max} and dir_{diff_min} as specified in Section 4.3.2.

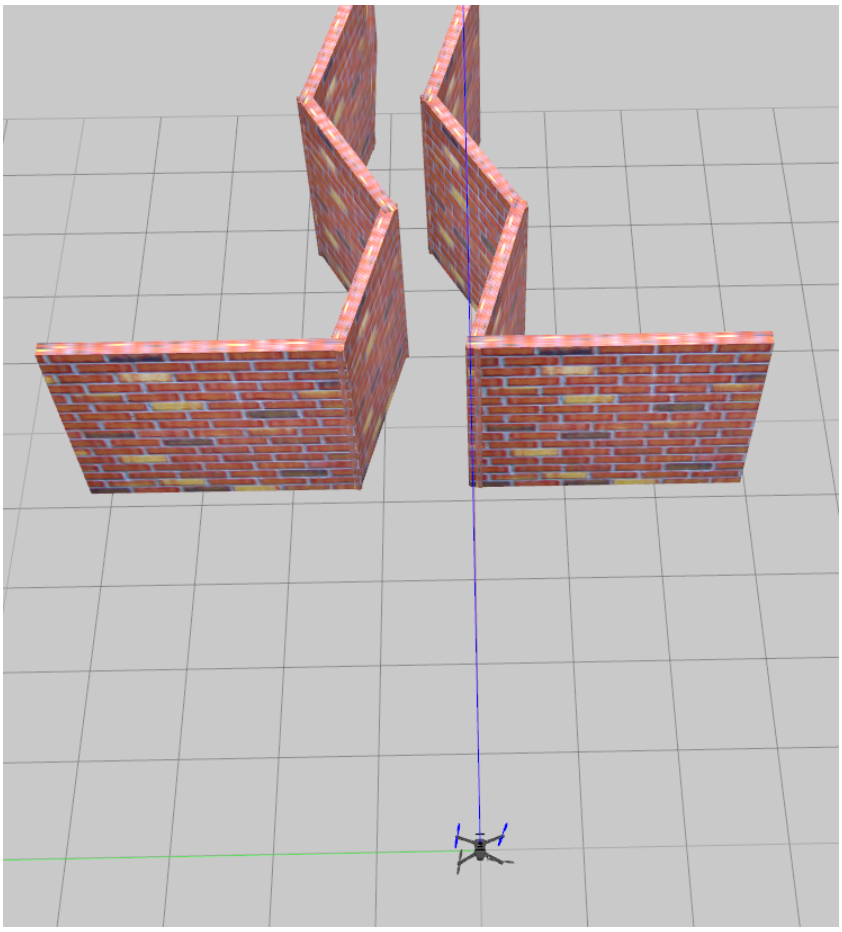
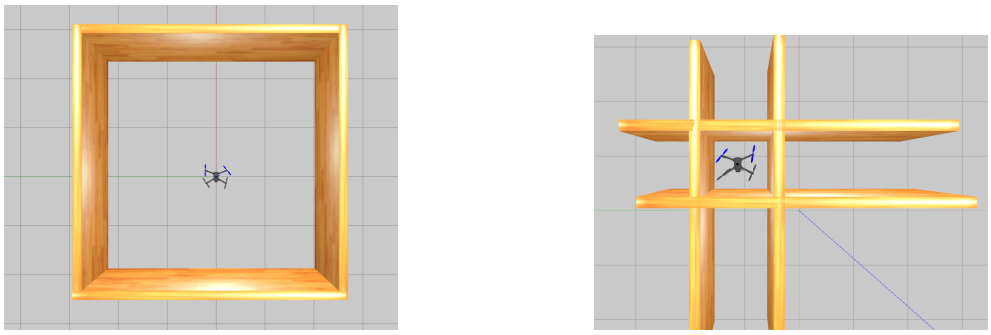


Figure 4.5: The environment in use case 4. The UAV approaches a narrow zigzag corridor and should traverse through it with a static input command.

Use Case 4: Narrow Zigzag Corridor

The UAV is placed in front slightly to the right of the opening of a narrow corridor (see Figure 4.5), in this use case. There are walls on either side of the opening, preventing the UAV from moving outside of the corridor. The corridor itself has a zigzag pattern, meaning the UAV has to alter its direction to traverse it. With a static input command the UAV should be able to enter the corridor and traverse it.

The static input command given to the UAV is straight forward with a magnitude, m, of 100% of m_{max}.



(a) Beginning of use case 5. **(b)** End of use case 5.

Figure 4.6: Illustrates the setup in use case 5.

Use Case 5: No Input

There is no input to the UAV in this use case. The UAV is surrounded by four walls, one on each side (see Figure 4.6a). One wall at the time is moved closer to the UAV, such that the UAV has to move to prevent a collision (see Figure 4.6b). The purpose of this use case is to evaluate the aspect of the developed collision avoidance system presented in Section 3.1.3.

4.2.2 User Study

A user study, with 8 people, is conducted to evaluate the system as a whole. It is used to determine if the performance of real human operators is increased when using the proposed system, based on a number of criteria. The criteria that will be compared are: distance from obstacles, number of collisions, path length, time taken to complete the task, how many that manages to complete the task, and velocity. Given an environment, an initial pose of the UAV, and a goal position, the operator is to traverse the environment from the initial pose to the goal position six times, each time with a different configuration. Six different configurations, seen in Table 4.1, are tested to better isolate which parts of the system that contributes the most.

In configuration 1 and 2 the operator is presented with the whole interface, meaning the track-up map of the nearby environment, the compass, haptic feedback, and the 360° view. In these configurations it is not possible to rotate the UAV, instead the camera view changes such that the operator always sees in the direction in which he/she wants to move. This is to make sure that the tangent point of a curve is always in view and to simplify the controls, such that it is possible to operate the UAV with only a single stick.

Configuration 3 and 4 are meant to replicate the typical tele-operating experience, as mentioned in Section 2.2.1. Therefore, the interface consists of only the camera view from the forward facing camera. In these two configurations it is possible to rotate the UAV using two buttons and one stick is used to move the UAV.

The last two configurations, 5 and 6, also presents the operator with the whole interface, just as with configuration 1 and 2. The difference is that the controls are the same as with configuration 3 and 4, except that the operator also can use a second stick to change the camera view.

The configurations that is numbered with an odd number has the collision avoidance activated to assist the operator, while the even numbered configurations does not. That is the only difference between each pair (1 and 2), (3 and 4), and (5 and 6). This division makes it possible to determine the effect of the collision avoidance system.

The 8 people were assigned the six configurations in a random order, and they had only one attempt with each configuration. They also got a practice round with each of the six configurations on a different map before the real experiment started. After the experiment

they were asked which of the six configurations they preferred the most/least, and what part of the system they found to be most important.

The environment for the user study is presented in Figure 4.7. The UAV in the bottom part of the figure marks the starting location, and the goal is to maneuver the UAV to the top part.

Configuration #	Collision avoidance	Interface	Control
1	Yes	Complete	Look where moving
2	No	Complete	Look where moving
3	Yes	Forward camera	Normal
4	No	Forward camera	Normal
5	Yes	Complete	Normal
6	No	Complete	Normal

Table 4.1: The six different configurations for the user study. Collision avoidance indications if the user is assisted by the collision avoidance system. Complete interface means the user is presented with the map, 360° view, compass, and haptic feedback, otherwise only the camera feed from the forward facing camera is presented. With the control look where moving the camera view changes such that the user always sees in the direction that she/he wants to go.

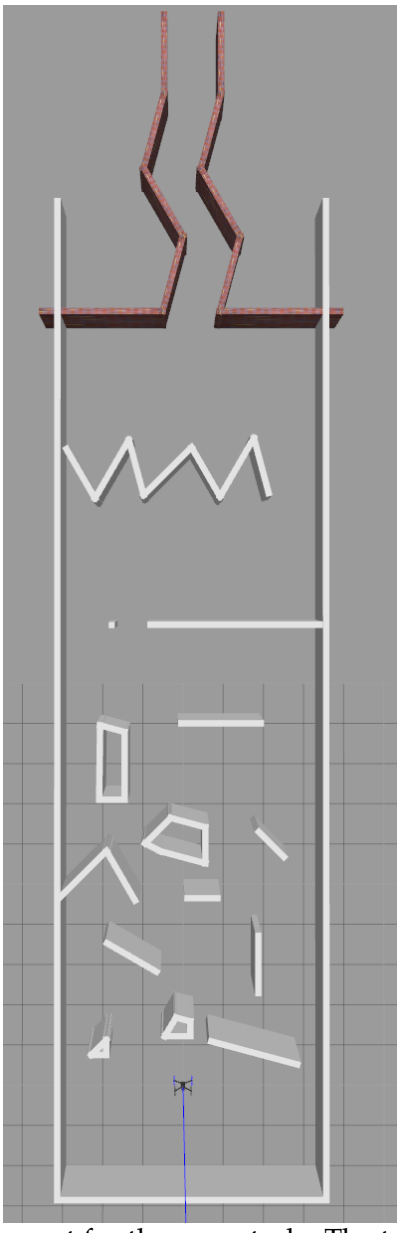


Figure 4.7: The environment for the user study. The task is to move the UAV from the bottom part to the top part.

Figure 4.8: The controller used by the operators.

4.3 Implementation Details

To fully understand how the developed system works and how the human operator controls and receives feedback, a couple of specific implementation details has to be mentioned.

4.3.1 Controls and Haptic Feedback

A Dualshock 4 controller, seen in Figure 4.8, was used in order for the operators to control the UAV in the user study. *LS* and *RS*, in Figure 4.8, refers to the left stick and right stick respectively and *LB* and *RB* refers to the left button and right button respectively.

The controls for each of the six configurations in the user study can be seen in Table 4.2. For all configurations the left stick controls the translational motion of the UAV. The UAV will move in the direction that the left stick is pushed, relative to its heading. For configuration 1 and 2 the left stick also controls the camera view, such that the view presented to the operator is in the direction that the stick is pushed relative to the UAVs heading.

For configurations 1, 2, 5, and 6 the right stick controls the camera view, in similar fashion as the left stick for configurations 1 and 2. If both the left stick and the right stick is pushed simultaneously in configuration 1 and 2 the right stick has higher priority over the camera view, meaning the right stick dictates the camera view in that case.

The left button and right button rotates the UAV around its yaw to the left or right respectively, in configurations 3, 4, 5, and 6. It is not possible to rotate the UAV in configuration 1 and 2.

The Dualshock 4 has the ability to vibrate, which is utilized to give haptic feedback to the operator.

Configuration #	LS	RS	LB	RB
1 & 2	Controls the UAV and the camera view	Controls the camera view	Not used	Not used
3 & 4	Controls the UAV	Not used	Rotates the UAV around its yaw to the left	Rotates the UAV around its yaw to the right
5 & 6	Controls the UAV	Controls the camera view	Rotates the UAV around its yaw to the left	Rotates the UAV around its yaw to the right

Table 4.2: The controls for the six different configurations in the user study.

4.3.2 Default Hyperparameters

There are a number of hyperparameters that affect the behaviour and performance of the developed system. There are hyperparameters that relate to the UAV, collision avoidance system, cameras, and distance sensor, the values used for these hyperparameters in the experiments can be seen in Table 4.3.

For a UAV, such as the one used in this thesis, to move it has to tilt in the direction it wants to move. The more it tilts in a given direction the higher the force towards that direction is, which translates to a higher velocity in that direction. Therefore, if the UAV is moving at a higher velocity it will tilt more than at a lower velocity. If the UAV tilts too much it will cause the distance sensor to point down towards the ground, to prevent this the maximum velocity was set to 2 m/s.

The parameter ω in Table 4.3 is used to decide when haptic feedback should be activated. If the absolute difference between the direction given by the operator and the direction computed by the collision avoidance system is above ω then haptic feedback is activated.

UAV parameters	
Mass	1.535 kg
Width	0.57 m
Length	0.57 m
Height	0.15 m
Radius, R	0.4 m
Max acceleration	3 m/s^2
Max velocity, vel_{max}	2 m/s
Collision avoidance parameters	
Security distance, D_s	0.4 m
dir_{diff_max}	45°
dir_{diff_min}	-15°
opp_{diff_max}	10°
opp_{diff_min}	0°
h_m	1.5 m
Cameras	
Number of cameras	4
FOV per camera	170°
Update rate	30 hz
Resolution	320x240 pixels
Distance sensor	
FOV	360°
Update rate	30 hz
Maximum range	3 m
Minimum range	0.1 m
Resolution	360
Haptic feedback	
ω	70°

Table 4.3: Default hyperparameters for the experiments.

Chapter 5

Results

5.1 Use Cases

In this section the results from the use cases will be presented. There were no collisions in any of the use cases.

5.1.1 Use Case 1

As with the original ORM, the collision avoidance system presented in this thesis performed well in open spaces. Since an omnidirectional distance sensor was used, the UAV was able to move in every direction without any problems. The UAV behaved exactly the same as with the collision avoidance system deactivated. There were no false positives in form of haptic feedback, meaning there were no haptic feedback at all. The only difference compared to without the developed system is that the UAV is actively braking when no input is given.

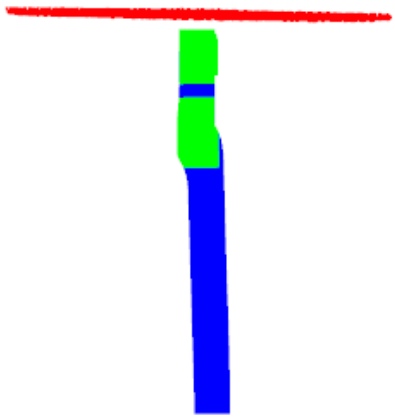


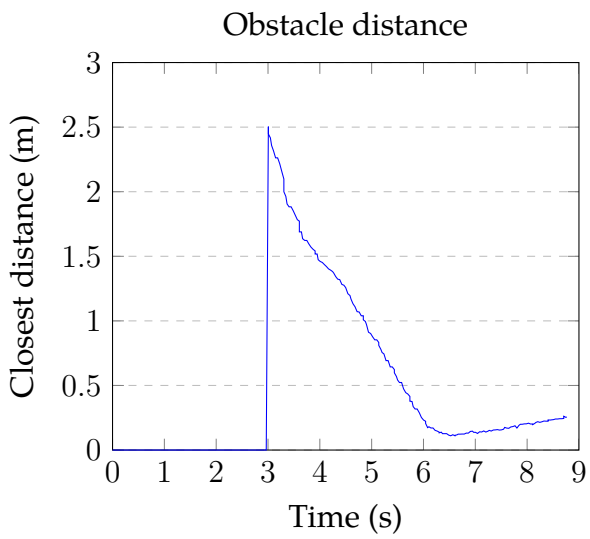
Figure 5.1: The path in use case 2. Red is obstacles detected by sensor, blue is the UAV, and green is when haptic feedback is activated.

Minimum measured obstacle distance (m)	Average minimum measured obstacle distance (m)	Maximum velocity ($\frac{m}{s}$)	Average velocity ($\frac{m}{s}$)	Maximum direction changed (°)
0.11	0.73	1.38	0.63	180.00

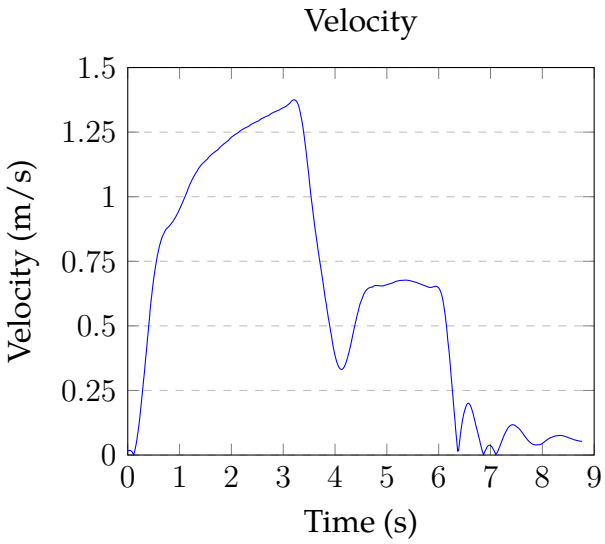
Table 5.1: Results from use case 2.

5.1.2 Use Case 2

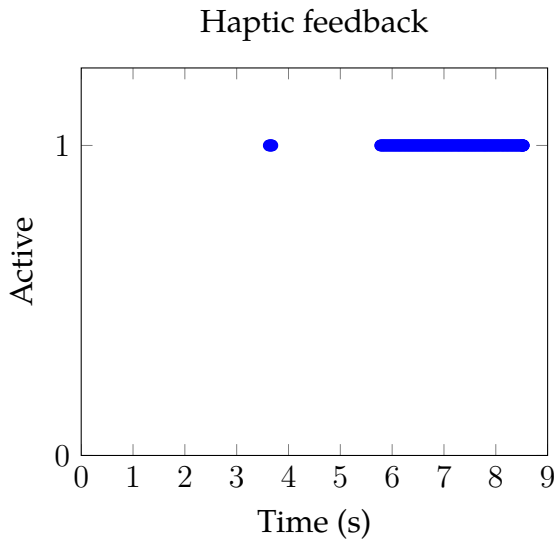
The path the UAV took can be seen in Figure 5.1. The results can be seen in Table 5.1 and Figure 5.2. The UAV increases its velocity up to the three second mark and reaches a maximum velocity of 1.38 m/s, as can be seen in Figure 5.2b. Around the 3s mark the obstacle, in this case in form of a wide wall, is observed by the distance sensor. As the UAV further approaches the obstacle, the velocity decreases. Around the 6s mark the UAV is 0.11 m from the wall, the collision avoidance system prevents the UAV from moving closer to the wall. At the same time haptic feedback is activated.



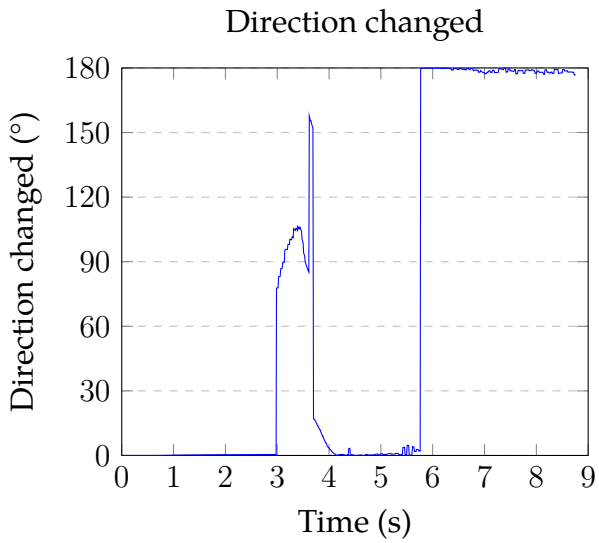
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.



(d) Direction changed over time.

Figure 5.2: Results from use case 2.

Magnitude (%)	Minimum measured obstacle distance (m)	Average minimum measured obstacle distance (m)	Maximum velocity ($\frac{m}{s}$)	Average velocity ($\frac{m}{s}$)	Maximum direction changed (°)
100	0.40	1.02	1.25	0.69	60.61
25	0.24	1.45	0.44	0.32	179.48

Table 5.2: Results from use case 3 with different input magnitudes.

Figure 5.3: The path in use case 3 with 100% magnitude. Red is obstacles detected by sensor and blue is the UAV.

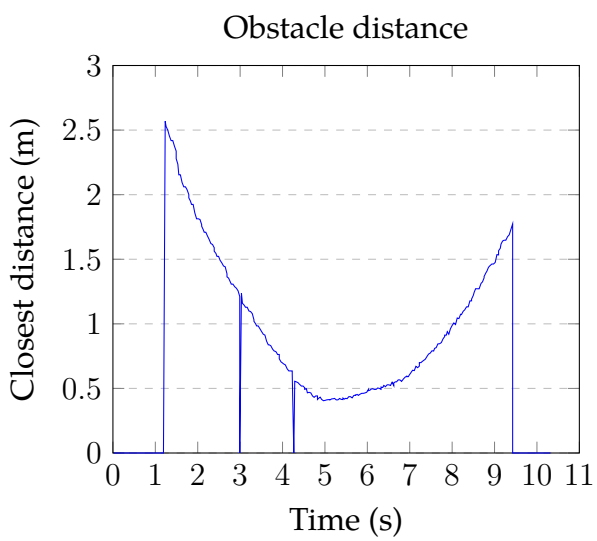
5.1.3 Use Case 3

From Figure 5.3 and Figure 5.4 it can be seen that with a magnitude of 100% the UAV moves around the obstacle, while with 25% magnitude the UAV stops in front of the obstacle. Haptic feedback is activated only with 25% magnitude because the collision avoidance system does not find a way around the obstacle.

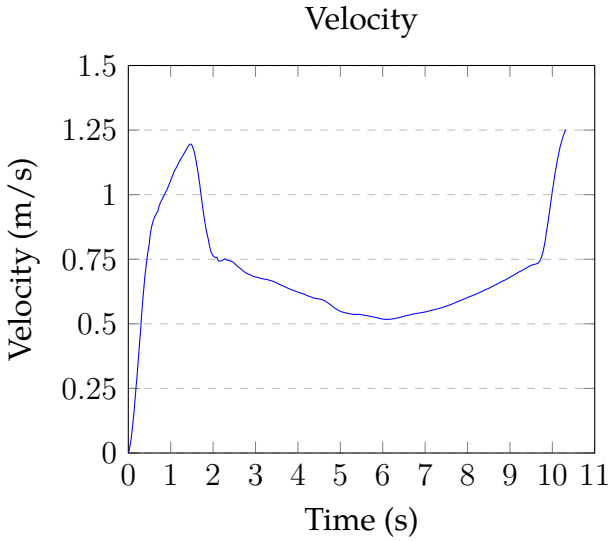
In Figure 5.5 and Figure 5.6, more in depth results are presented. In both Figure 5.5a and Figure 5.6a there are sudden drops, this is believed to be caused by the distance sensor not seeing the obstacle.

Additional results from both 100% and 25% magnitude is presented in Table 5.2.

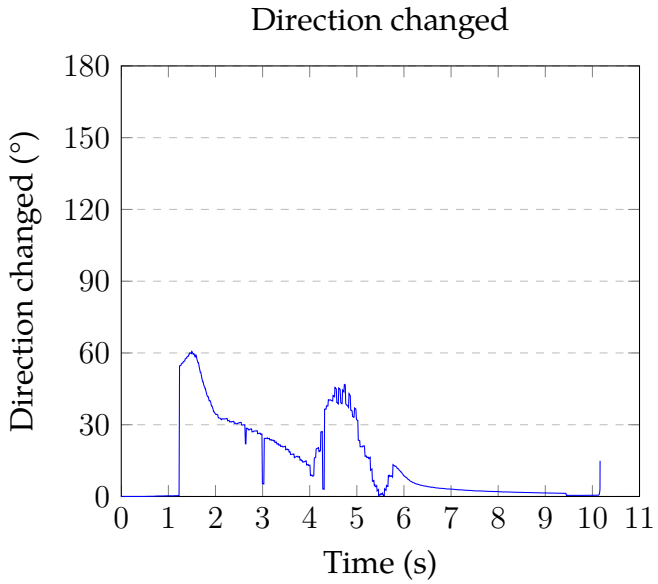
Figure 5.4: The path in use case 3 with 25% magnitude. Red is obstacles detected by sensor, blue is the UAV, and green is when haptic feedback is activated.



(a) The closest obstacle distance from sensor measurements over time.

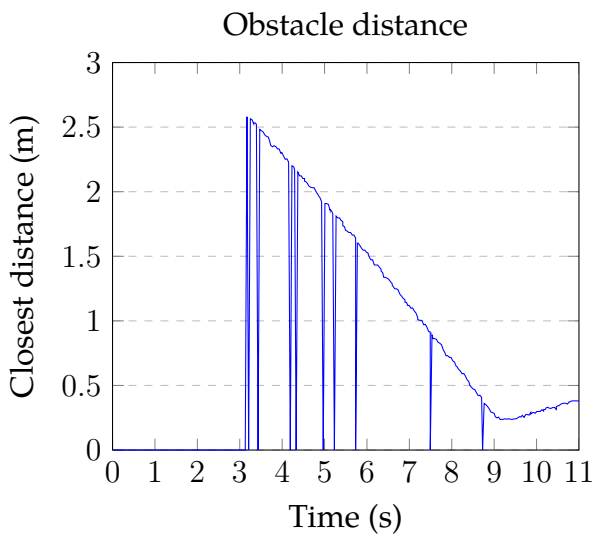


(b) The velocity over time.

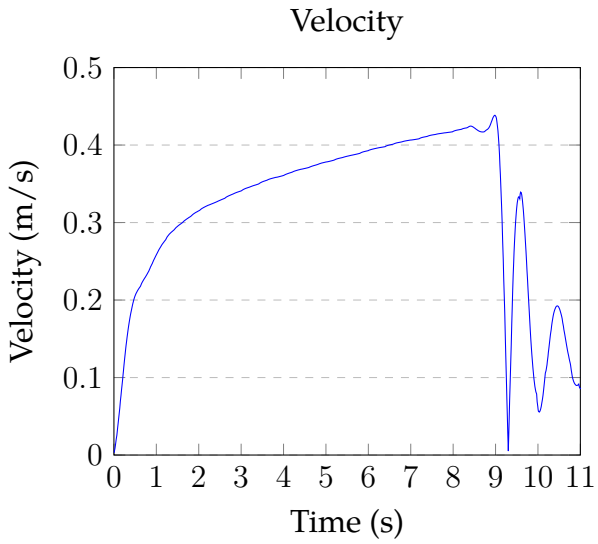


(c) Direction changed over time.

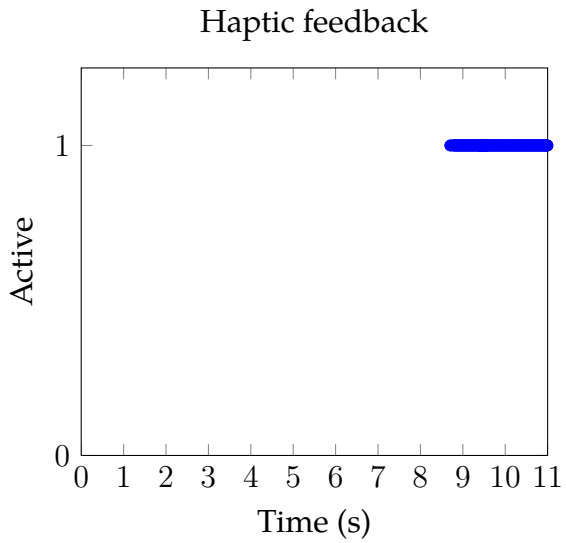
Figure 5.5: Results from use case 3 with 100% magnitude.



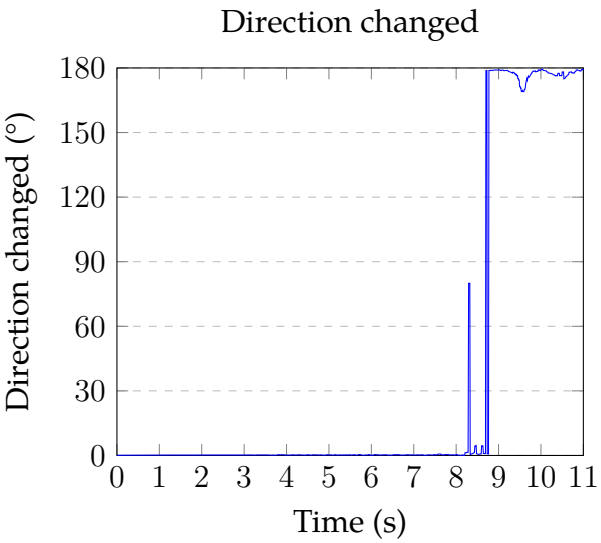
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.



(d) Direction changed over time.

Figure 5.6: Results from use case 3 with 25% magnitude.

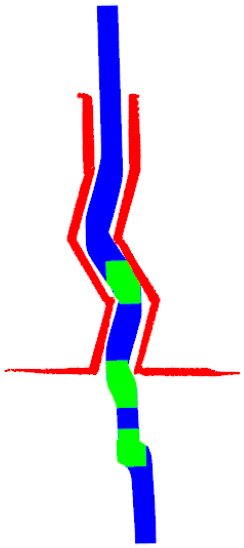


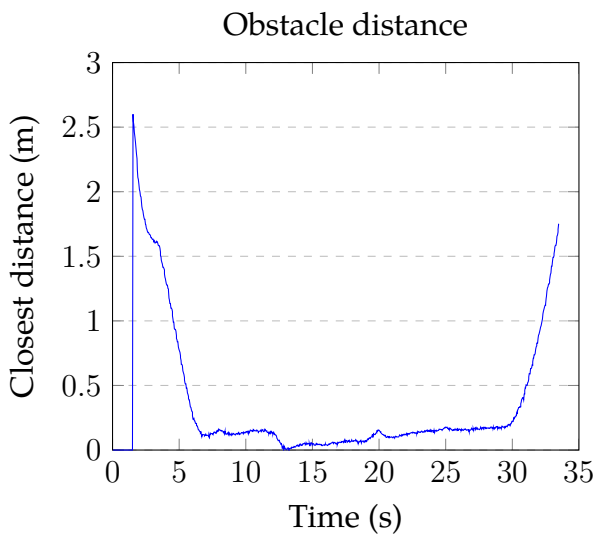
Figure 5.7: The path in use case 4. Red is obstacles detected by sensor, blue is the UAV, and green is when haptic feedback is activated.

Minimum measured obstacle distance (m)	Average minimum measured obstacle distance (m)	Maximum velocity ($\frac{m}{s}$)	Average velocity ($\frac{m}{s}$)	Maximum direction changed (°)
0.01	0.37	1.22	0.42	180.00

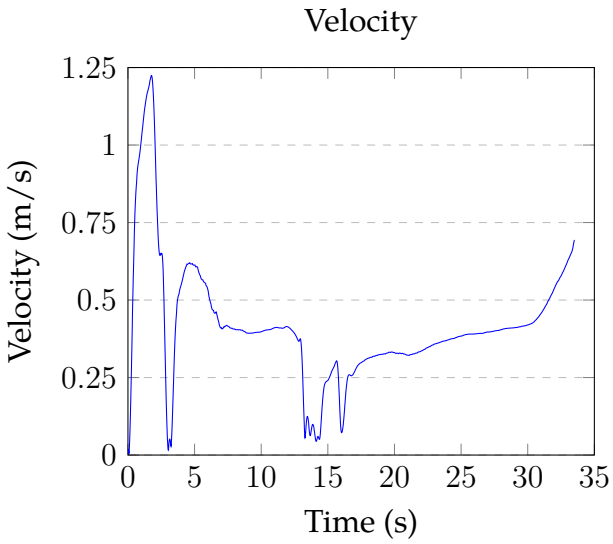
Table 5.3: Results from use case 4.

5.1.4 Use Case 4

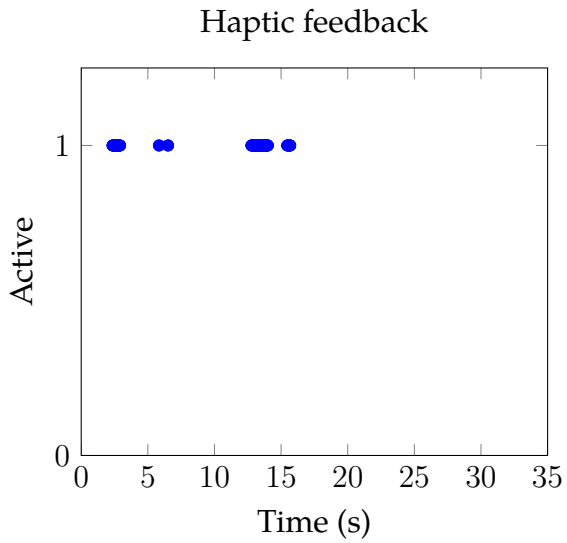
As can be seen in Figure 5.7, the UAV managed to enter and traverse the narrow zigzag corridor with a static input command. Inside the corridor haptic feedback was activated multiple times. More detailed information is presented in Table 5.3 and Figure 5.8.



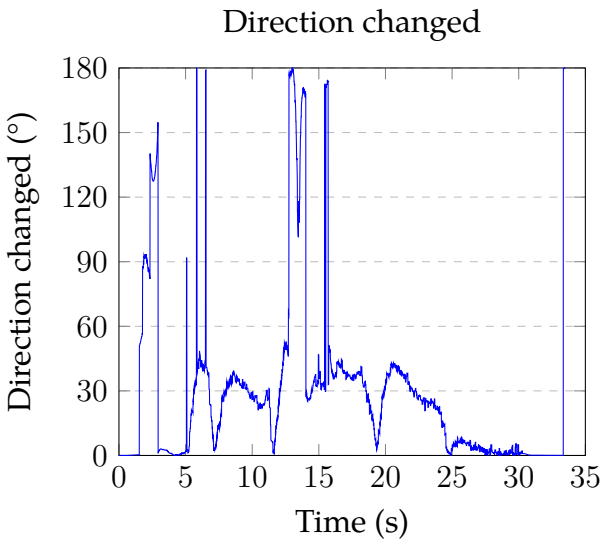
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.



(d) Direction changed over time.

Figure 5.8: Results from use case 4.

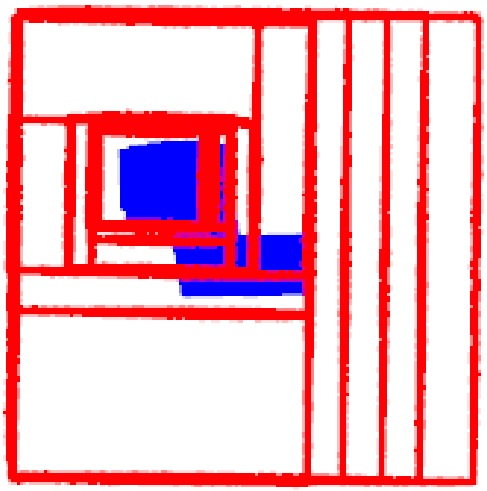
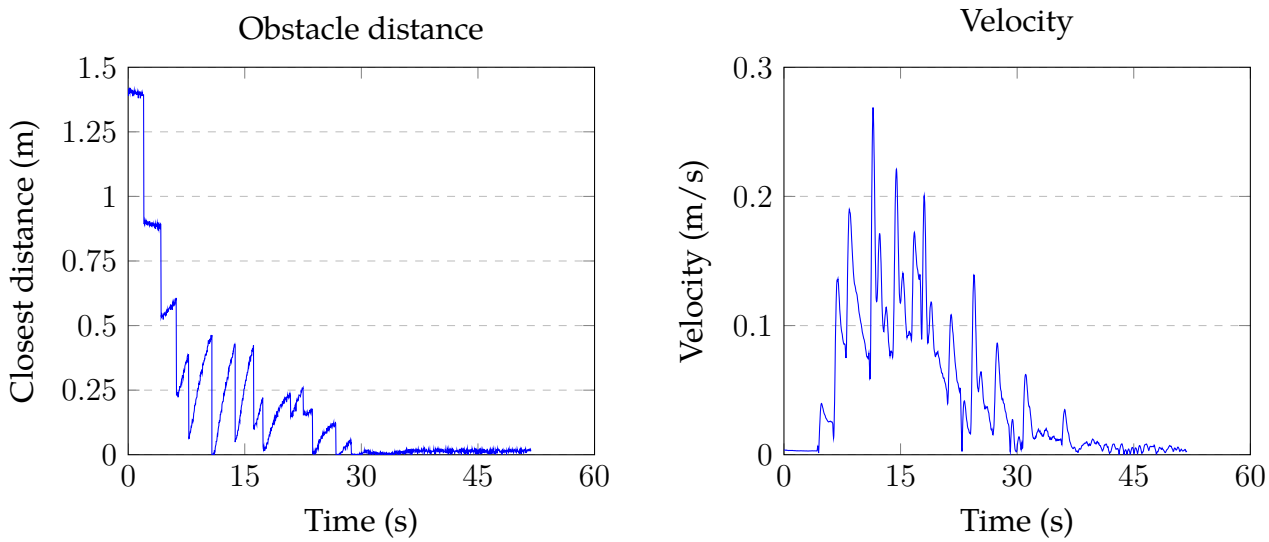


Figure 5.9: The path in use case 5. Red is obstacles detected by sensor and blue is the UAV.

5.1.5 Use Case 5

As seen in Figure 5.9, which can be hard to interpret, the UAV successfully avoided the approaching obstacles by moving from the center towards the top left. In Figure 5.10 and Table 5.4 are more comprehensive information displayed. The reason why the closest distance in Figure 5.10a "jumps" from time to time, and why Figure 5.9 looks the way it does, is because the obstacles are moved instantly from one position to another, meaning there were no continuous motion. Haptic feedback was never activated since no input was given.



(a) The closest obstacle distance from sensor measurements over time.

(b) The velocity over time.

Figure 5.10: Results from use case 5.

Minimum measured obstacle distance (m)	Average minimum measured obstacle distance (m)	Maximum velocity ($\frac{m}{s}$)	Average velocity ($\frac{m}{s}$)
0.00	0.20	0.27	0.05

Table 5.4: Results from use case 5.

Configuration #	# reached goal	Average # collisions	Average path length (m)	Average time taken (s)	Average minimum measured obstacle distance (m)	Average velocity $\left(\frac{m}{s}\right)$
1	**8 / 8**	**0**	40.84	130.68	**0.44**	0.33
2	4 / 8	10.25	37.49	121.28	0.27	0.38
3	**8 / 8**	**0**	**37.26**	111.76	0.43	0.34
4	**8 / 8**	9	42.19	110.93	0.34	0.47
5	**8 / 8**	**0**	37.82	111.74	0.41	0.34
6	**8 / 8**	4.25	43.04	**103.55**	0.38	**0.50**

Table 5.5: Results from the user study. The values in bold are the best in that column.

5.2 User Study

Table 5.5 presents results from the user study. There were no collisions with the collision avoidance system active. Configuration 6 had the least amount of collision of the configurations where the collision avoidance system was inactive. Configuration 2 was the only configuration in which not all operators were able to reach the goal. Only half of the operators, operators 1, 2, 6, and 8, managed to reach the goal, the other half either got stuck to a wall or flipped upside down.

The configurations with the collision avoidance system active had the shortest average path length, the result for configuration 2 can be ignored since not all reached the goal and therefore moved a shorter distance. With the collision avoidance system active the average minimum measured distance to the closest obstacle was also higher than with it inactive. However, the velocity was higher and the time taken to reach the goal was lower without the collision avoidance system.

Figure 5.11 displays how many collisions each operator had with the different configurations. The configurations with the collision system active is not present since there were no collision with them. For all except one operator, configuration 6 has the least amount of collisions.

In Figure 5.12 the path length for each operator with each config-

uration is shown. It can be seen that for operator 3, 5, and 7 the path is noticeable shorter with configuration 2, this is most likely because these three did not reach the goal. Noteworthy is that for 6 of the 8 operators the configurations with the collision avoidance active have similar path length, while the other configurations are more spread out.

The average minimum obstacle distance is presented in Figure 5.13. The configurations with the collision avoidance system active are in most cases the ones that are furthest from obstacles. However, configuration 6 has similar results for multiple operators.

The time taken and average velocity is depicted in Figure 5.14 and Figure 5.15. The three configurations with collision avoidance show similar results here. The configurations that stands out is configuration 6 which has the highest average velocity for most of the operators and the lowest time to reach the goal.

As mentioned in Section 4.2.2 the 8 people were asked a couple of questions after the experiment. All 8 said that the map and the collision avoidance system was the most important part of the system. No one said they used the compass. Most preferred configuration 6. Configuration 2 was the least preferred with the arguments that they got confused by the camera view changing, the artifacts due to the stitching process as mentioned in Section 4.1.1 and shown in Figure 4.1, and lastly because the UAV did not actively break, meaning it would drift when the operator stopped giving an input. The only complain about the collision avoidance system was that it made the UAV move slower, and one person said it made it impossible to go through some narrow passages that the UAV otherwise is capable of.

Additional results from the user study can be found in Appendix A.

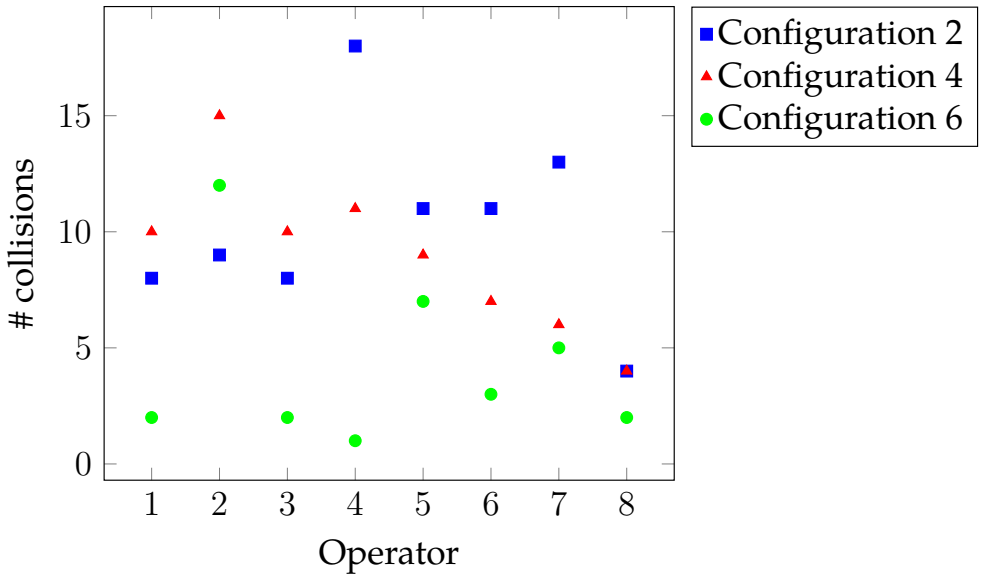


Figure 5.11: Number of collisions per operator for the configurations with the collision avoidance inactive in the user study. There were no collisions with the collision avoidance activated.

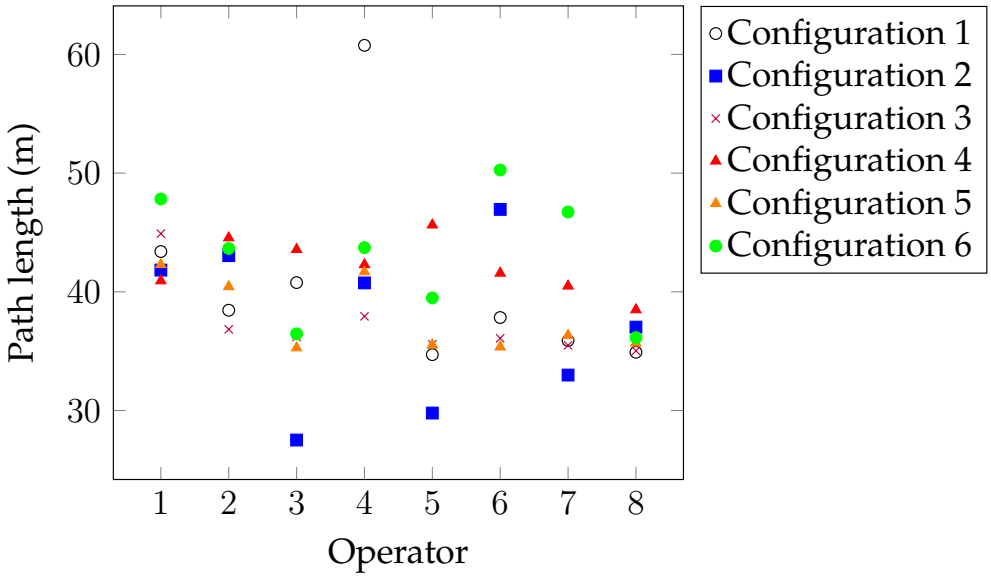


Figure 5.12: Path length in meters per operator for all six configurations in the user study.

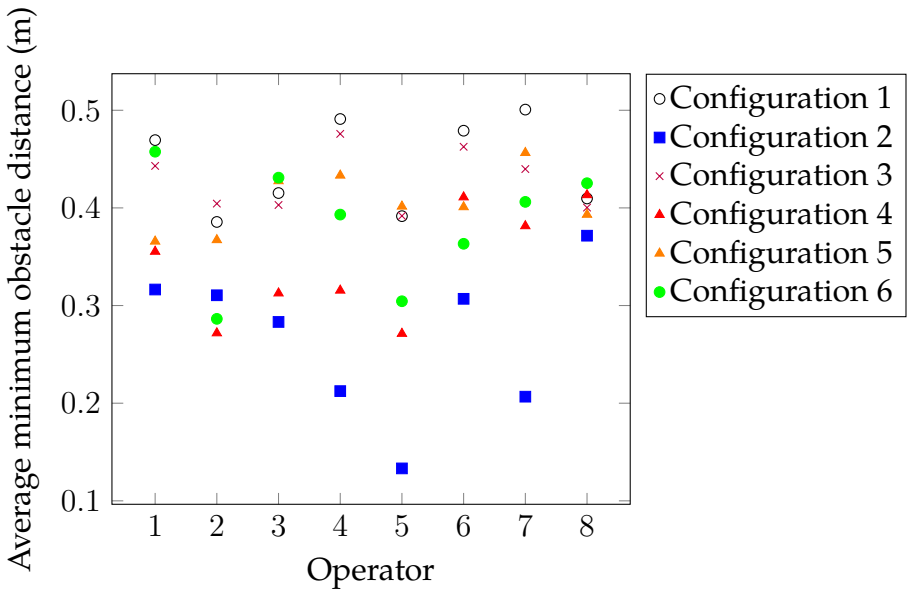


Figure 5.13: Average minimum distance in meters to closest obstacle per operator for all six configurations in the user study.

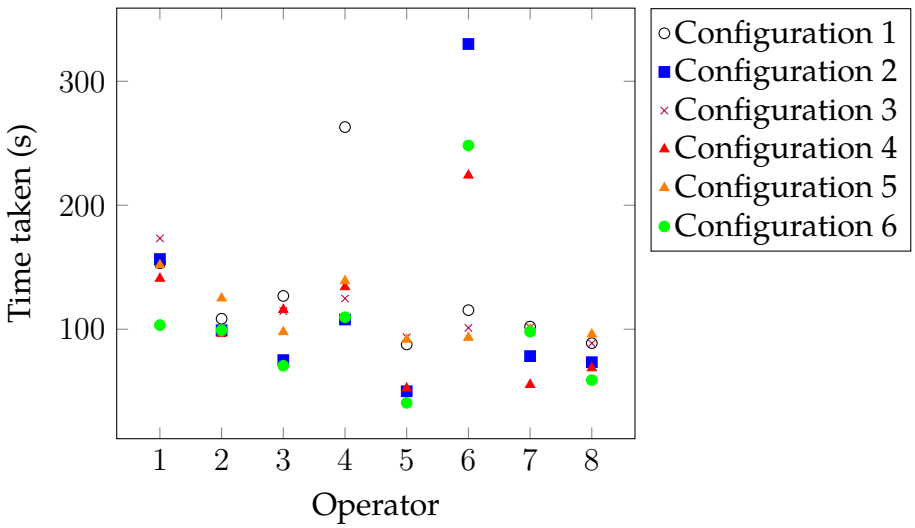


Figure 5.14: Time taken in seconds per operator for all six configurations in the user study.

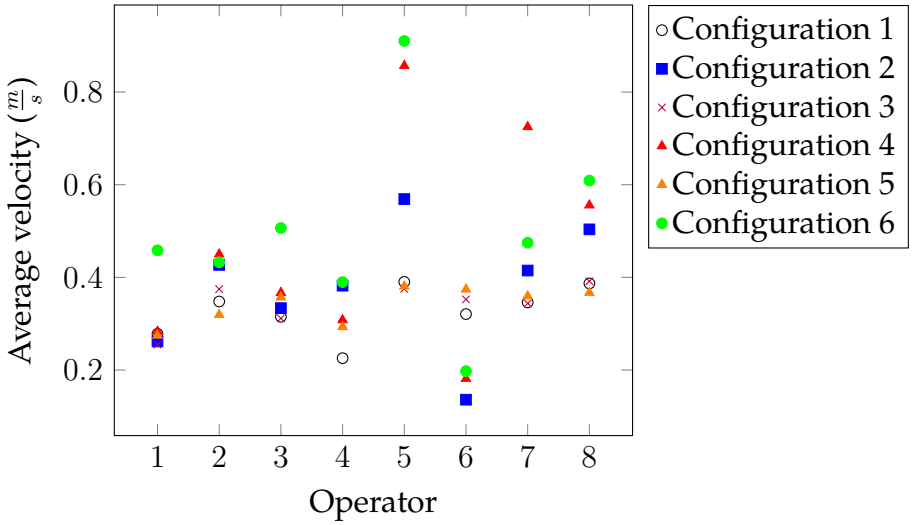


Figure 5.15: Average velocity in meters per seconds per operator for all six configurations in the user study.

Chapter 6

Discussion

In this chapter, the results and findings of this thesis are discussed. First a number of error sources will be mentioned and the possible effect on the results that these might have will be discussed. After that the use cases will be discussed, and last the user study.

6.1 Error Sources

The UAV uses the IMU for calculating the velocity, positioning, and so forth. If the UAV collides, the IMU can be perplexed and think the UAV is moving in a way it is not. This can lead to the total path and velocity in Section 5.2 being wrong if a collision has occurred. This can be why the average velocity and the average path length is higher for the configurations *without* the collision avoidance system.

The distances to the obstacles is measured using the distance sensor mounted on the UAV. Since this sensor does not give ground truth, see Section 4.1.1, the results might not be correctly depicted with respect to obstacle distance.

As seen in Figure 4.7, all the walls in the environment used for the user study was white, except the end, and the ground gray. This can make it unnecessarily hard to navigate through it, especially in the case without the full interface since the video feed is the only feedback from the environment.

6.2 Use Cases

The purpose of the use cases were mainly to showcase and evaluate all aspects of the proposed collision avoidance method. However, the haptic feedback portion of the human-UAV interface was also evaluated in the use cases, since it is closely coupled with the collision avoidance system. Based on the results in Section 5.1 it can be concluded that the collision avoidance system developed in this thesis functions as intended. The following sections discuss each use case in depth.

6.2.1 Use Case 1

In use case 1 the system was tested in an open area. The UAV acted as if the collision avoidance system was not enabled. This is the desired behaviour, since there are no reason for the collision avoidance system to make any adjustments to the inputted commands. One can argue that if the collision avoidance system had had an impact in an open area, then one or more parameters are not correctly set. An example would be if the velocity were too high such that the collision avoidance system has to slow the UAV down. Then the maximum velocity has to be changed. If the UAV cannot maintain that velocity in an open area, then it will not be able to when there are obstacles around.

6.2.2 Use Case 2

Use case 2 demonstrated the ability of the system to prevent the UAV from colliding into an obstacle, even head on. This is an especially important aspect when it comes to collision avoidance systems for UAVs controlled by human tele-operators. A human tele-operator might unwillingly/unknowingly or on purpose steer the UAV towards an obstacle. It can be that the connection to the UAV is lost for a couple of seconds, or that the operator focuses on something else.

For new operators that has no previous experience, this means that they do not have to worry as much about avoiding collisions. Instead they can focus on learning the controls and how the UAV behaves.

With fully autonomous systems this situation is less likely to occur since it is, or should be, aware of all information available about the environment and does exactly as the collision avoidance system wants at all time.

6.2.3 Use Case 3

The parameters dir_{diff} and opp_{diff} were introduced to make it possible for the operator to both perform exploration and investigate objects. In use case 3, it can be seen that this in fact work.

A situation where this is especially useful is in search and rescue missions where only the general location of a person is known. In such a situation the exploration aspect can be utilized first to quickly get to the general location of the person. At that location the investigation aspect can be used to find the person in that area.

6.2.4 Use Case 4

In use case 4, the UAV successfully traverses the narrow zigzag corridor with a static input command. This demonstrates that with the proposed system the operator does not have to focus entirely on how to move the UAV. Instead the operator can look around or plan where to go next. The haptic feedback makes the operator aware of when the UAV cannot move in the desired direction and thus the operator might have to focus more on directing the motion towards the desired location.

6.2.5 Use Case 5

A situation that collision avoidance methods for autonomous UAVs often does not take into account is when there is no input to the UAV, which is in focus in use case 5. The reason why collision avoidance methods for autonomous UAVs often does not consider this case is because the UAV will always have somewhere to move or else it is done. However, when tele-operating the connection can be lost for a couple of seconds as mentioned earlier in Section 6.2.2, and also the operator might need some time to think and therefore does not input a new command. The latter can especially be the case for new operators that has not had any training, and might therefore not be familiar with how a UAV and the controls functions.

6.3 User Study

From the user study it can be concluded that the proposed system with collision avoidance results in safer navigation, with 0 collisions for all three configurations. The UAV also travels at a safer distance from obstacles and the total path is shorter with the developed collision avoidance system.

With the collision avoidance system both the maximum and average velocity was lower, and therefore it takes longer to complete the task. A reason for this is that it is simpler for a human to control the velocity since she/he knows where the long term goal is. The collision avoidance system on the other hand has to be prepared for any kind of movement, which results in the velocity being decreased when there are obstacle near the UAV.

Another drawback found with the collision avoidance system is that it makes it impossible to go through some narrow passages. However, this can be adjusted but there is a trade-off between safety and maneuverability. In the real world it is not possible to have as perfect of position as in the simulator. A UAV can not move just a couple of centimeters or millimeters from an obstacle safely in the real world, if the air changes or their is a small error in the position the UAV is likely to hit the obstacle.

One way of increasing the velocity would be to only consider obstacles in the direction of motion when calculating the velocity. However, in this thesis this was not opted for because there are more drawbacks than benefits from doing this. Imagine moving down a narrow corridor, the walls are close to the UAV on both sides but there are no obstacles in front of the UAV so the velocity is high. Suddenly a door-opening appears on one side of the corridor, even if the operator would try to move through the door-opening it would not be possible since the velocity is too high in the orthogonal direction, so the UAV moves passed it.

In the real world the environment is dynamic, meaning obstacles can move and suddenly appear from behind corners. If instead of the operator wanting to move through the door-way in the narrow corridor, a person came out from the door-way. Then the UAV would not be able to stop in order to avoid a collision if it is close enough and the velocity is only depended on the obstacles in front of it. With the velocity dependeding on the closest obstacle at all time the UAV would

more likely be able to avoid a collision.

The fact that configuration 6, which has the full interface but without the collision avoidance and the changing camera view, performs better in every aspect compared to configuration 4 which only has the forward facing camera view proves how useful the developed interface is. It can safely be said that the changing camera view is not to recommend, at least in the way it was implemented here. It is the worst configuration without the collision avoidance and only half of the operators manage to reach the goal with that configuration. Configuration 1 proves the usefulness of the collision avoidance system since this configuration also has the changing camera view, but everyone still manages to reach the goal without a single collision. However, it falls short to both of the other configurations with collision avoidance in all aspects except average minimum measured obstacle distance, which it is the best with a centimeter.

From Table 5.5 it can be seen that the difference between configuration 3 and 5 is insignificant. One would think that the difference would be similar to that of configuration 4 and 6 since the two pair only differs in that the former of each pair has only the forward facing camera view while the latter has the complete interface. The explanation to this is that the collision avoidance system, that is present in configuration 3 and 5, lowers the velocity such that the operator can not take advantage of the additional information to complete the task faster.

The fact that the operator collided many times when trying to complete the task without the collision avoidance system demonstrates how useful it is. UAVs can be very fragile meaning that any of the collisions that occurred without the collision avoidance system could have been fatal for the task in the real world, which it was four times in the user study.

Chapter 7

Conclusion

The purpose of this thesis was to develop a system that allowed for safe navigation of a UAV in an indoor environment. In order to do so, a collision avoidance method that was optimized for human operators were introduced and tested. Together with the collision avoidance method a human-UAV interface was developed to increase the operators awareness of the environment, such that the operator can better localize and find where to move.

The proposed system was tested against a number of use cases and a user study with different configurations. The use cases were completed as desired, which proves that the system is capable of handling most situations. Situations such as: approaching an obstacle head on, move close to inspect an object, do exploration without having to focus on avoiding collisions, the UAV moves away from approaching obstacles, and will avoid obstacles even when no input is present. Use case 2 and 5 also showed that the proposed system makes it easier for operators with no previous experience, since they do not have to worry as much about colliding into obstacles.

The user study demonstrated that the complete proposed system ensures a safer navigation compared to systems that are often used. The proposed human-UAV interface, which is based on previous research, makes the operator more aware of the surroundings. However, the changing camera view aspect of the human-UAV interface was worse than only using a forward facing camera.

The collision avoidance system was successful in that no collisions occurred with it activated for all three configurations. The full human-UAV interface without the collision avoidance system and without the

changing camera view, configuration 6, showed that the interface has a positive impact on the performance of the operator compared to only having a forward facing camera view as interface.

From all this it can be concluded that the question investigated in this thesis (see Section 1.1) has successfully been answered.

7.1 Future Work

The drawback of the proposed collision avoidance method is the decreased velocity, it would be of interest to explore this further to be able to increase the velocity, especially since this was believed to be the reason why configuration 5 did not perform better than configuration 3, even though configuration 6 performed better than configuration 4.

By extending the collision avoidance method, with ideas from [66], to consider a 3D environment, the full potential of the UAV can be utilized. This would make it possible for the UAV to reach places which are not possible with the current system. In a situation such as the one describe in use case 5 (see Section 4.2.1) the UAV would be able to escape by flying over the approaching walls.

In Section 2.2.1, it was briefly mentioned that the placement of the cameras can have an impact on the operator, since it can present an unnatural view of the environment. It would be of interest to do research about this to determine the optimal altitude for tele-operating a UAV. To fly at an altitude such that the cameras are at the same height as the averaged human eyes might lead to serious injuries if the UAVs and humans share the same environment.

The proposed system does not take into account the shape of the UAV as it considers it as a sphere. This means that the collision avoidance system will not let the UAV pass through openings with a width that is smaller than the radius of the UAV, even though the UAV would fit through if it was oriented correctly. It is therefore of interested to extend this system to consider the shape of the UAV. Since an omnidirectional view is present it is possible to implement this in at least two ways. One would be to let the operator manually rotate the UAV. The other would be to automatic rotate the UAV and counter change the view with respect to the rotating, such that the operator does not have to be aware and concerned about it.

The proposed system is constrained to move in the horizontal plane.

It would be of interest to study how this can be extended to 3D, initially by releasing the constraint of a fixed flying height. Furthermore, and most importantly as the next step is to validate the approach with real-world experiments. This requires a reliable positioning system. Initial experiments could be performed using a motion capture system but this limits the area in which such experiments can be conducted.

Bibliography

[1] Anthony J. Aretz. "The design of electronic map displays". In: *Human Factors: The Journal of the Human Factors and Ergonomics Society* 33.1 (1991), pp. 85–101.

[2] Kevin W. Arthur, Kellogg S. Booth, and Colin Ware. "Evaluating 3d task performance for fish tank virtual worlds". In: *ACM Transactions on Information Systems (TOIS)* 11.3 (1993), pp. 239–265.

[3] Kumar Bipin, Vishakh Duggal, and K. Madhava Krishna. "Autonomous navigation of generic monocular quadcopter in natural environment". In: *2015 IEEE International Conference on 2015 IEEE International Conference on*. IEEE. 2015, pp. 1063–1070.

[4] J. Borenstein and Y. Koren. "High-speed Obstacle Avoidance for Mobile Robots". In: *Proceedings IEEE International Symposium on Intelligent Control*. 1988, pp. 382–384.

[5] Johann Borenstein and Yoram Koren. "Real-time Obstacle Avoidance for Fast Mobile Robots in Cluttered Environments". In: *Proceedings of the IEEE International Conference on Robotics and Automation*. 1990, pp. 572–577.

[6] Johann Borenstein and Yoram Koren. "The Vector Field Histogram - Fast Obstacle Avoidance for Mobile Robots". In: *IEEE Transactions on Robotics and Automation* 7.3 (June 1991), pp. 278–288.

[7] Johann Borenstein and Yorem Koren. "Real-Time Obstacle Avoidance for Fast Mobile Robots". In: *IEEE Transactions on Systems, Man, and Cybernetics* 19.5 (Sept. 1989), pp. 1179–1187.

[8] Ulrich Borgolte et al. "Architectural Concepts of a Semi-autonomous Wheelchair". In: *Journal of Intelligent and Robotic Systems* 22.3 (1998), pp. 233–253. DOI: 10.1023/A:1007944531532. URL: http://dx.doi.org/10.1023/A:1007944531532.

[9] J. L. Campbell, C. Carney, and B. H. Kantowitz. *Human Factors Design Guidelines for Advanced Traveler Information Systems (ATIS) and Commercial Vehicle Operations (CVO)*. Aug. 2016. URL: https://www.fhwa.dot.gov/publications/research/safety/98057/ch05.cfm.

[10] Stephen M. Casner. "The effect of GPS and moving map displays on navigational awareness while flying under VFR". In: *International Journal of Applied Aviation Studies* 5.1 (2005), pp. 153–165.

[11] Jennifer Casper and Robin R. Murphy. "Human-robot interactions during the robot-assisted urban search and rescue response at the world trade center". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 33.3 (June 2003), pp. 367–385.

[12] Jessie Y. Chen et al. *Human-robot interface: Issues in operator performance, interface design, and technologies*. Tech. rep. DTIC Document, 2006.

[13] Rudolph P. Darken and Helsin Cevik. "Map usage in virtual environments: Orientation issues". In: *Proceedings IEEE Virtual Reality*. IEEE. 1999, pp. 133–140.

[14] Rudolph P. Darken and Barry Peterson. *Spatial Orientation, Wayfinding, and Representation*. 2001.

[15] Fred W. DePiero, Timothy E. Noell, and Timothy F. Gee. "Remote Driving With Reduced Bandwidth Communication". In: *Proceedings of the Sixth Annual Space Operation, Application, and Research Symposium (SOAR'92): Houston, TX*. 1992.

[16] Wolfram Burgard Dieter Fox and Sebastian Thrun. "The Dynamic Window Approach to Collision Avoidance". In: *IEEE Robotics & Automation Magazine* (1997), pp. 23–33.

[17] *DJI*. Accessed: 2017-05-10. URL: https://www.dji.com/.

[18] Stephen R. Ellis et al. "Generalizeability of latency detection in a variety of virtual environments". In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. Vol. 48. 23. SAGE Publications Sage CA: Los Angeles, CA. 2004, pp. 2632–2636.

[19] Stephen R. Ellis et al. "Sensor spatial distortion, visual latency, and update rate effects on 3D tracking in virtual environments". In: *Proceedings Virtual Reality*. IEEE. 1999, pp. 218–221.

[20] William R. Ferrell. "Remote manipulation with transmission delay". In: *IEEE Transactions on Human Factors in Electronics* 1 (1965), pp. 24–32.

[21] Lawrence H. Frank, John G. Casali, and Walter W. Wierwille. "Effects of visual display and motion system delays on operator performance and uneasiness in a driving simulator". In: *Human Factors* 30.2 (1988), pp. 201–217.

[22] *Gazebo*. Accessed: 2017-06-01. URL: http://gazebosim.org/.

[23] Michael A. Goodrich et al. "Supporting wilderness search and rescue using a camera-equipped mini UAV". In: *Journal of Field Robotics* 25.1-2 (2008), pp. 89–110.

[24] Soonshin Han and JangMyung Lee. "Tele-operation of a mobile robot using a force reflection joystick with a single hall sensor". In: *The 16th IEEE International Symposium on Robot and Human interactive Communication*. IEEE. 2007, pp. 206–211.

[25] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. "A formal basis for the heuristic determination of minimum cost paths". In: *IEEE transactions on Systems Science and Cybernetics* 4.2 (1968), pp. 100–107.

[26] Dieter Hausamann et al. "Monitoring of gas pipelines–a civil UAV application". In: *Aircraft Engineering and Aerospace Technology* 77.5 (2005), pp. 352–360.

[27] Bryan W. Jones, Robert E. Marc, and Carl B. Watt. "Visual Prosthetics: Physiology, Bioengineering, Rehabilitation". In: ed. by Gislin Dagnelie. Boston, MA: Springer US, 2011. Chap. Retinal Remodeling and Visual Prosthetics, pp. 59–75. ISBN: 978-1-4419-0754-7. DOI: 10.1007/978-1-4419-0754-7_3. URL: http://dx.doi.org/10.1007/978-1-4419-0754-7_3.

[28] Hande Kaymaz Keskinpala and Julie A. Adams. "Objective data analysis for a PDA-based human robotic interface". In: *IEEE International Conference on Systems, Man and Cybernetics*. Vol. 3. IEEE. 2004, pp. 2809–2814.

[29] Oussama Khatib. "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots". In: *The International Journal of Robotics Research* 5.1 (1986), pp. 90–98.

[30] Nikolai Vladimirovich Kim and Mikhail Alekseevich Chervonenkis. “Situation control of unmanned aerial vehicles for road traffic monitoring”. In: *Modern Applied Science* 9.5 (2015), pp. 1–13.

[31] L. Kitagawa et al. “Semi-autonomous obstacle avoidance of omnidirectional wheelchair by joystick impedance control”. In: *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*. Vol. 4. IEEE. 2001, pp. 2148–2153.

[32] Yoram Koren and Johann Borenstein. “Potential Field Methods and Their Inherent Limitations for Mobile Robot Navigation”. In: *Proceedings IEEE International Conference on Robotics and Automation*. Apr. 1991, pp. 1398–1404.

[33] Manish Kumar, Kelly Cohen, and Baisravan HomChaudhuri. “Cooperative control of multiple uninhabited aerial vehicles for monitoring and fighting wildfires”. In: *Journal of Aerospace Computing, Information, and Communication* 8.1 (2011), pp. 1–16.

[34] Michael F. Land and David N. Lee. “Where we look when we steer”. In: *Nature* 369.6483 (1994), pp. 742–744.

[35] J. Corde Lane et al. “Effects of time delay on telerobotic control of neutral buoyancy vehicles”. In: *Proceedings. ICRA’02. IEEE International Conference on Robotics and Automation*. Vol. 3. IEEE. 2002, pp. 2874–2879.

[36] Maura C. Lohrenz et al. “Demonstration of a moving-map system for improved precise lane navigation of amphibious vehicles and landing craft”. In: *Proceedings OCEANS*. Vol. 3. IEEE. 2003, pp. 1247–1254.

[37] Jason P. Luck et al. “An investigation of real world control of robotic assets under communication latency”. In: *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*. ACM. 2006, pp. 202–209.

[38] I. Scott MacKenzie and Colin Ware. “Lag as a determinant of human performance in interactive systems”. In: *Proceedings of the INTERACT’93 and CHI’93 conference on Human factors in computing systems*. ACM. 1993, pp. 488–493.

[39] Michael J. Massimino and Thomas B. Sheridan. “Teleoperator performance with varying force and visual feedback”. In: *Human factors* 36.1 (1994), pp. 145–157.

[40] Douglas E. McGovern. *Experiences in teleoperation of land vehicles*. Tech. rep. Sandia National Labs., Albuquerque, NM (USA), 1989.

[41] Javier Minguez. "The Obstacle-Restriction Method for Robot Obstacle Avoidance in Difficult Environments". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2005, pp. 2284–2290.

[42] Javier Minguez, Florent Lamiraux, and Jean-Paul Laumond. "Springer Handbook of Robotics". In: *Springer handbook of robotics*. Ed. by Bruno Siciliano and Oussama Khatib. Springer Berlin Heidelberg, 2008. Chap. Motion Planning and Obstacle Avoidance, pp. 827–852. DOI: 10.1007/978-3-540-30301-5_36. URL: http://dx.doi.org/10.1007/978-3-540-30301-5_36.

[43] Javier Minguez and Luis Montano. "Extending Collision Avoidance Methods to Consider the Vehicle Shape, Kinematics, and Dynamics of a Mobile Robot". In: *IEEE Transactions on Robotics* 25.2 (2009), pp. 367–381.

[44] Javier Minguez and Luis Montano. "Nearness Diagram Navigation (ND): A New Real Time Collision Avoidance Approach". In: *IEEE/RSJ International Conference on Intelligent Robots and System*. 2000.

[45] Javier Minguez and Luis Montano. "Nearness Diagram (ND) Navigation: Collision Avoidance in Troublesome Scenarios". In: *IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION* 20.1 (Feb. 2004), pp. 45–59.

[46] Javier Minguez and Luis Montano. "The Ego-KinoDynamic Space: Collision Avoidance for any Shape Mobile Robots with Kinematic and Dynamic Constraints". In: *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*. Vol. 1. IEEE. 2003, pp. 637–643.

[47] Javier Minguez, Luis Montano, and Oussama Khatib. "Reactive Collision Avoidance for Navigation with Dynamic Constraints". In: *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*. Vol. 1. IEEE. 2002, pp. 588–594.

[48] Javier Minguez, Luis Montano, and José Santos-Victor. "Abstracting Vehicle Shape and Kinematic Constraints from Obstacle Avoidance Methods". In: *Autonomous Robots* 20.1 (2006), pp. 43–59.

[49] Javier Minguez, Luis Montano, and José Santos-Victor. "Reactive Navigation for Non-holonomic Robots using the Ego-Kinematic Space". In: *Proceedings IEEE International Conference on Robotics and Automation*. Vol. 3. IEEE. 2002, pp. 3074–3080.

[50] Hans P. Moravec. "Sensor fusion in certainty grids for mobile robots". In: *AI magazine* 9.2 (1988), pp. 61–74.

[51] Hans Moravec and Alberto Elfes. "High resolution maps from wide angle sonar". In: *Proceedings. 1985 IEEE International Conference on Robotics and Automation*. Vol. 2. 1985, pp. 116–121.

[52] Robin R. Murphy. "Human-robot interaction in rescue robotics". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 34.2 (2004), pp. 138–153.

[53] Francesco Nex and Fabio Remondino. "UAV for 3D mapping applications: a review". In: *Applied Geomatics* 6.1 (2014), pp. 1–15.

[54] Curtis W. Nielsen and Michael A. Goodrich. "Comparing the usefulness of video and map information in navigation tasks". In: *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*. ACM. 2006, pp. 95–101.

[55] *Parrot*. Accessed: 2017-05-10. URL: `https://www.parrot.com/`.

[56] Paul Richard et al. "Effect of frame rate and force feedback on virtual object manipulation". In: *Presence: Teleoperators & Virtual Environments* 5.1 (1996), pp. 95–108.

[57] *ROS*. Accessed: 2017-06-01. URL: `http://www.ros.org/`.

[58] Richard Schiffman. "Drones flying high as new tool for field biologists". In: *Science* 344.6183 (2014).

[59] David R. Scribner and James W. Gombash. *The effect of stereoscopic and wide field of view conditions on teleoperator performance*. Tech. rep. DTIC Document, 1998.

[60] Thomas B. Sheridan and William R. Ferrell. "Remote manipulative control with transmission delay". In: *IEEE Transactions on Human Factors in Electronics* 1 (1963), pp. 25–29.

[61] Christopher C. Smyth. "Indirect vision driving with fixed flat panel displays for near unity, wide, and extended fields of camera view". In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. Vol. 44. 36. SAGE Publications. 2000, pp. 541–544.

[62] Jennifer E. Thropp and Jessie Y. Chen. *The effects of slow frame rates on human performance*. Tech. rep. DTIC Document, 2006.

[63] Iwan Ulrich and Johann Borenstein. "VFH+: Reliable obstacle avoidance for fast mobile robots". In: *Proceedings. 1998 IEEE International Conference on Robotics and Automation*. Vol. 2. IEEE. 1998, pp. 1572–1577.

[64] Iwan Ulrich and Johann Borenstein. "VFH/sup*: local obstacle avoidance with look-ahead verification". In: *Proceedings. ICRA'00. IEEE International Conference on Robotics and Automation*. Vol. 3. IEEE. 2000, pp. 2505–2511.

[65] Jan BF Van Erp and Pieter Padmos. "Image parameters for driving with indirect viewing systems". In: *Ergonomics* 46.15 (2003), pp. 1471–1499.

[66] D. Vikerimark and J. Minguez. "Reactive obstacle avoidance for mobile robots that operate in confined 3D workspaces". In: *Electrotechnical Conference, 2006. MELECON 2006. IEEE Mediterranean*. IEEE. 2006, pp. 1246–1251.

[67] Sonia Waharte and Niki Trigoni. "Supporting search and rescue operations with UAVs". In: *International Conference on Emerging Security Technologies (EST)*. IEEE. 2010, pp. 142–147.

[68] Benjamin Watson et al. "Effects of variation in system responsiveness on user performance in virtual environments". In: *Human Factors: The Journal of the Human Factors and Ergonomics Society* 40.3 (1998).

[69] Benjamin Watson et al. "Evaluation of the effects of frame time variation on VR task performance". In: *Virtual Reality Annual International Symposium*. IEEE. 1997, pp. 38–44.

[70] Bob G. Witmer and Wallace J. Sadowski Jr. "Nonvisually guided locomotion to a previously viewed target in real and virtual environments". In: *Human factors* 40.3 (Sept. 1998), pp. 478–488.

[71] David D. Woods et al. “Envisioning human-robot coordination in future operations”. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 34.2 (May 2004), pp. 210–218.

[72] Chunhua Zhang and John M. Kovacs. “The application of small unmanned aerial systems for precision agriculture: a review”. In: *Precision agriculture* 13.6 (2012), pp. 693–712.

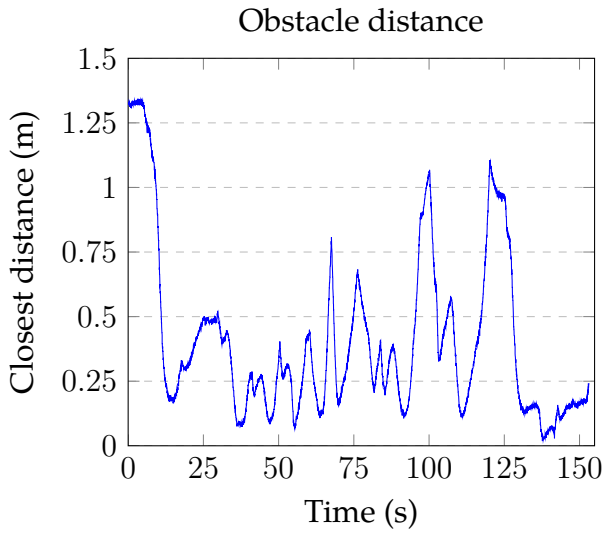
Appendix A

Additional User Study Results

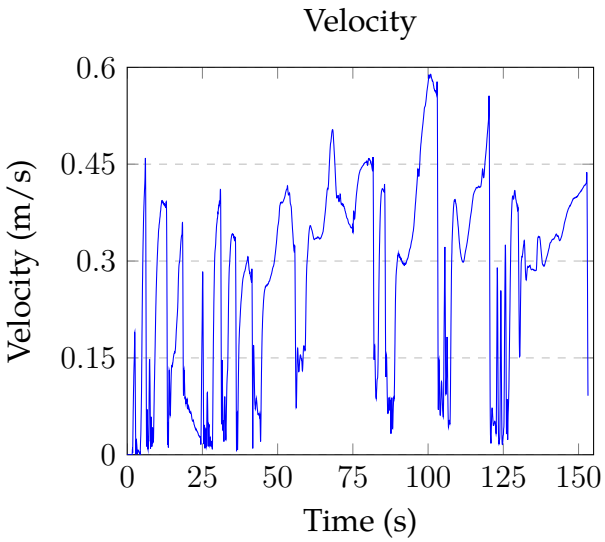
A.1 Operator 1

Configuration 1

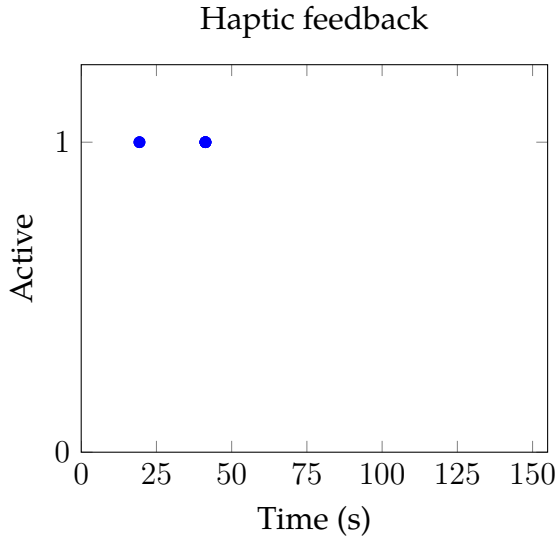
Figure A.1: The path for operator 1 with configuration 1 in the user study. Red is obstacles detected by sensor, blue is the UAV, and green is when haptic feedback is activated.



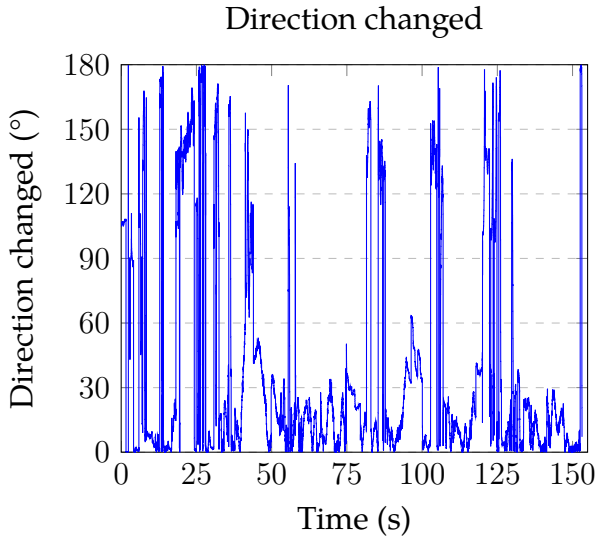
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.

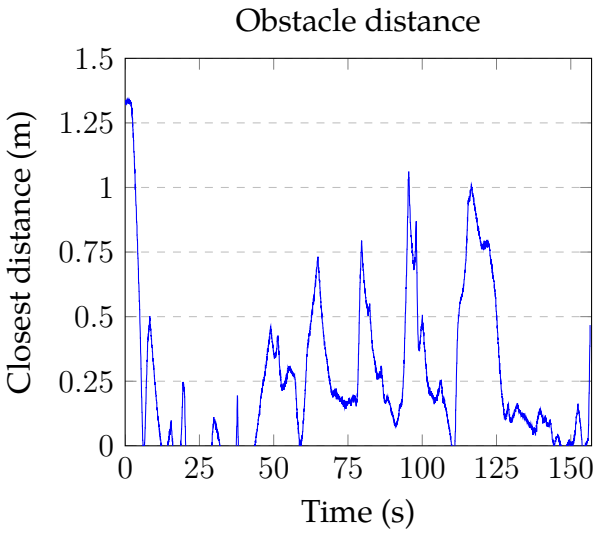


(d) Direction changed over time.

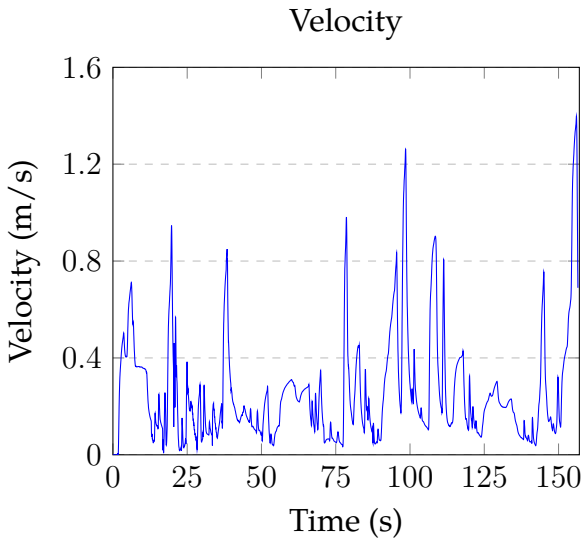
Figure A.2: Results for operator 1 from the user study with configuration 1.

Configuration 2

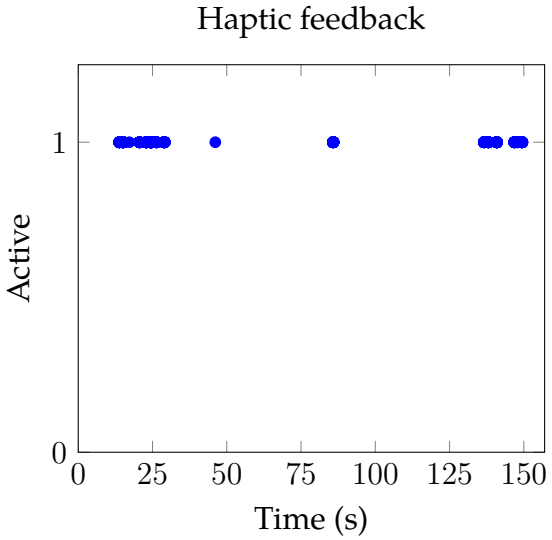
Figure A.3: The path for operator 1 with configuration 2 in the user study. Red is obstacles detected by sensor, blue is the UAV, green is when haptic feedback is activated, and pink indicates collision.



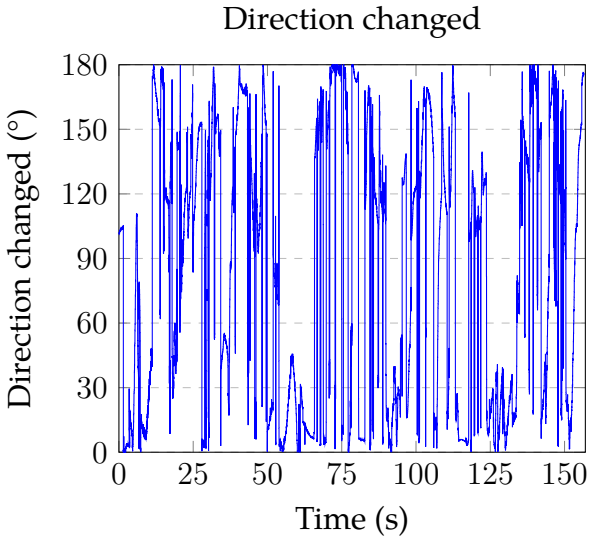
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.



(d) Direction changed over time.

Figure A.4: Results for operator 1 from the user study with configuration 2.

Configuration 3

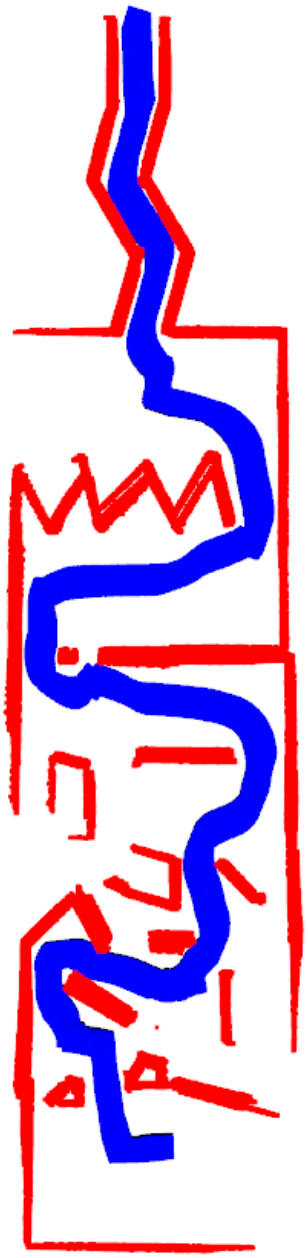
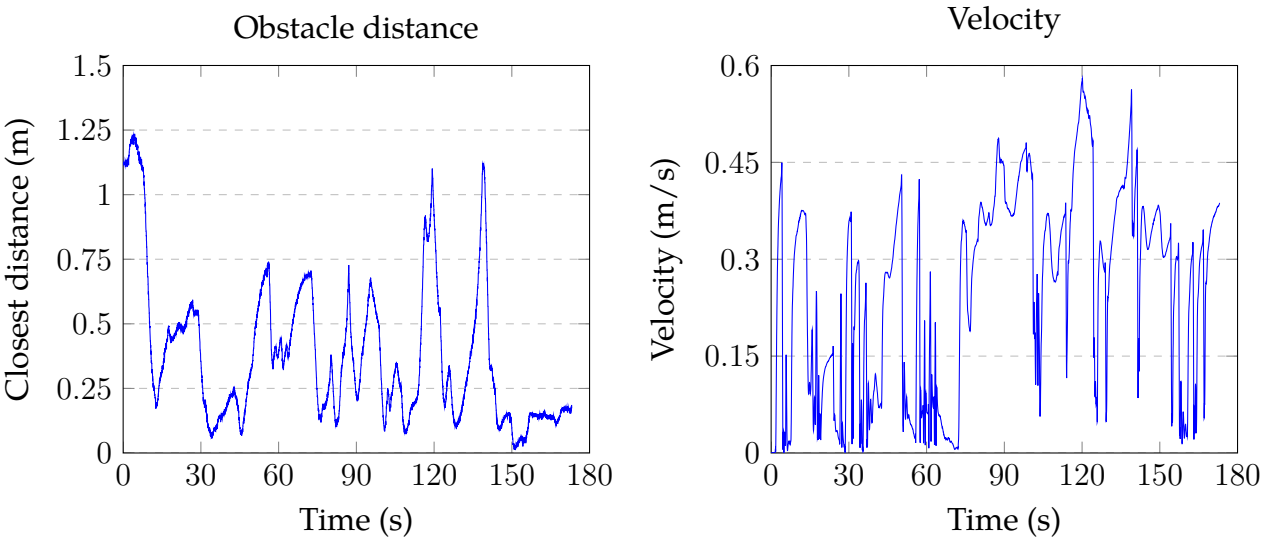


Figure A.5: The path for operator 1 with configuration 3 in the user study. Red is obstacles detected by sensor and blue is the UAV.



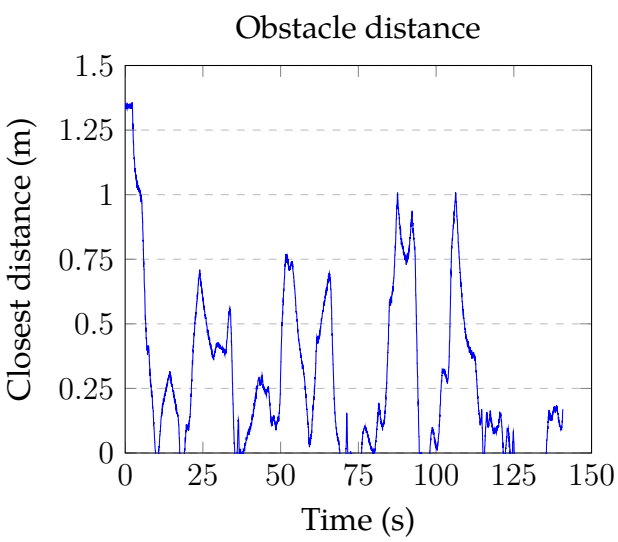
(a) The closest obstacle distance from sensor measurements over time.

(b) The velocity over time.

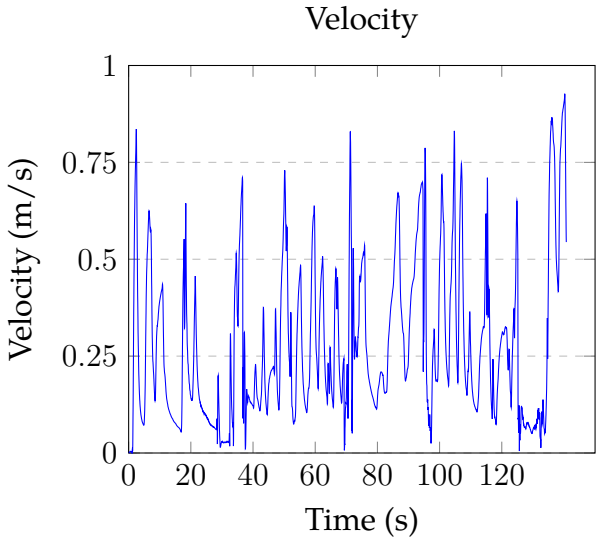
Figure A.6: Results for operator 1 from the user study with configuration 3.

Configuration 4

Figure A.7: The path for operator 1 with configuration 4 in the user study. Red is obstacles detected by sensor, blue is the UAV, and pink indicates collision.



(a) The closest obstacle distance from sensor measurements over time.

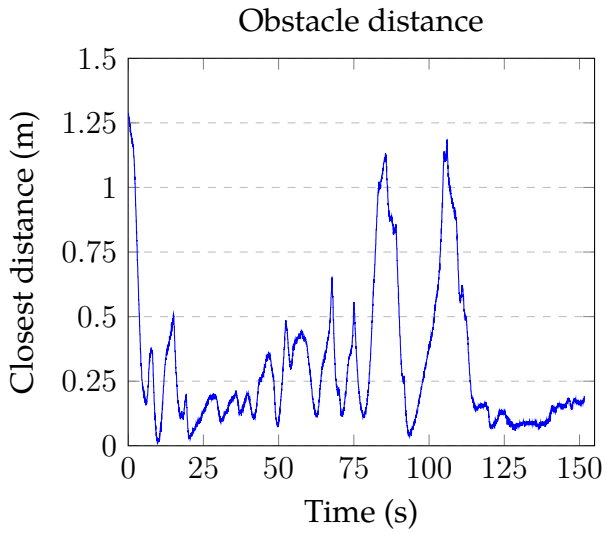


(b) The velocity over time.

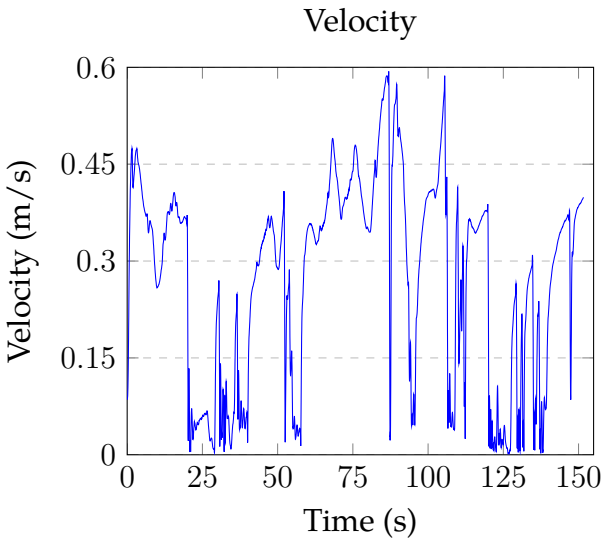
Figure A.8: Results for operator 1 from the user study with configuration 4.

Configuration 5

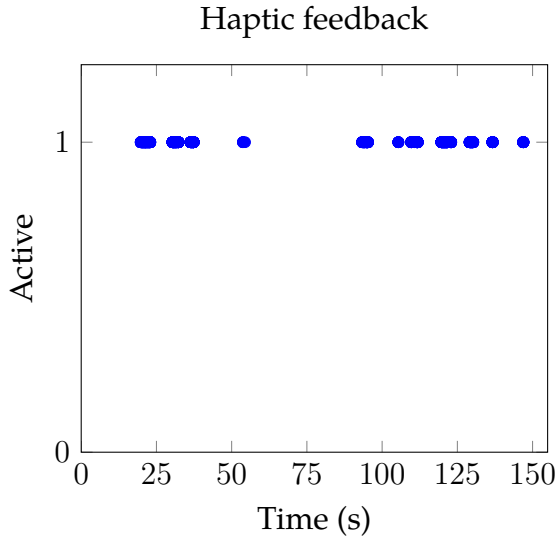
Figure A.9: The path for operator 1 with configuration 5 in the user study. Red is obstacles detected by sensor, blue is the UAV, and green is when haptic feedback is activated.



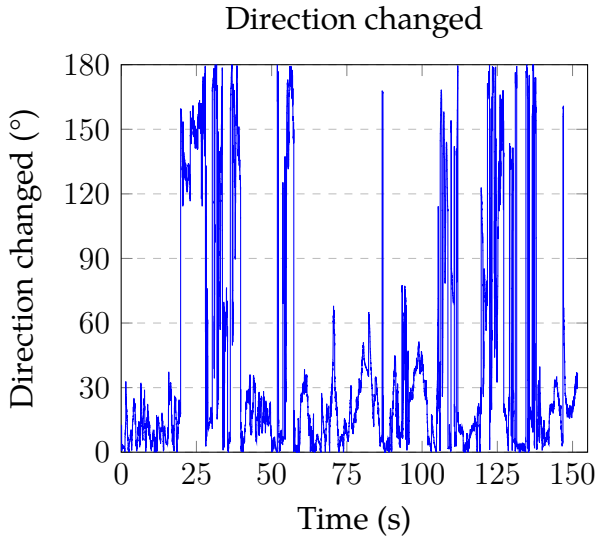
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.

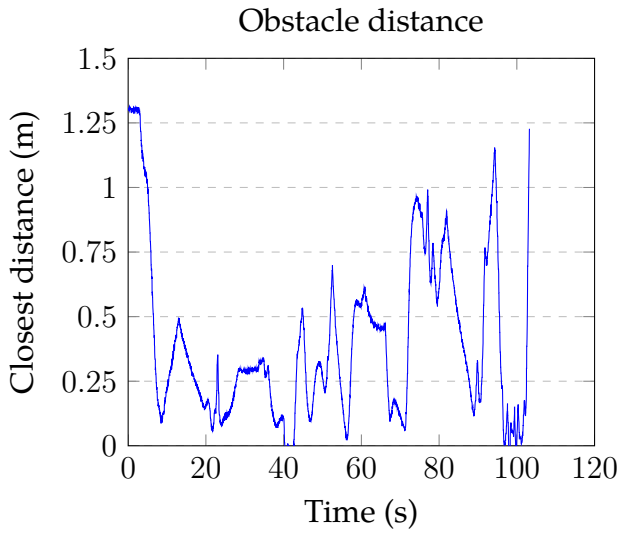


(d) Direction changed over time.

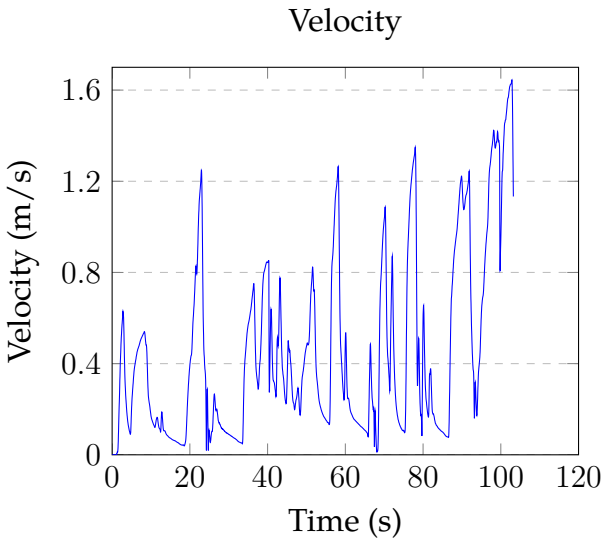
Figure A.10: Results for operator 1 from the user study with configuration 5.

Configuration 6

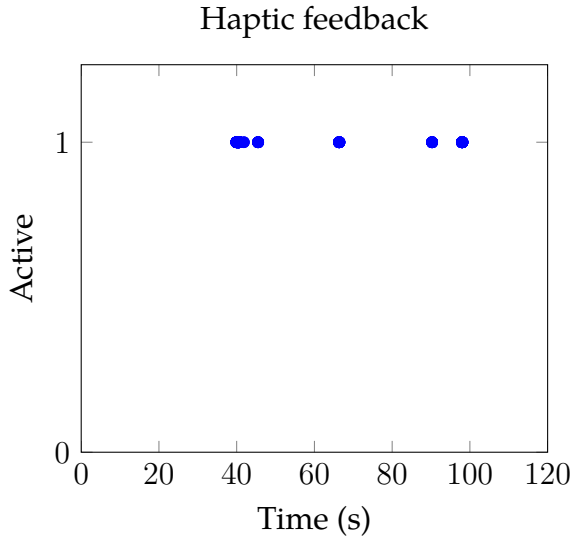
Figure A.11: The path for operator 1 with configuration 6 in the user study. Red is obstacles detected by sensor, blue is the UAV, green is when haptic feedback is activated, and pink indicates collision.



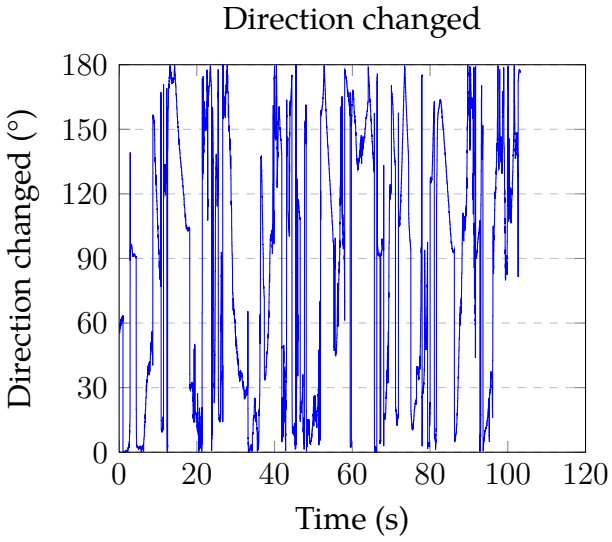
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.



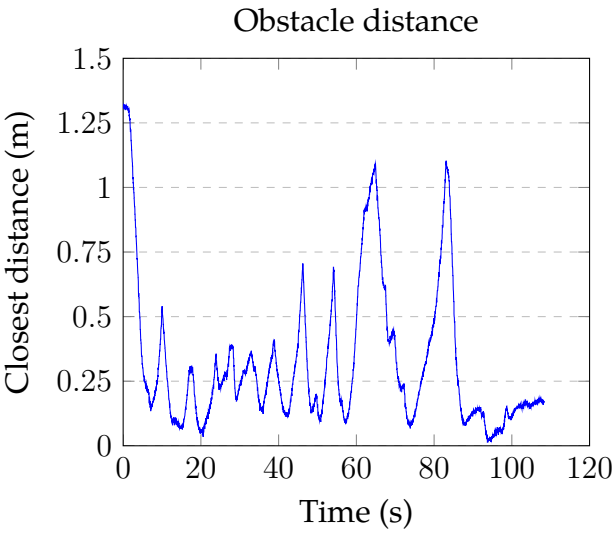
(d) Direction changed over time.

Figure A.12: Results for operator 1 from the user study with configuration 6.

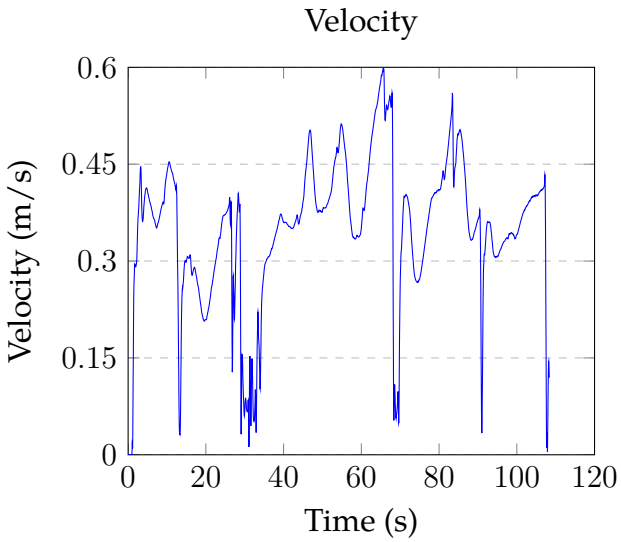
A.2 Operator 2

Configuration 1

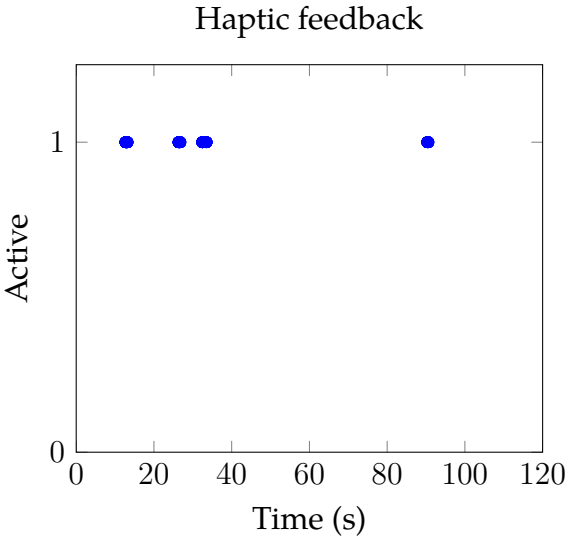
Figure A.13: The path for operator 2 with configuration 1 in the user study. Red is obstacles detected by sensor, blue is the UAV, and green is when haptic feedback is activated.



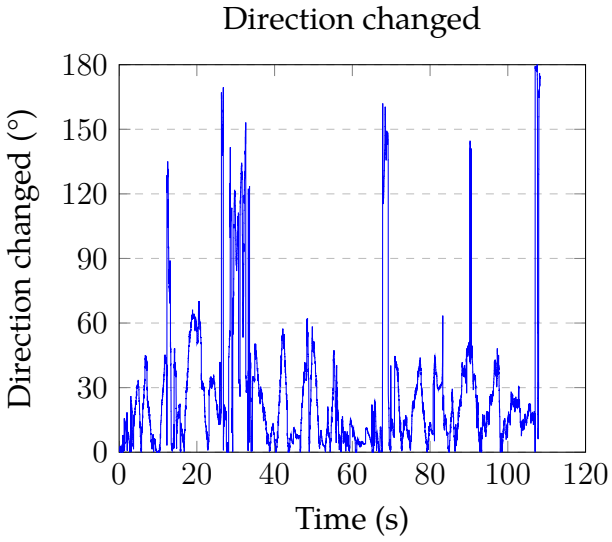
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.



(d) Direction changed over time.

Figure A.14: Results for operator 2 from the user study with configuration 1.

Configuration 2

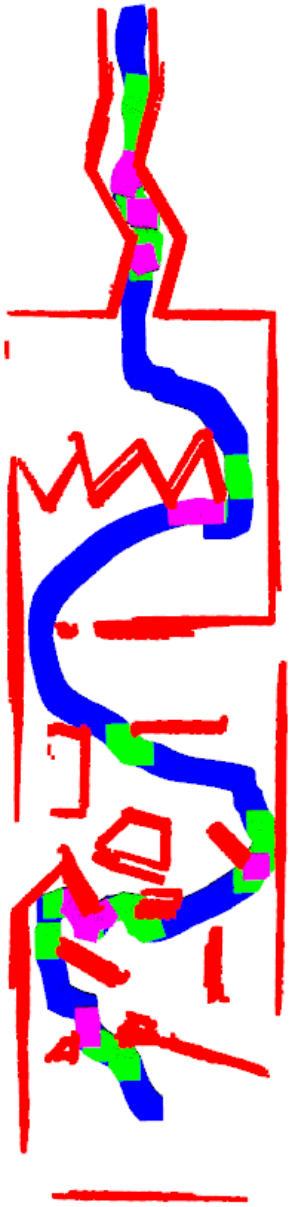
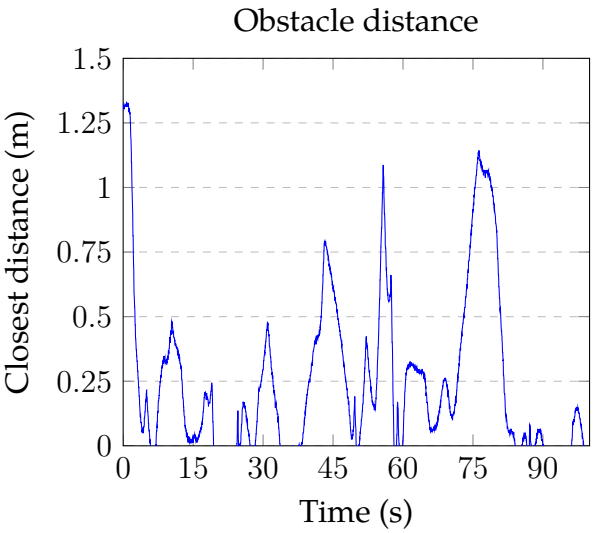
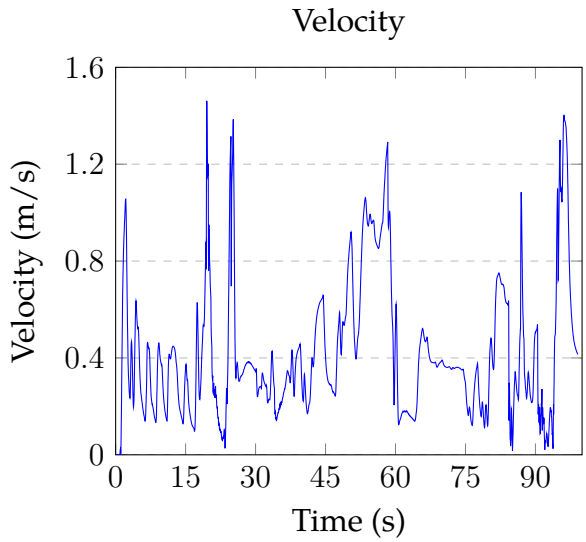


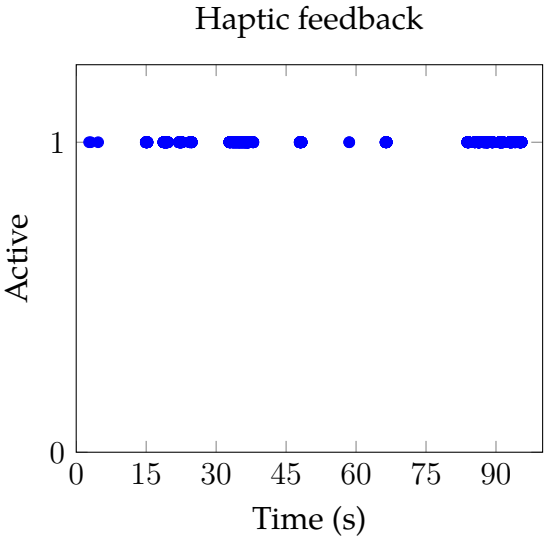
Figure A.15: The path for operator 2 with configuration 2 in the user study. Red is obstacles detected by sensor, blue is the UAV, green is when haptic feedback is activated, and pink indicates collision.



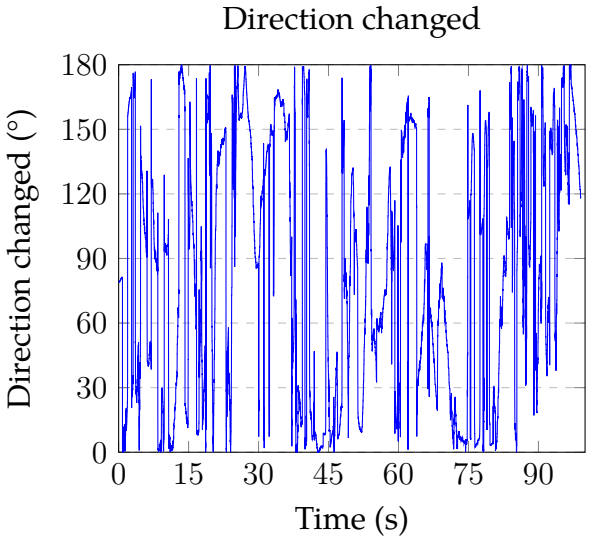
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.

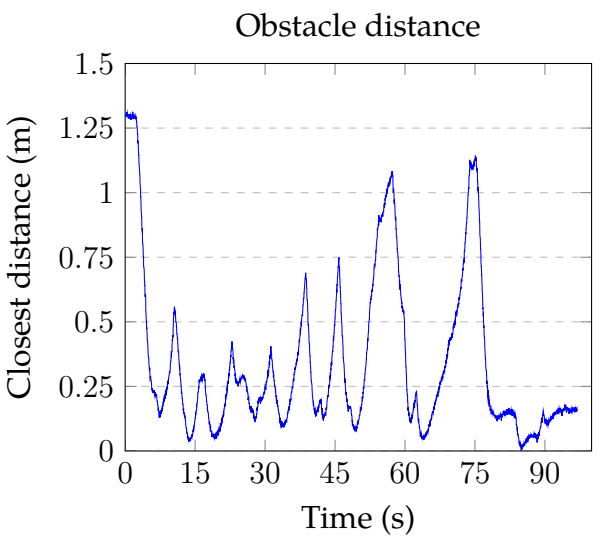


(d) Direction changed over time.

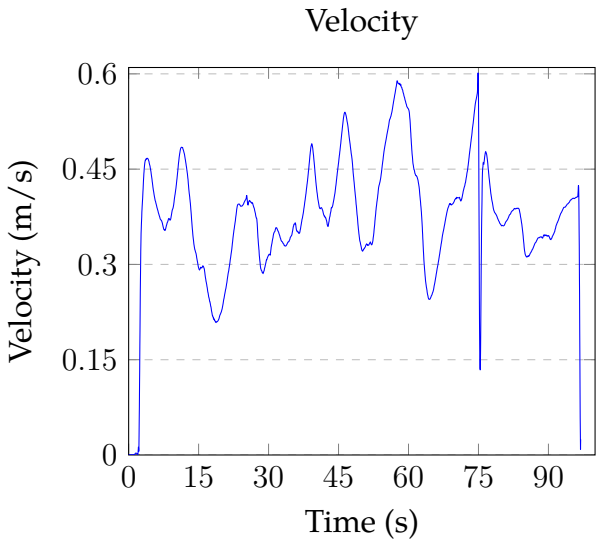
Figure A.16: Results for operator 2 from the user study with configuration 2.

Configuration 3

Figure A.17: The path for operator 2 with configuration 3 in the user study. Red is obstacles detected by sensor and blue is the UAV.



(a) The closest obstacle distance from sensor measurements over time.

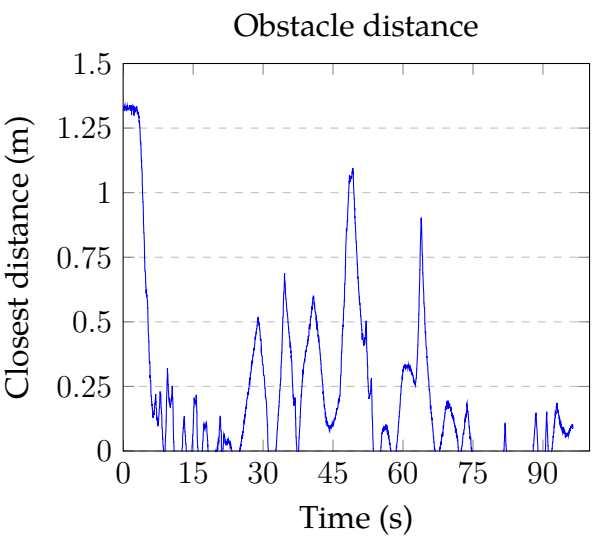


(b) The velocity over time.

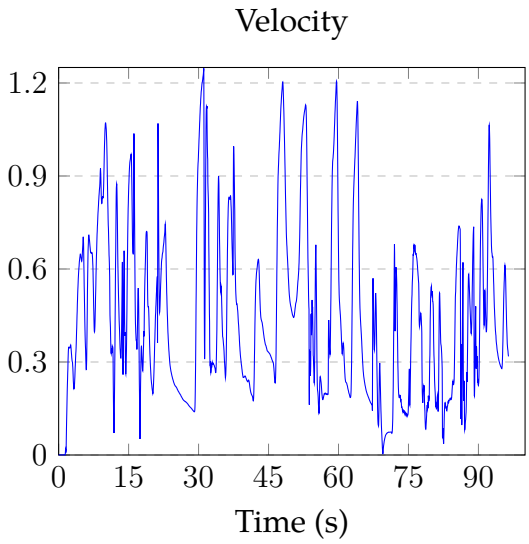
Figure A.18: Results for operator 2 from the user study with configuration 3.

Configuration 4

Figure A.19: The path for operator 2 with configuration 4 in the user study. Red is obstacles detected by sensor, blue is the UAV, and pink indicates collision.



(a) The closest obstacle distance from sensor measurements over time.

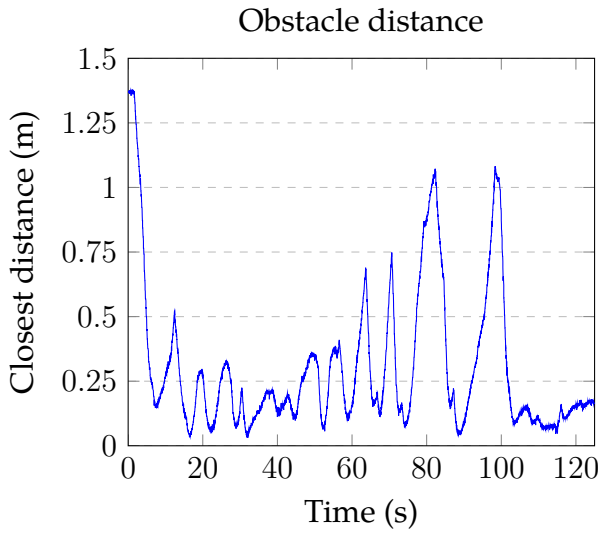


(b) The velocity over time.

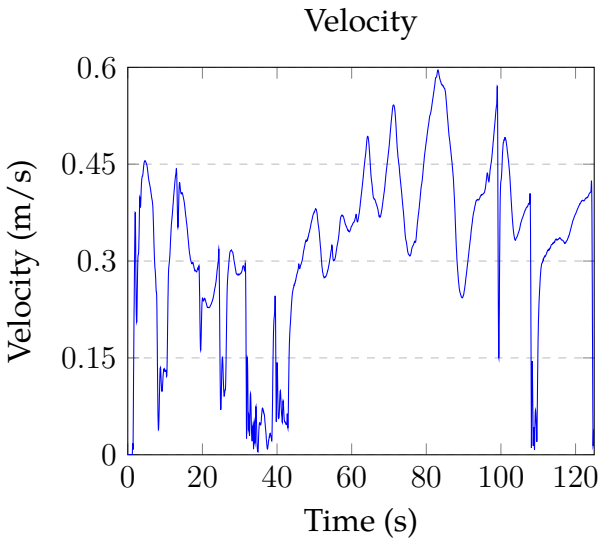
Figure A.20: Results for operator 2 from the user study with configuration 4.

Configuration 5

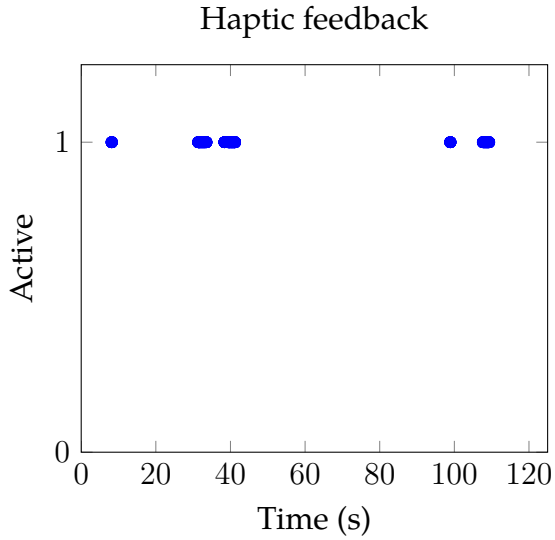
Figure A.21: The path for operator 2 with configuration 5 in the user study. Red is obstacles detected by sensor, blue is the UAV, and green is when haptic feedback is activated.



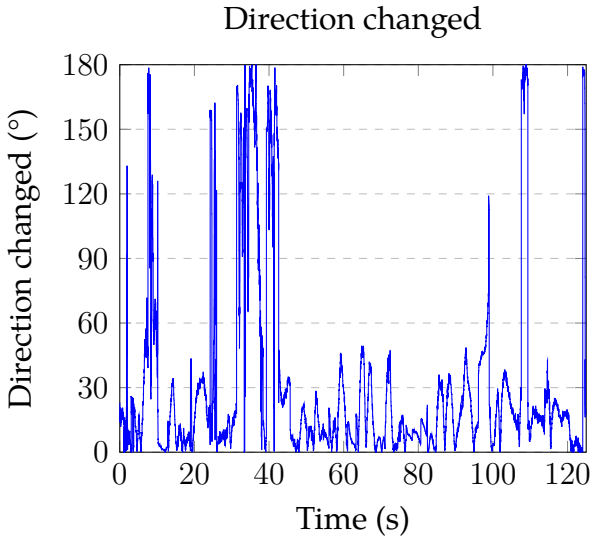
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.

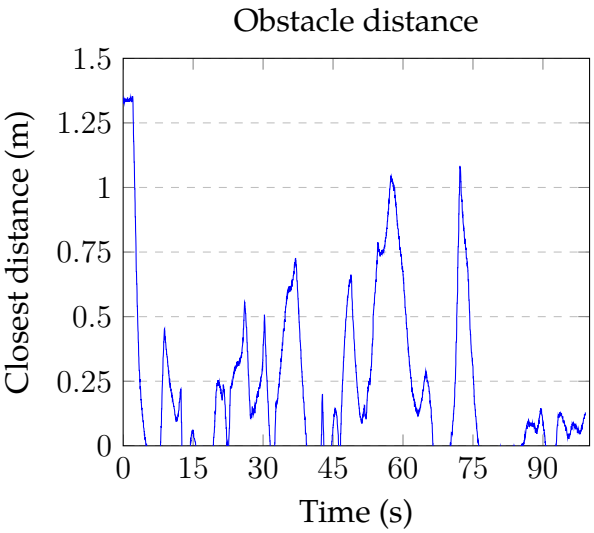


(d) Direction changed over time.

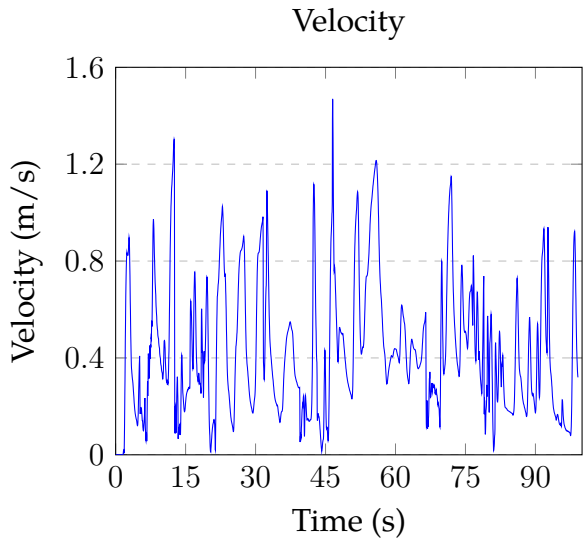
Figure A.22: Results for operator 2 from the user study with configuration 5.

Configuration 6

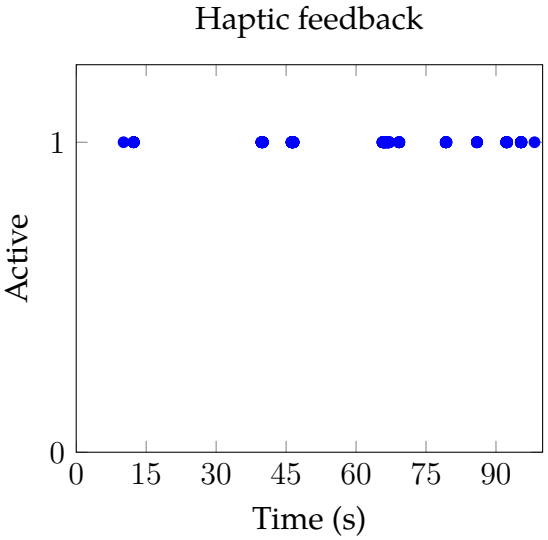
Figure A.23: The path for operator 2 with configuration 6 in the user study. Red is obstacles detected by sensor, blue is the UAV, green is when haptic feedback is activated, and pink indicates collision.



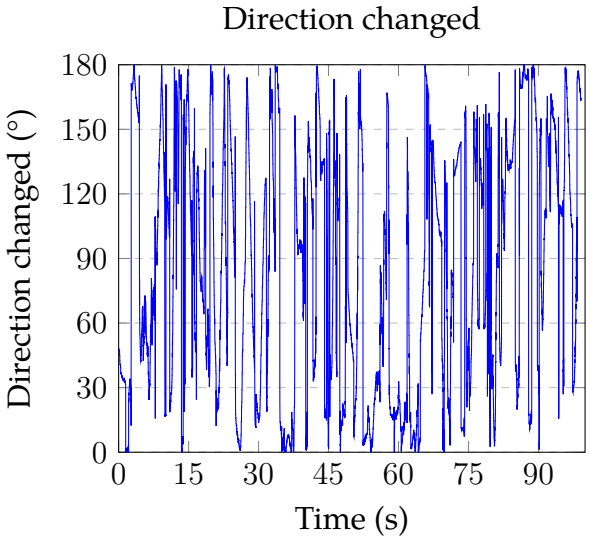
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.



(d) Direction changed over time.

Figure A.24: Results for operator 2 from the user study with configuration 6.

A.3 Operator 3

Configuration 1

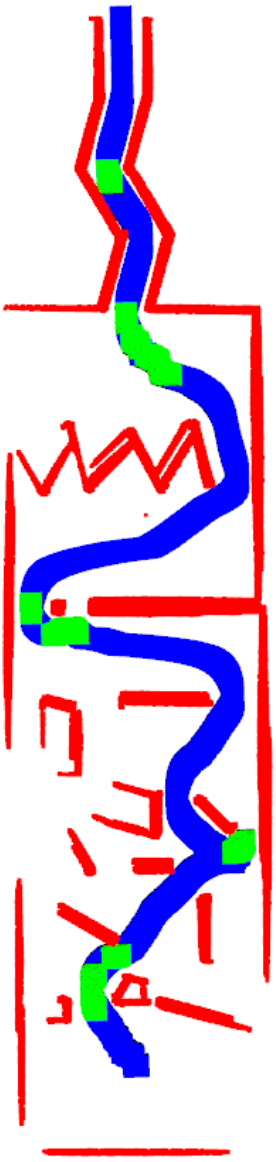
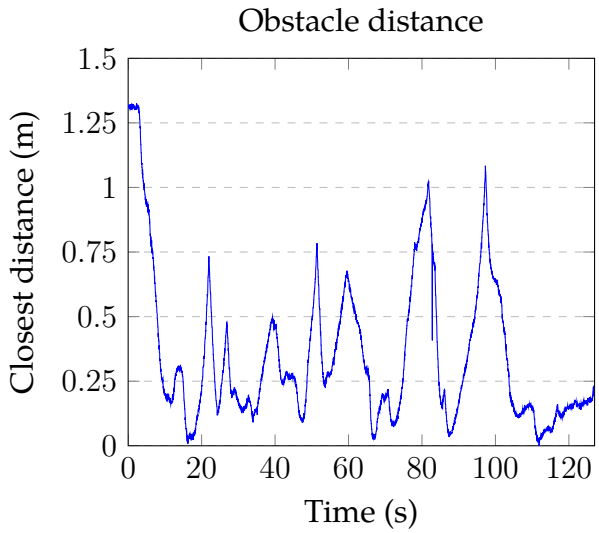
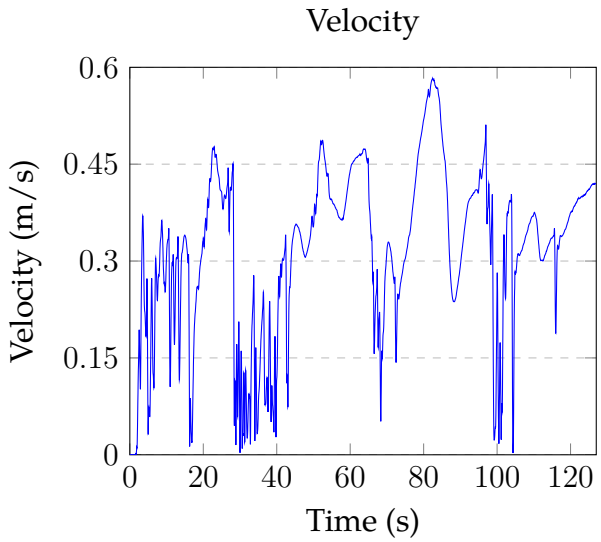


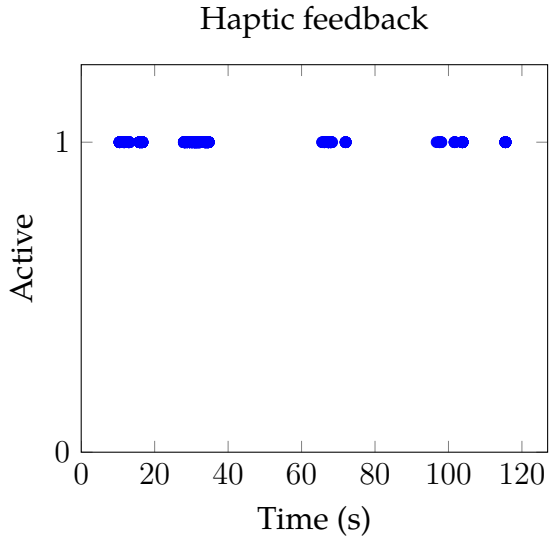
Figure A.25: The path for operator 3 with configuration 1 in the user study. Red is obstacles detected by sensor, blue is the UAV, and green is when haptic feedback is activated.



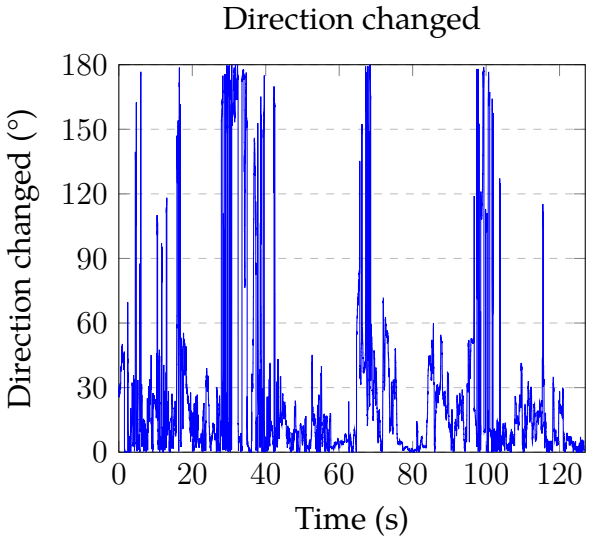
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.

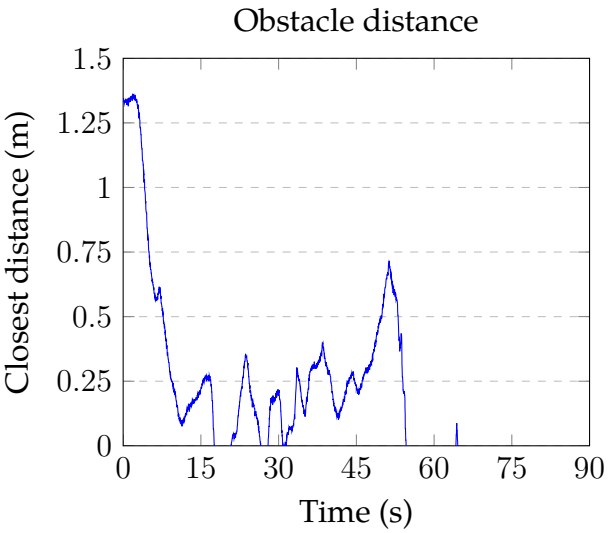


(d) Direction changed over time.

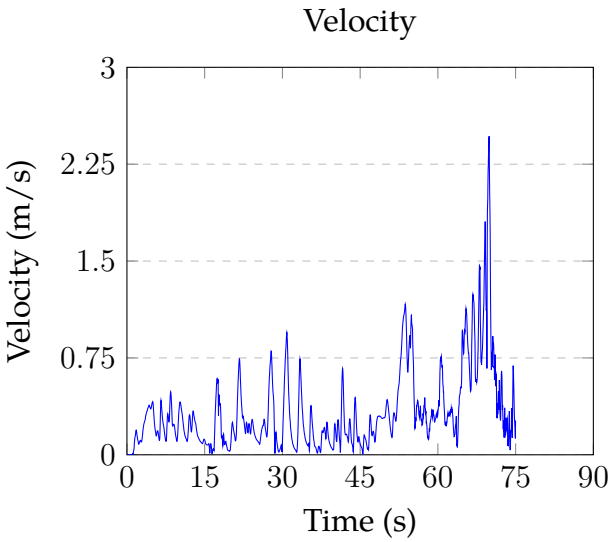
Figure A.26: Results for operator 3 from the user study with configuration 1.

Configuration 2

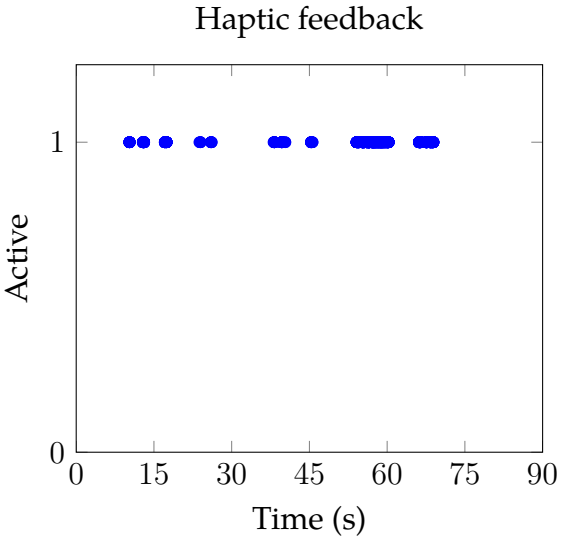
Figure A.27: The path for operator 3 with configuration 2 in the user study. Red is obstacles detected by sensor, blue is the UAV, green is when haptic feedback is activated, and pink indicates collision.



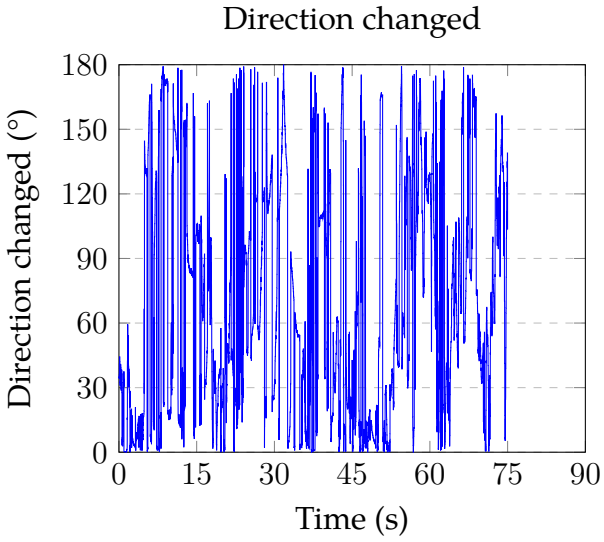
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.

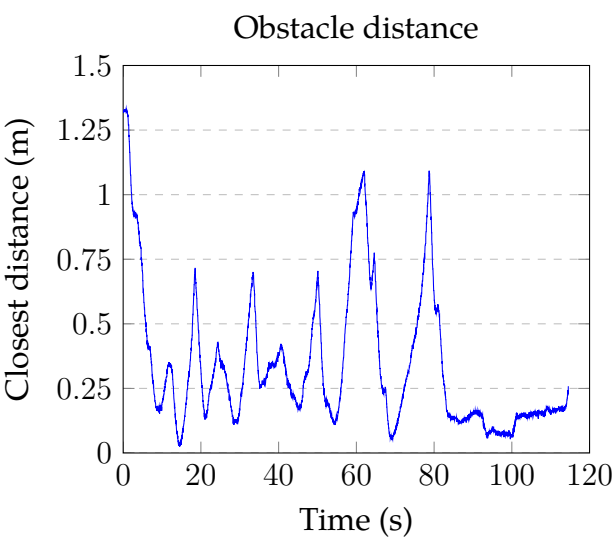


(d) Direction changed over time.

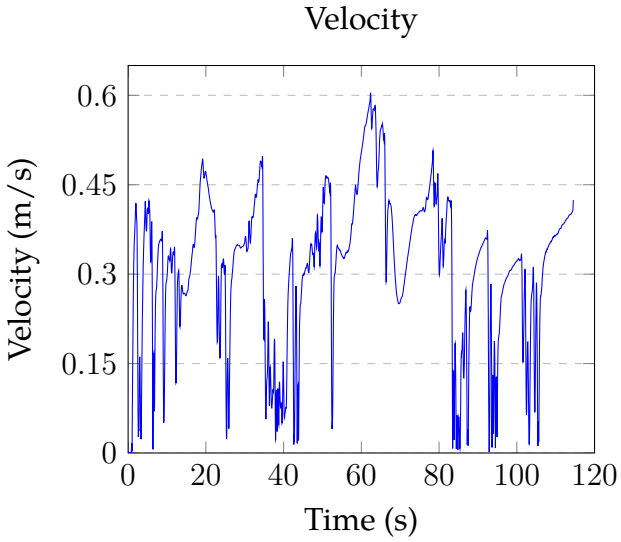
Figure A.28: Results for operator 3 from the user study with configuration 2.

Configuration 3

Figure A.29: The path for operator 3 with configuration 3 in the user study. Red is obstacles detected by sensor and blue is the UAV.



(a) The closest obstacle distance from sensor measurements over time.

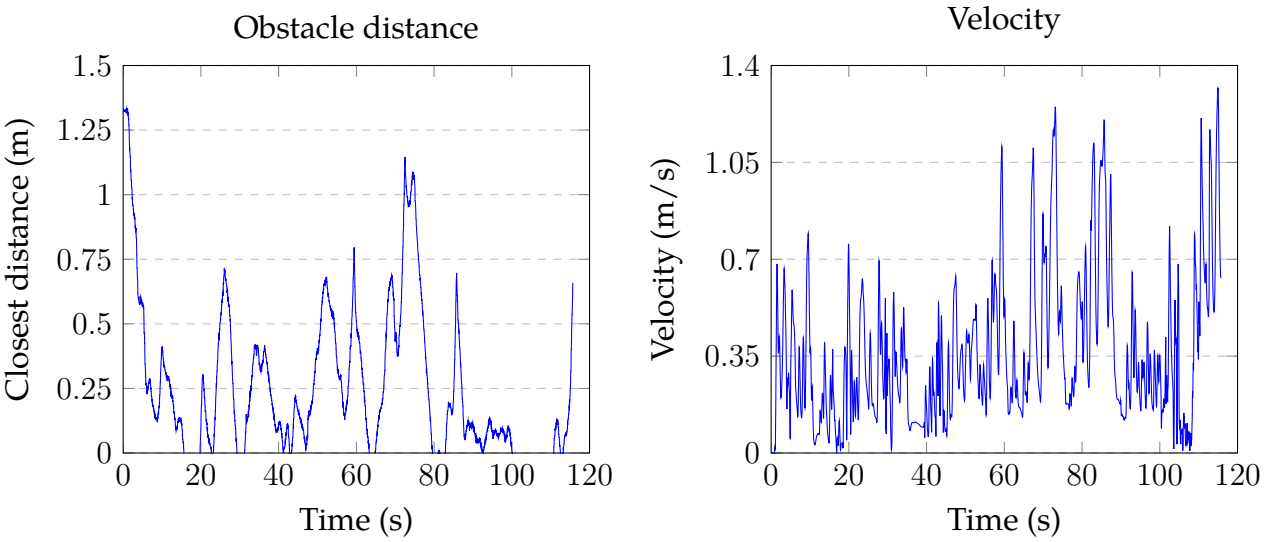


(b) The velocity over time.

Figure A.30: Results for operator 3 from the user study with configuration 3.

Configuration 4

Figure A.31: The path for operator 3 with configuration 4 in the user study. Red is obstacles detected by sensor, blue is the UAV, and pink indicates collision.



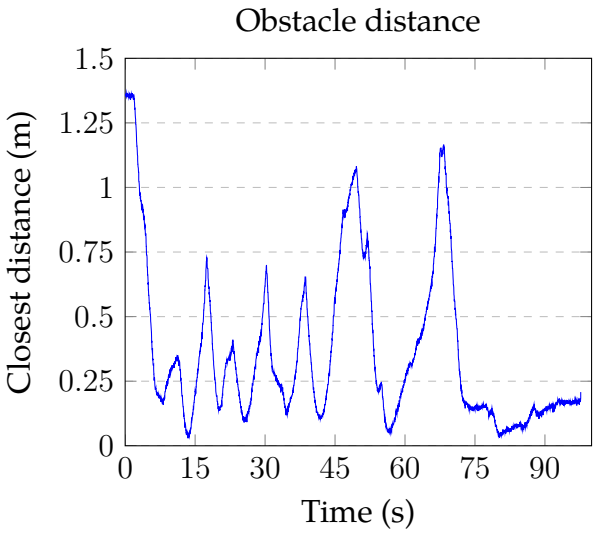
(a) The closest obstacle distance from sensor measurements over time.

(b) The velocity over time.

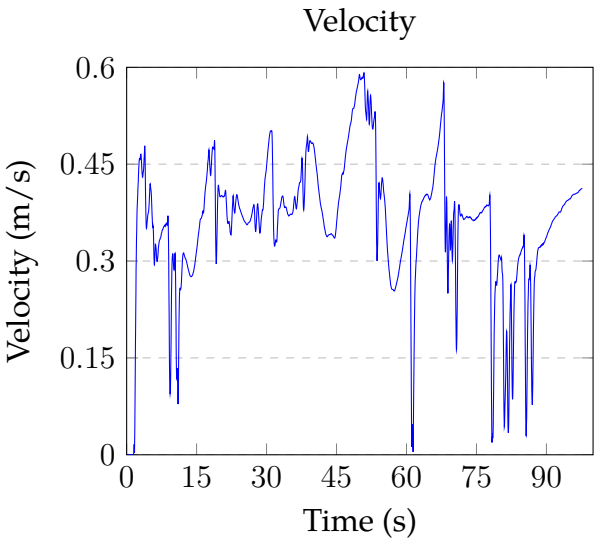
Figure A.32: Results for operator 3 from the user study with configuration 4.

Configuration 5

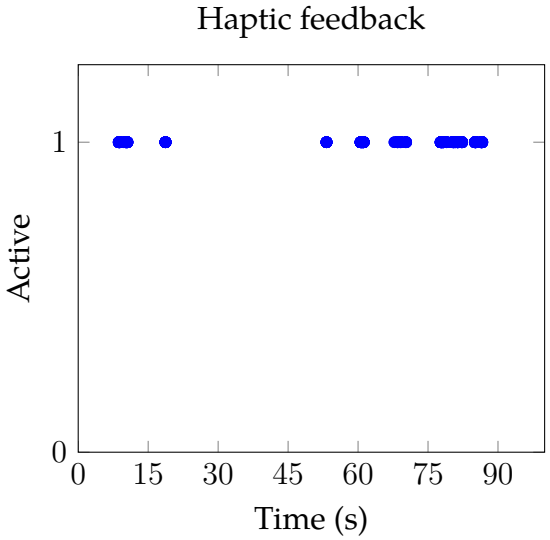
Figure A.33: The path for operator 3 with configuration 5 in the user study. Red is obstacles detected by sensor, blue is the UAV, and green is when haptic feedback is activated.



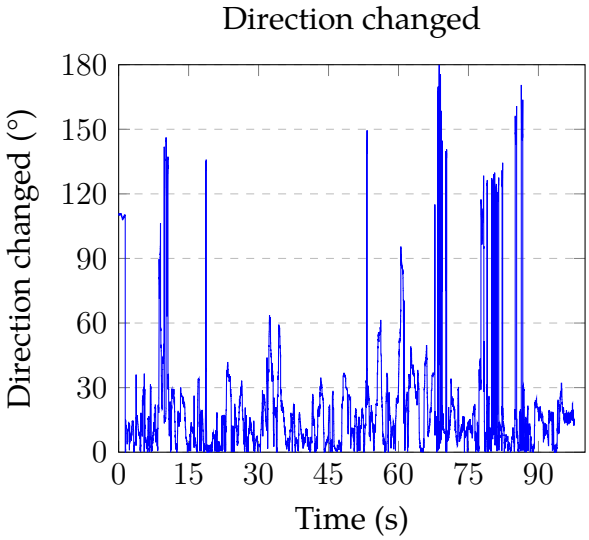
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.

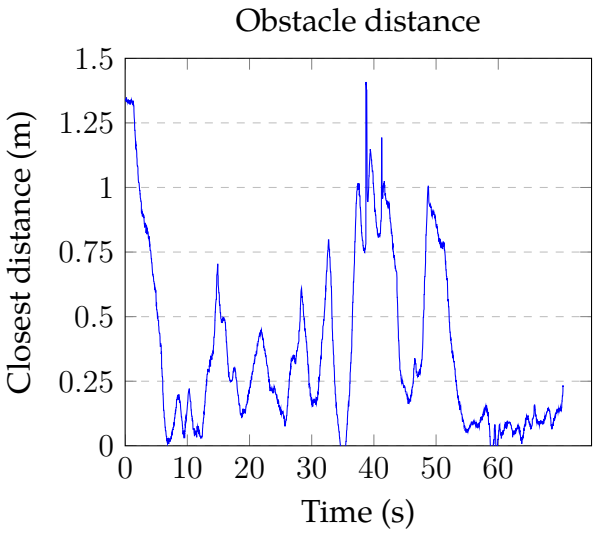


(d) Direction changed over time.

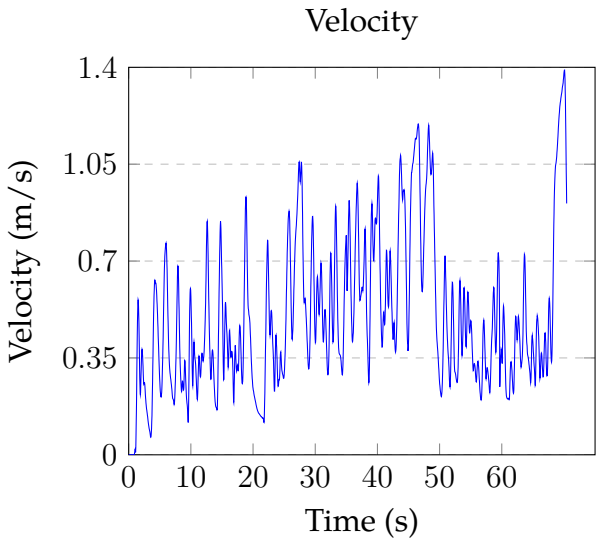
Figure A.34: Results for operator 3 from the user study with configuration 5.

Configuration 6

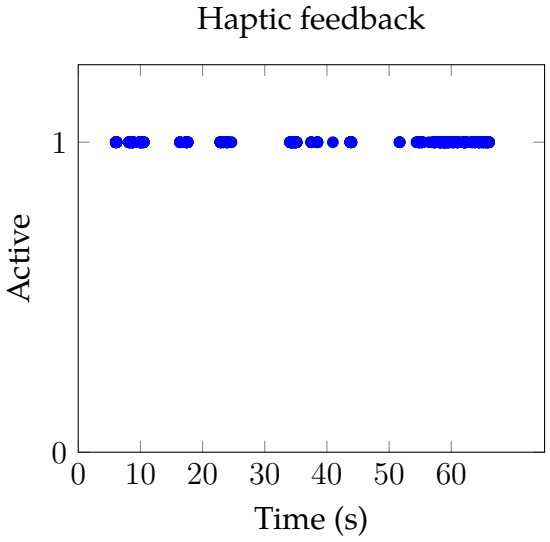
Figure A.35: The path for operator 3 with configuration 6 in the user study. Red is obstacles detected by sensor, blue is the UAV, green is when haptic feedback is activated, and pink indicates collision.



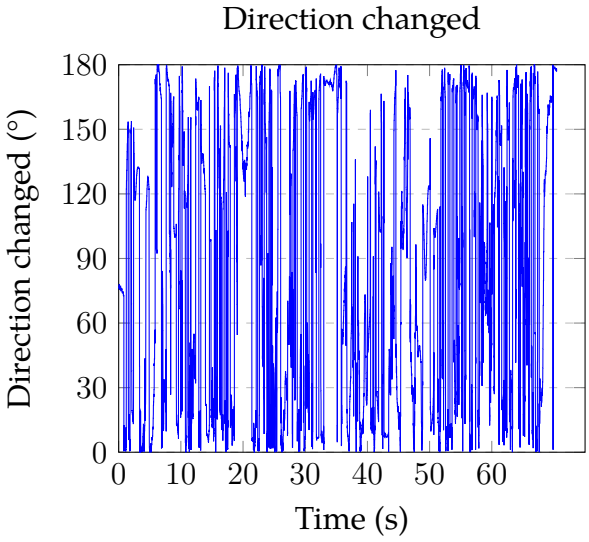
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.



(d) Direction changed over time.

Figure A.36: Results for operator 3 from the user study with configuration 6.

A.4 Operator 4

Configuration 1

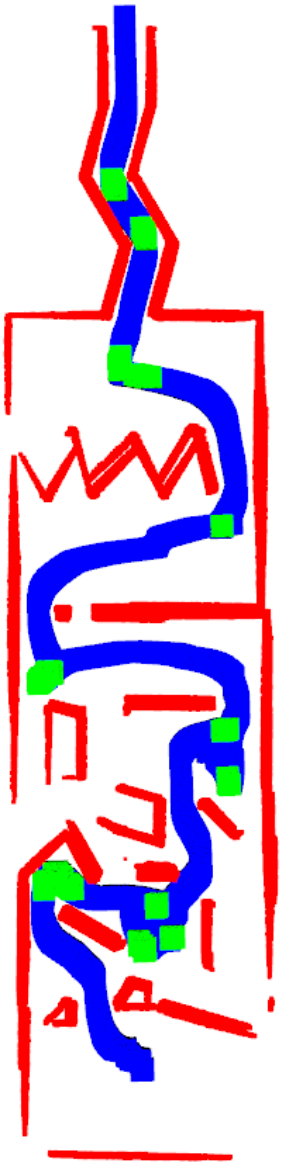
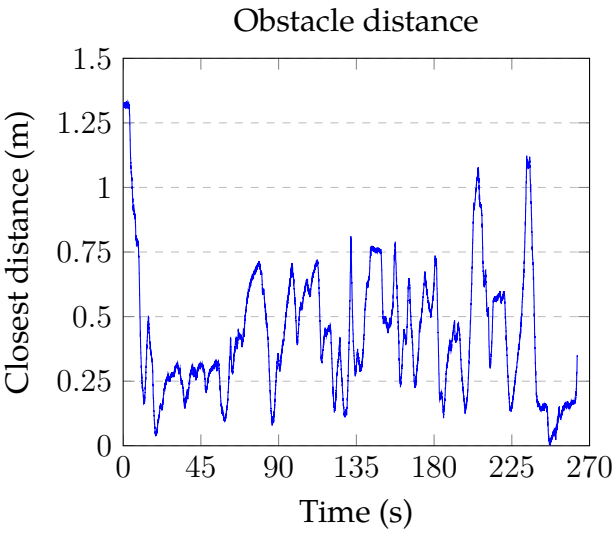
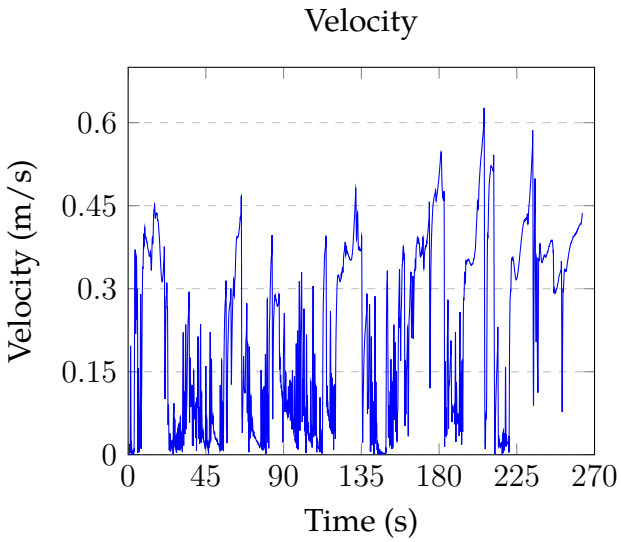


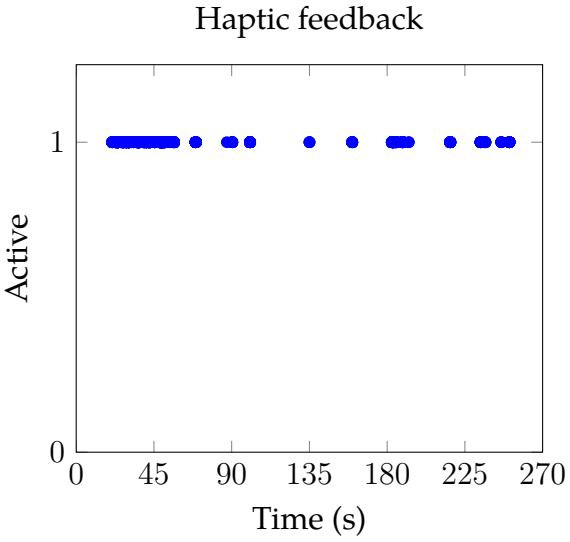
Figure A.37: The path for operator 4 with configuration 1 in the user study. Red is obstacles detected by sensor, blue is the UAV, and green is when haptic feedback is activated.



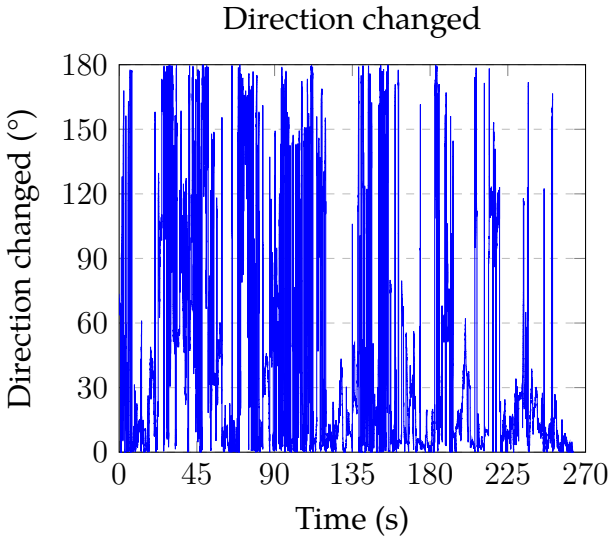
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.

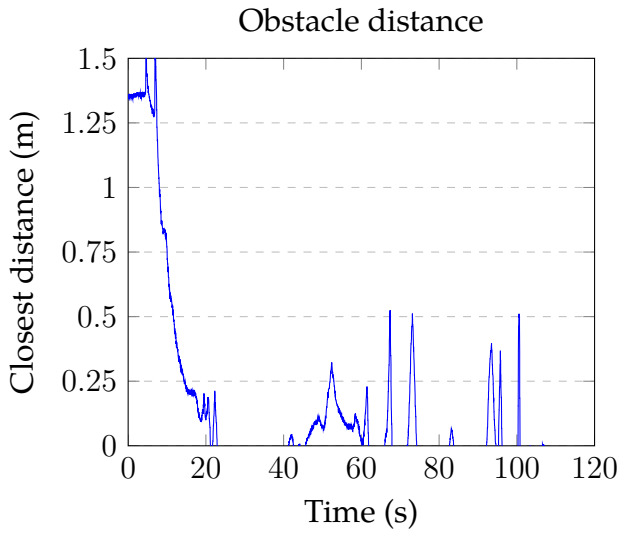


(d) Direction changed over time.

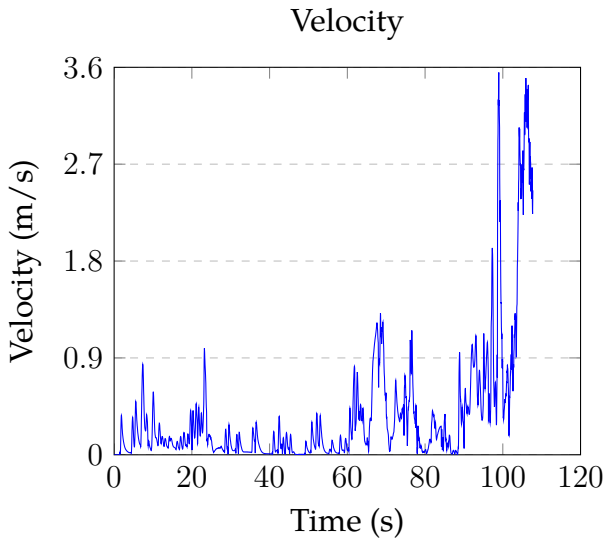
Figure A.38: Results for operator 4 from the user study with configuration 1.

Configuration 2

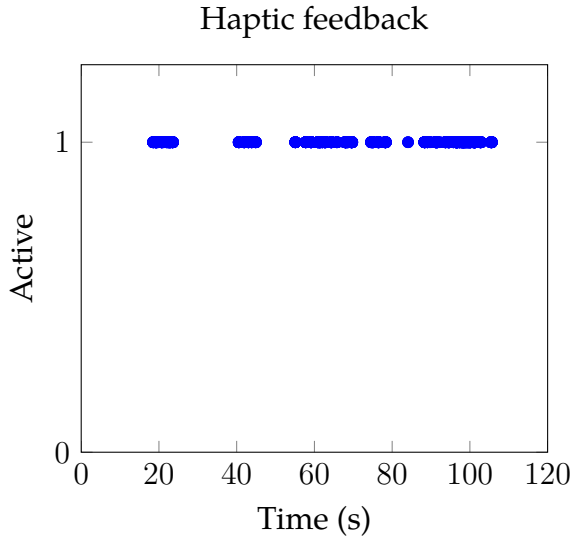
Figure A.39: The path for operator 4 with configuration 2 in the user study. Red is obstacles detected by sensor, blue is the UAV, green is when haptic feedback is activated, and pink indicates collision.



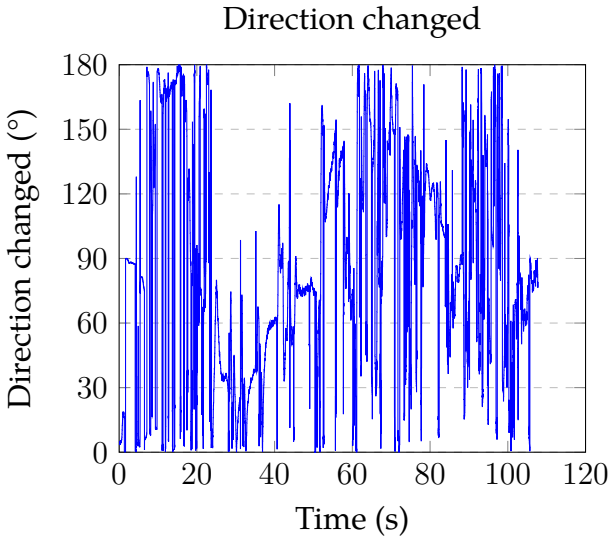
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.



(d) Direction changed over time.

Figure A.40: Results for operator 4 from the user study with configuration 2.

Configuration 3

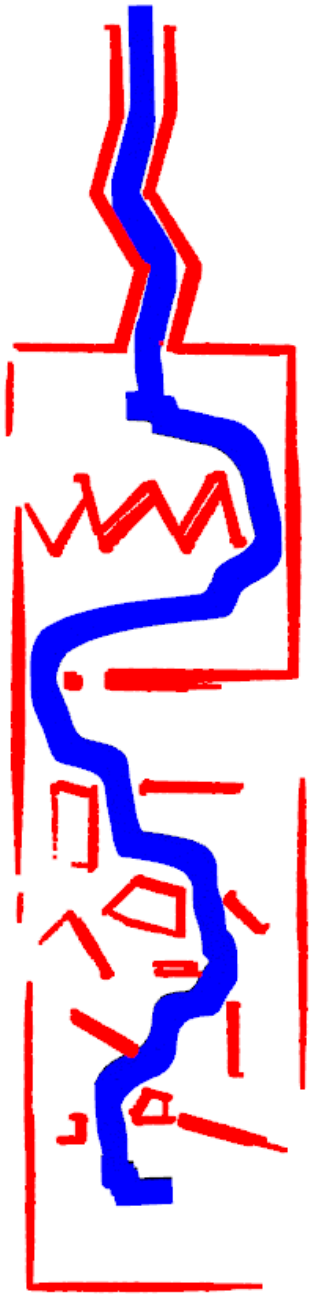
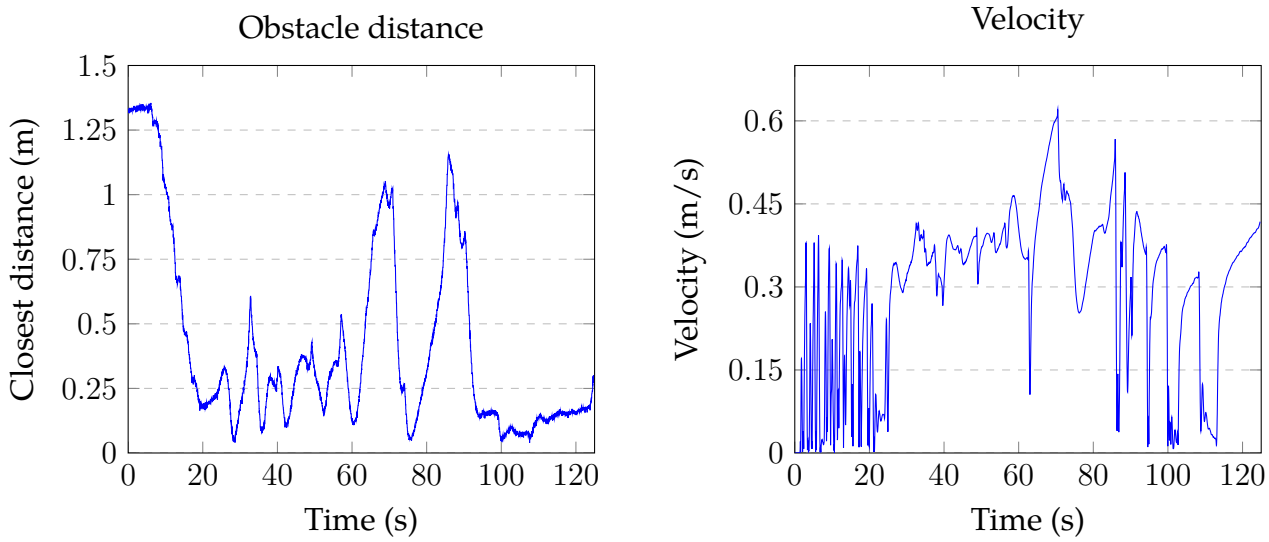


Figure A.41: The path for operator 4 with configuration 3 in the user study. Red is obstacles detected by sensor and blue is the UAV.



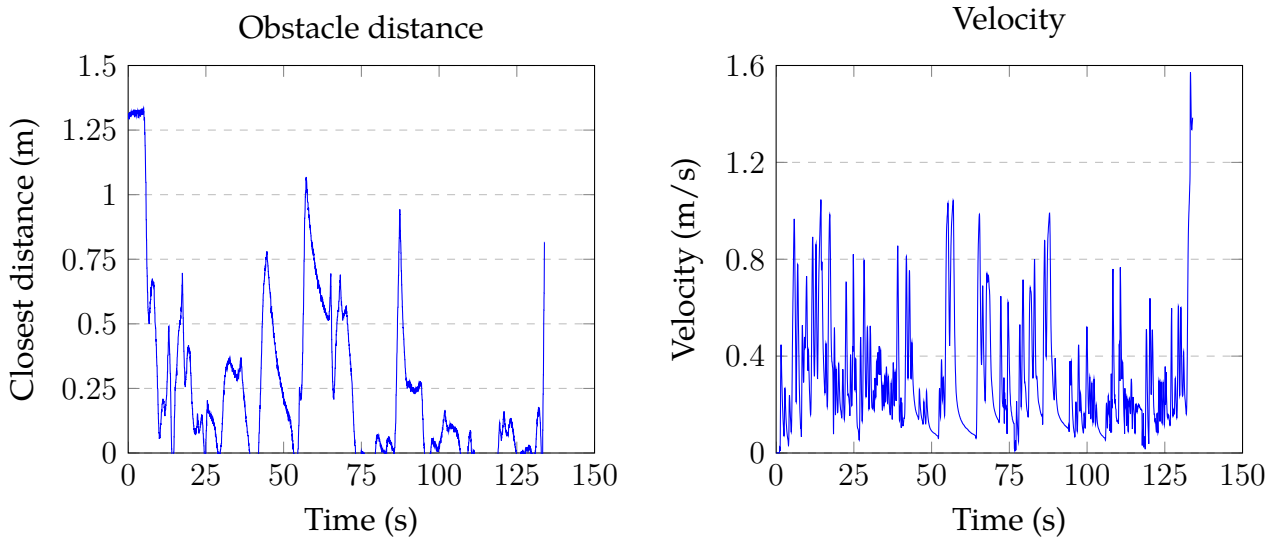
(a) The closest obstacle distance from sensor measurements over time.

(b) The velocity over time.

Figure A.42: Results for operator 4 from the user study with configuration 3.

Configuration 4

Figure A.43: The path for operator 4 with configuration 4 in the user study. Red is obstacles detected by sensor, blue is the UAV, and pink indicates collision.



(a) The closest obstacle distance from sensor measurements over time.

(b) The velocity over time.

Figure A.44: Results for operator 4 from the user study with configuration 4.

Configuration 5

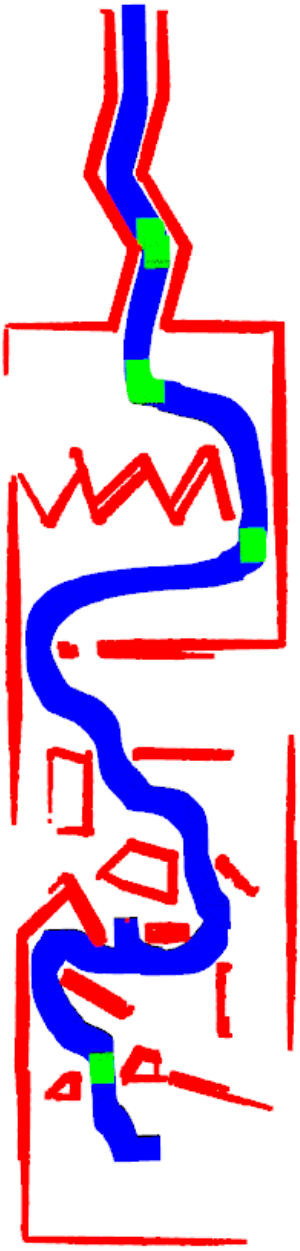
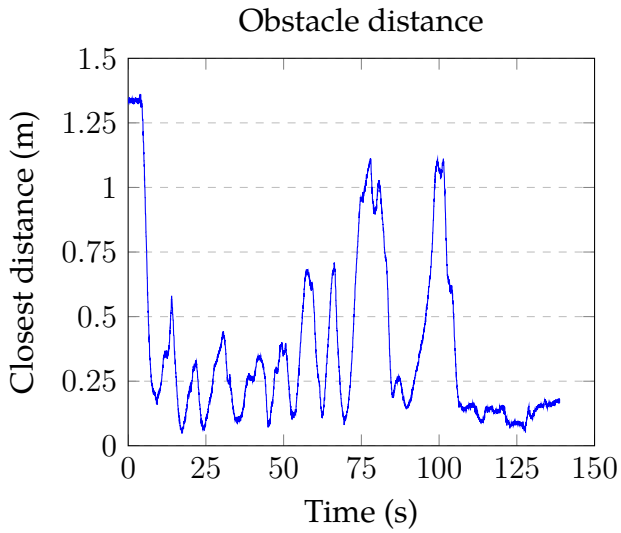
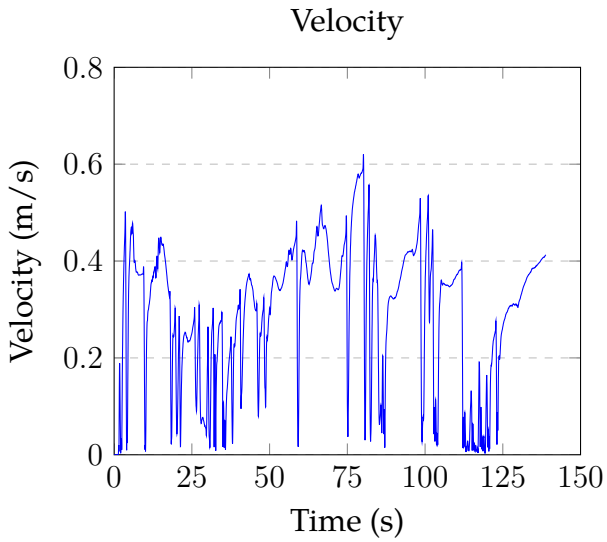


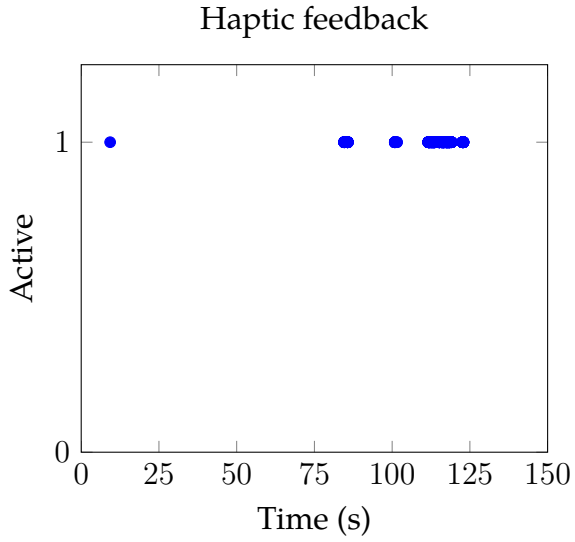
Figure A.45: The path for operator 4 with configuration 5 in the user study. Red is obstacles detected by sensor, blue is the UAV, and green is when haptic feedback is activated.



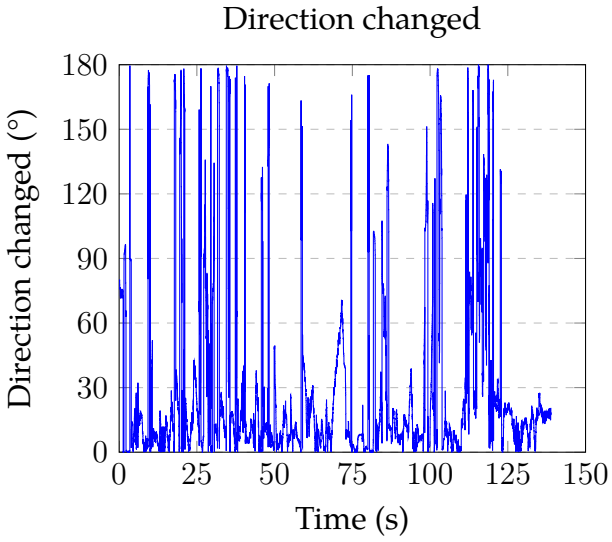
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.

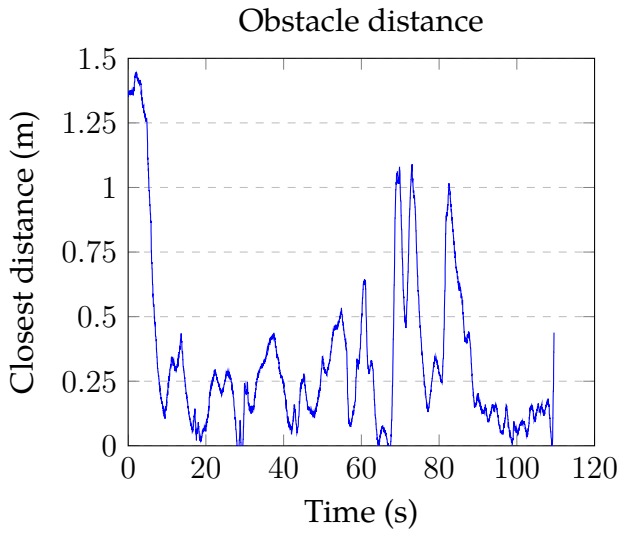


(d) Direction changed over time.

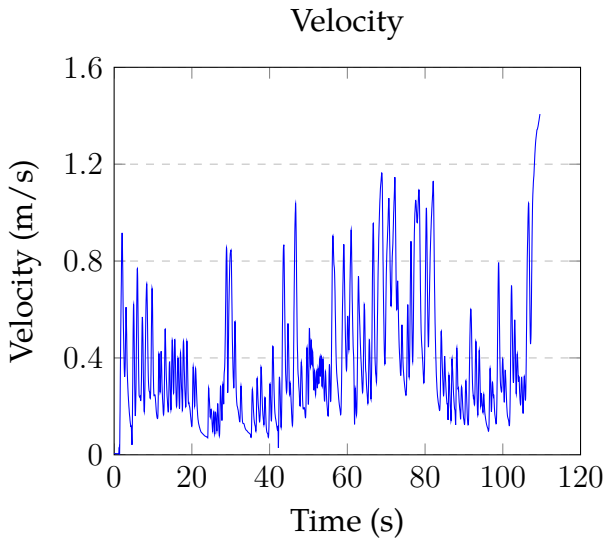
Figure A.46: Results for operator 4 from the user study with configuration 5.

Configuration 6

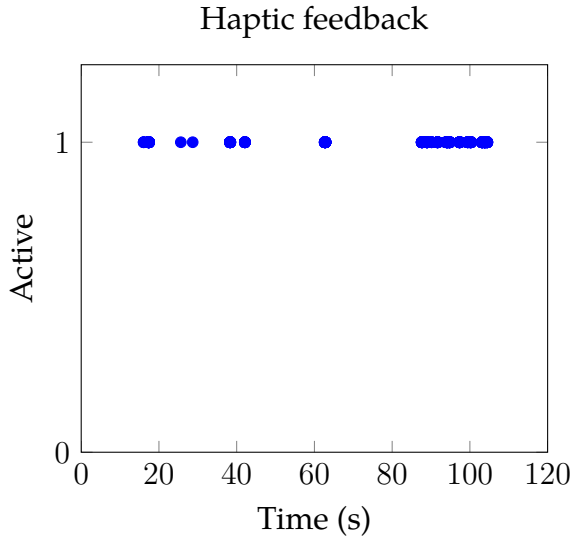
Figure A.47: The path for operator 4 with configuration 6 in the user study. Red is obstacles detected by sensor, blue is the UAV, green is when haptic feedback is activated, and pink indicates collision.



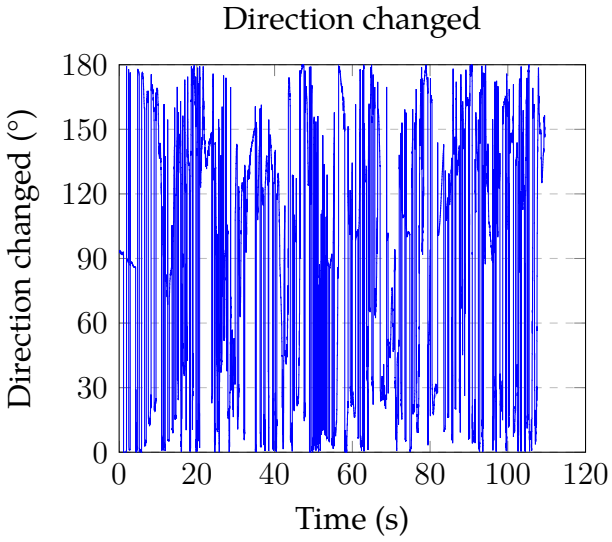
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.



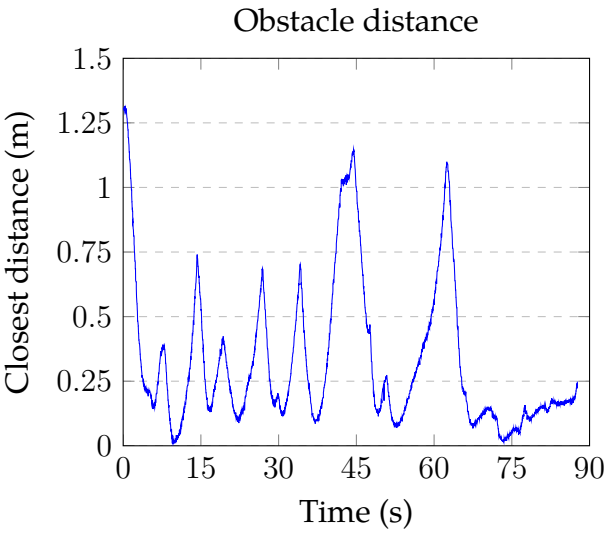
(d) Direction changed over time.

Figure A.48: Results for operator 4 from the user study with configuration 6.

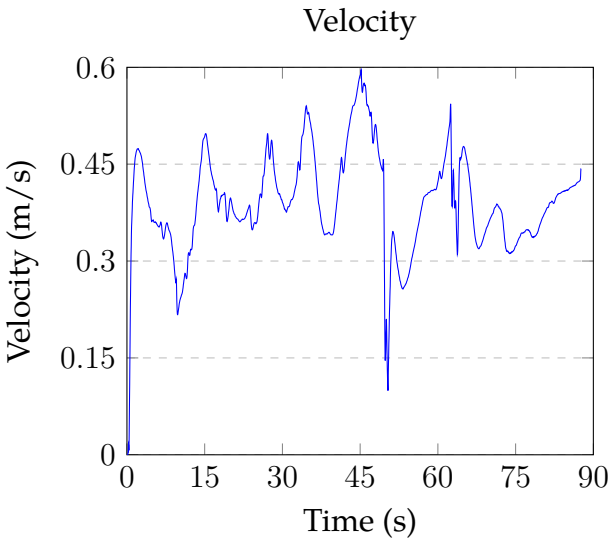
A.5 Operator 5

Configuration 1

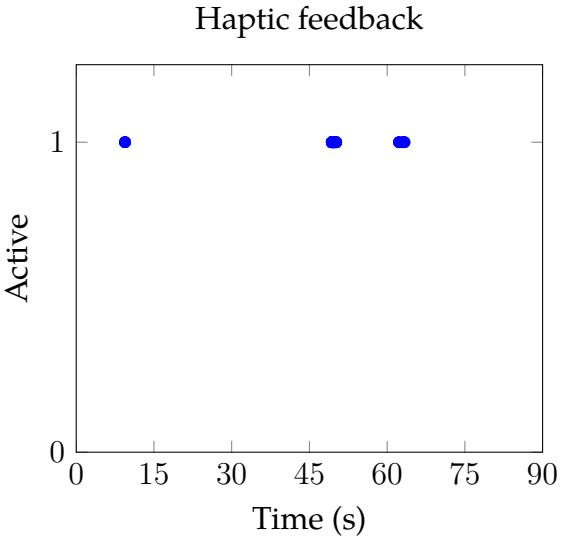
Figure A.49: The path for operator 5 with configuration 1 in the user study. Red is obstacles detected by sensor, blue is the UAV, and green is when haptic feedback is activated.



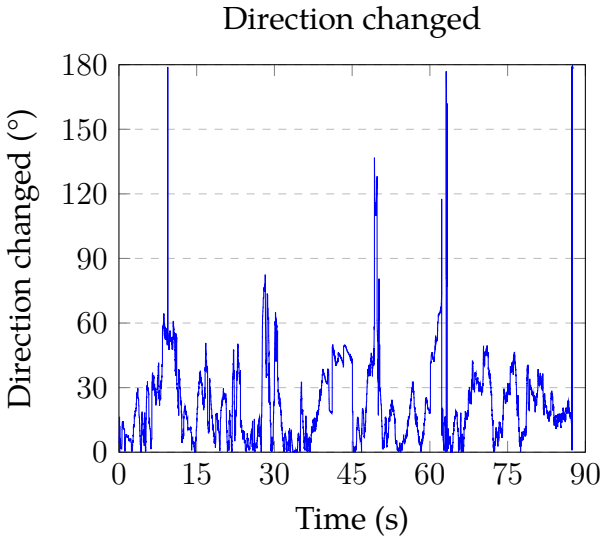
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.

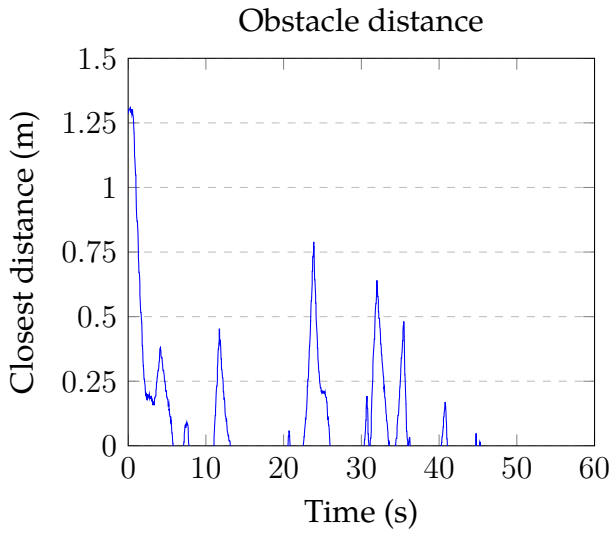


(d) Direction changed over time.

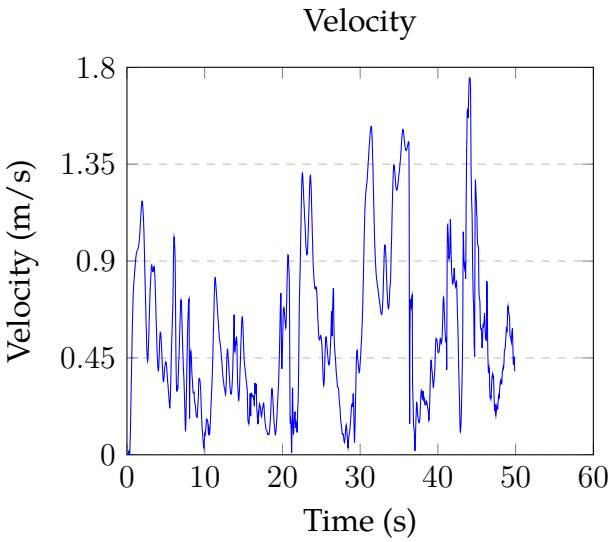
Figure A.50: Results for operator 5 from the user study with configuration 1.

Configuration 2

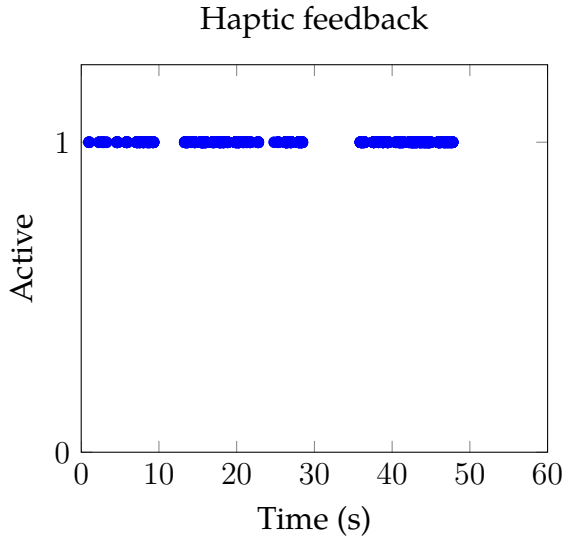
Figure A.51: The path for operator 5 with configuration 2 in the user study. Red is obstacles detected by sensor, blue is the UAV, green is when haptic feedback is activated, and pink indicates collision.



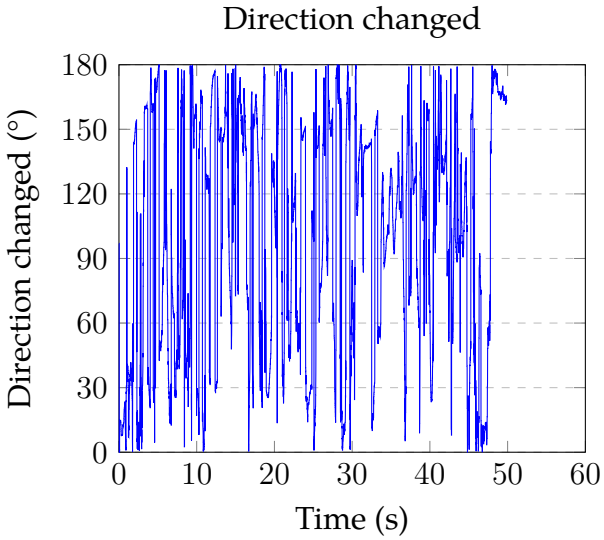
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.

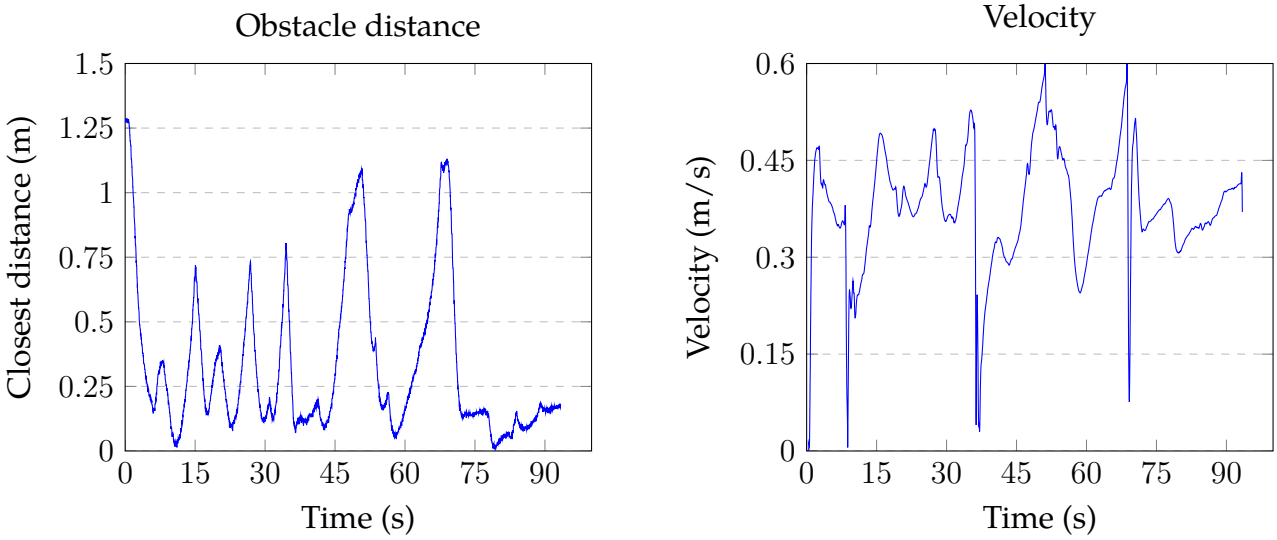


(d) Direction changed over time.

Figure A.52: Results for operator 5 from the user study with configuration 2.

Configuration 3

Figure A.53: The path for operator 5 with configuration 3 in the user study. Red is obstacles detected by sensor and blue is the UAV.



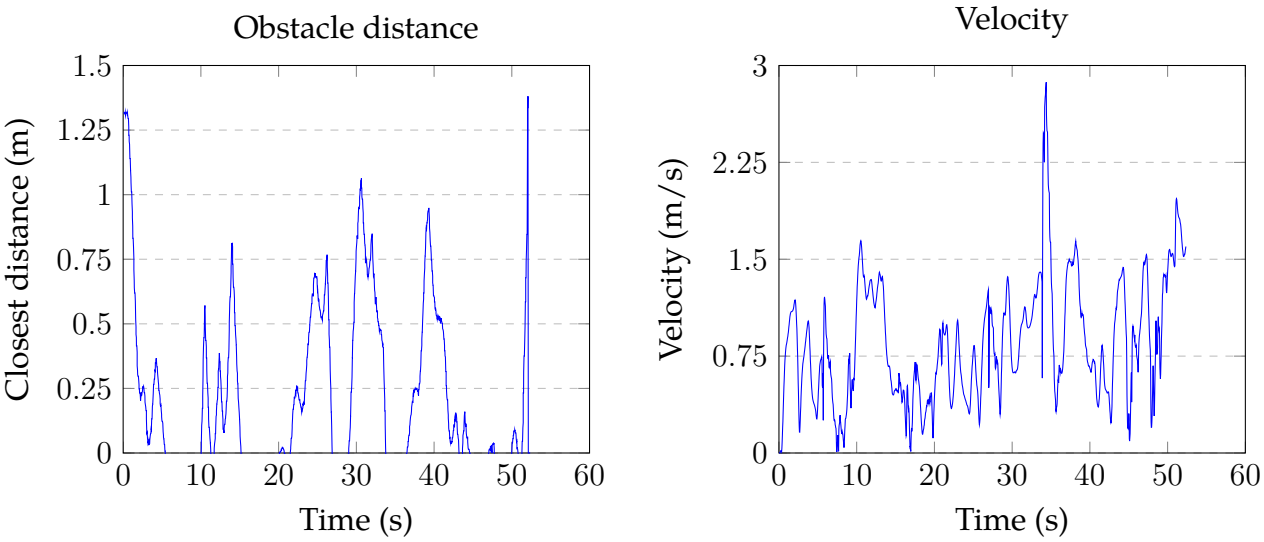
(a) The closest obstacle distance from sensor measurements over time.

(b) The velocity over time.

Figure A.54: Results for operator 5 from the user study with configuration 3.

Configuration 4

Figure A.55: The path for operator 5 with configuration 4 in the user study. Red is obstacles detected by sensor, blue is the UAV, and pink indicates collision.



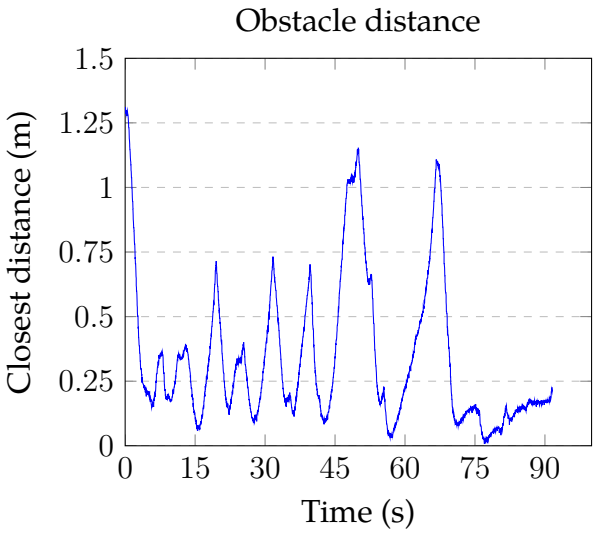
(a) The closest obstacle distance from sensor measurements over time.

(b) The velocity over time.

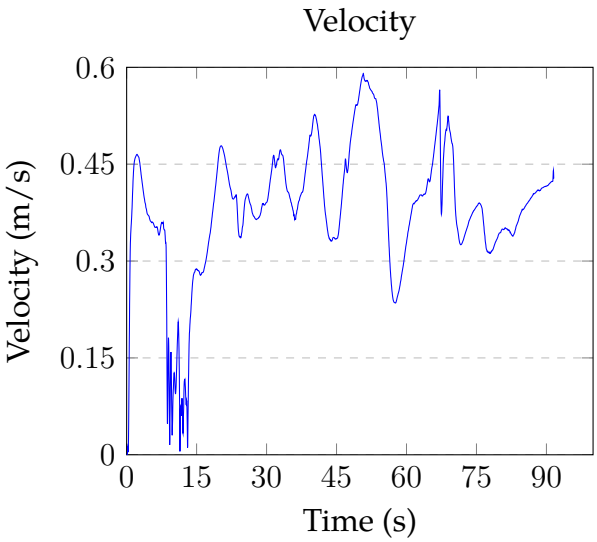
Figure A.56: Results for operator 5 from the user study with configuration 4.

Configuration 5

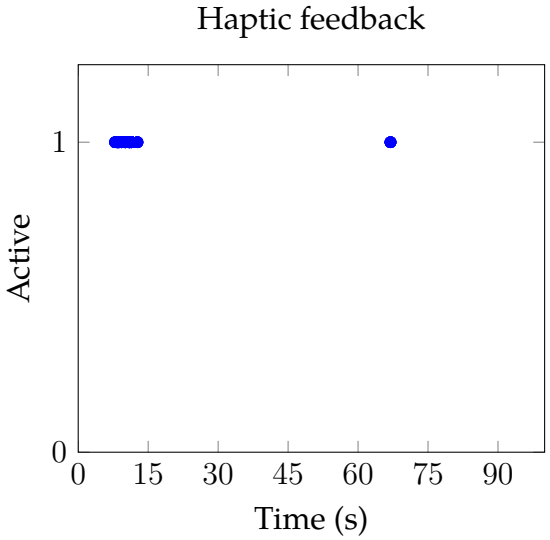
Figure A.57: The path for operator 5 with configuration 5 in the user study. Red is obstacles detected by sensor, blue is the UAV, and green is when haptic feedback is activated.



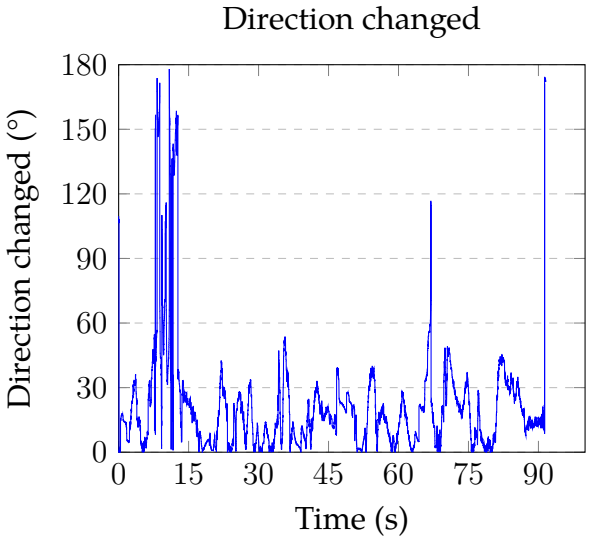
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.

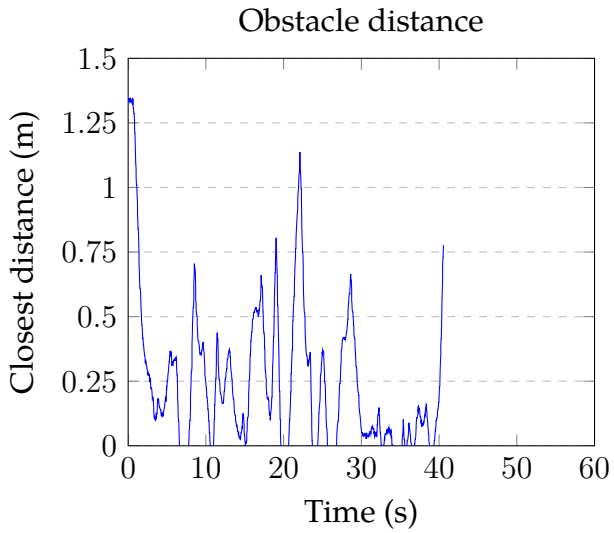


(d) Direction changed over time.

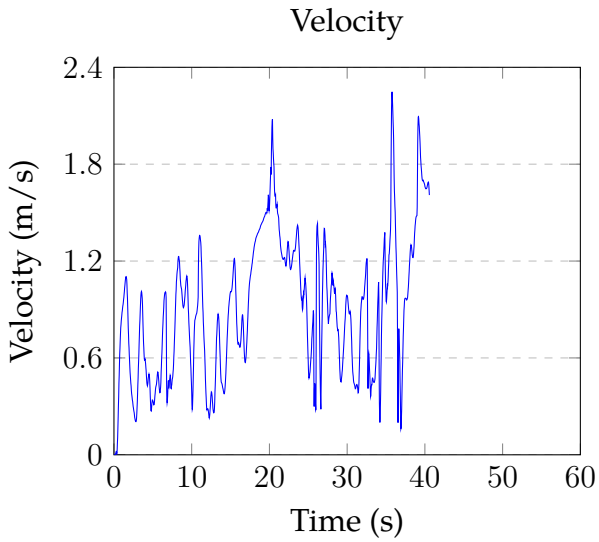
Figure A.58: Results for operator 5 from the user study with configuration 5.

Configuration 6

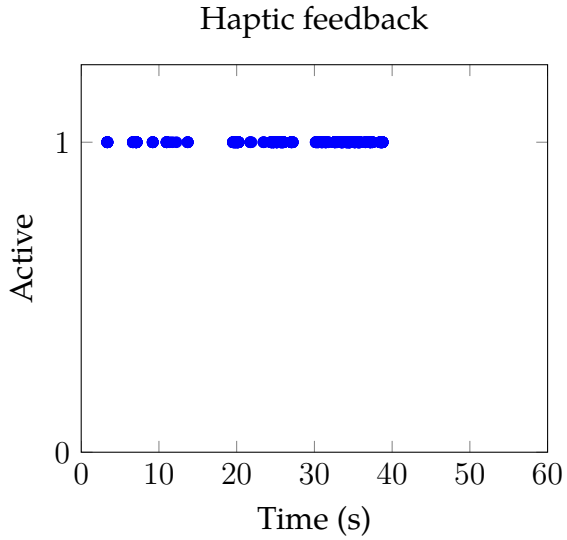
Figure A.59: The path for operator 5 with configuration 6 in the user study. Red is obstacles detected by sensor, blue is the UAV, green is when haptic feedback is activated, and pink indicates collision.



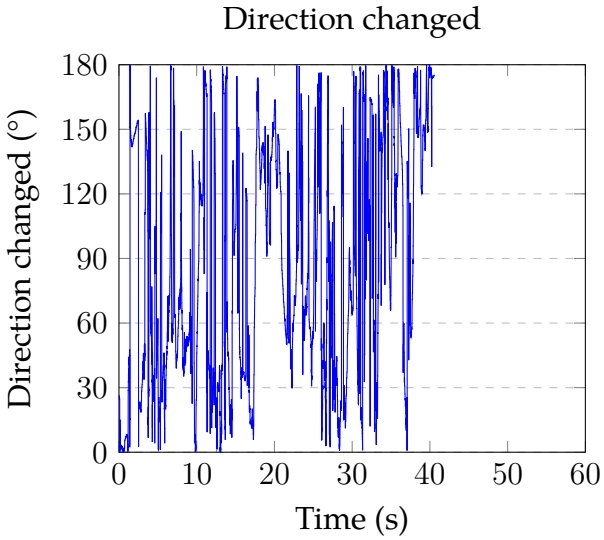
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.



(d) Direction changed over time.

Figure A.60: Results for operator 5 from the user study with configuration 6.

A.6 Operator 6

Configuration 1

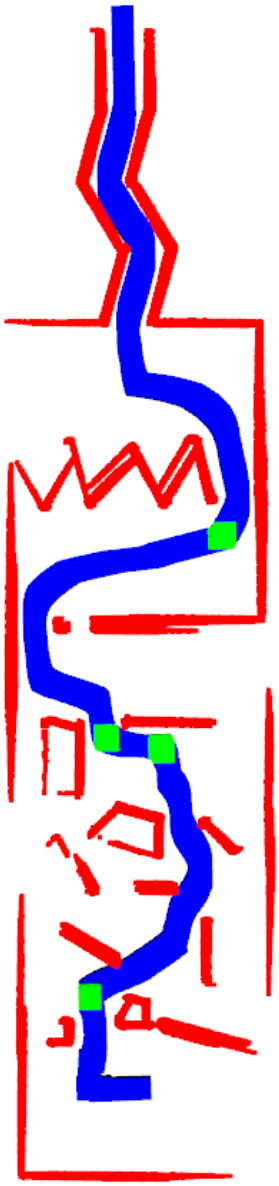
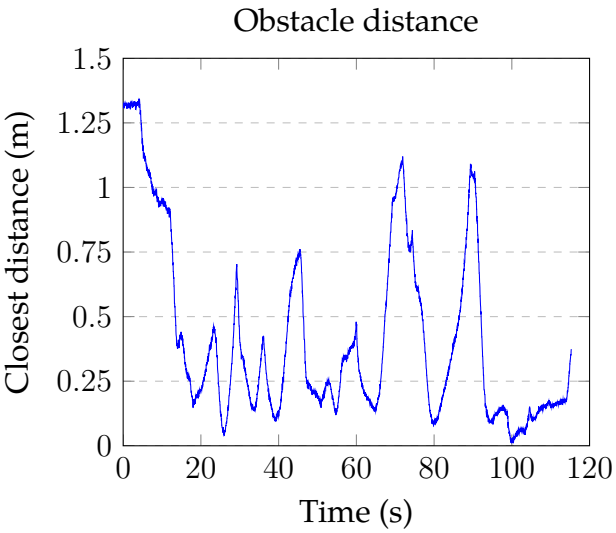
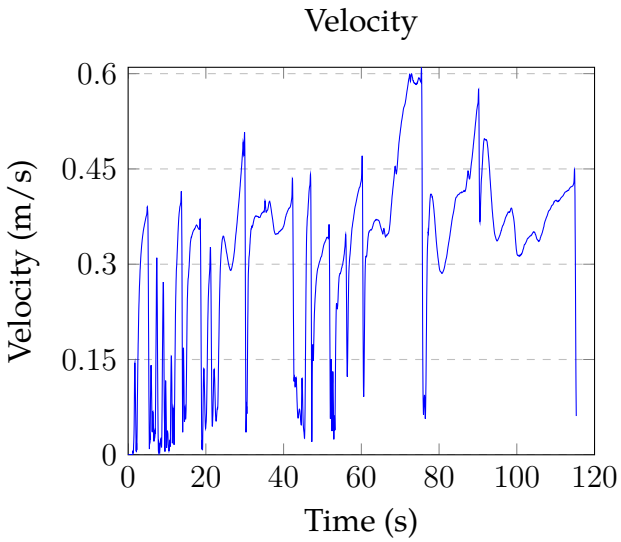


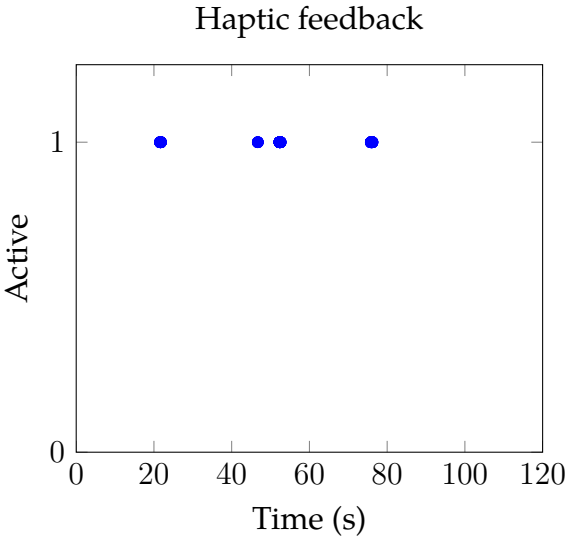
Figure A.61: The path for operator 6 with configuration 1 in the user study. Red is obstacles detected by sensor, blue is the UAV, and green is when haptic feedback is activated.



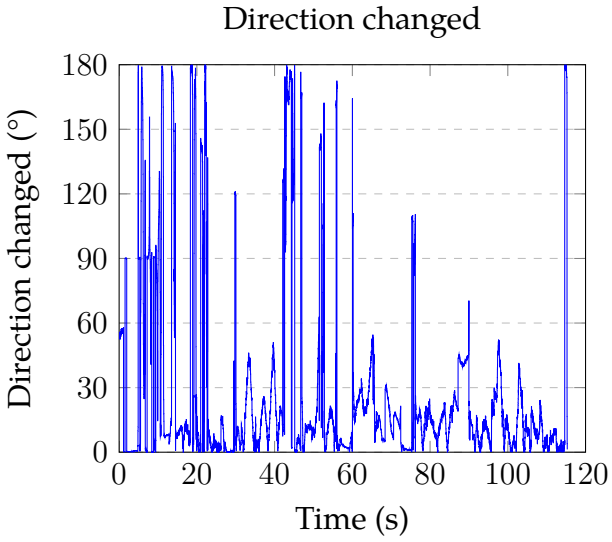
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.

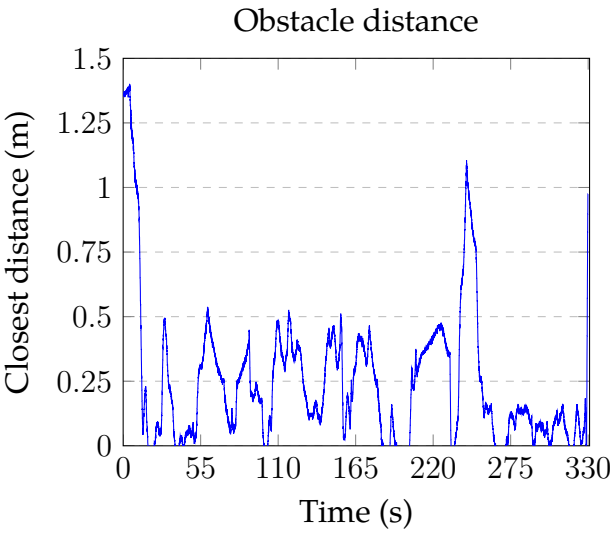


(d) Direction changed over time.

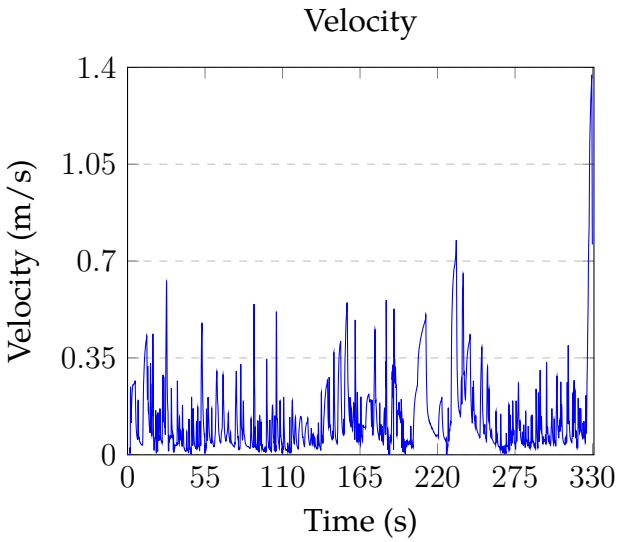
Figure A.62: Results for operator 6 from the user study with configuration 1.

Configuration 2

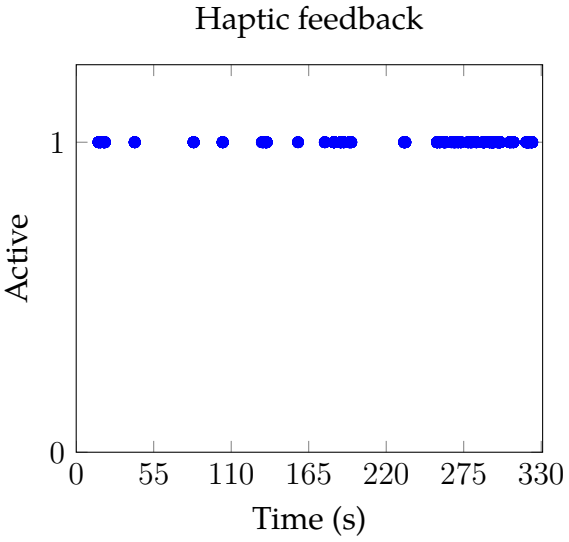
Figure A.63: The path for operator 6 with configuration 2 in the user study. Red is obstacles detected by sensor, blue is the UAV, green is when haptic feedback is activated, and pink indicates collision.



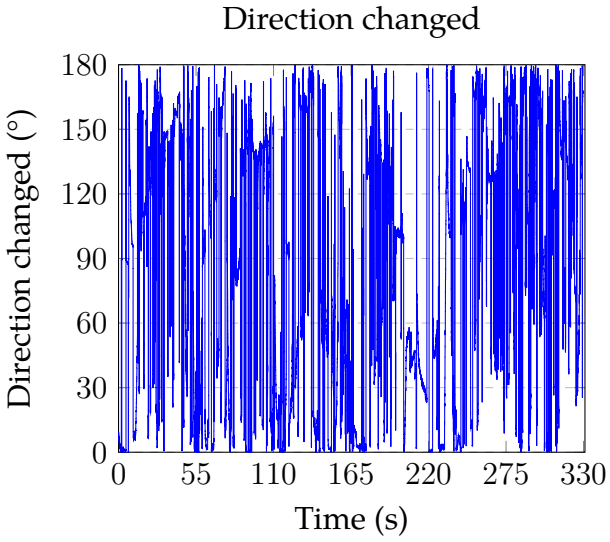
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.

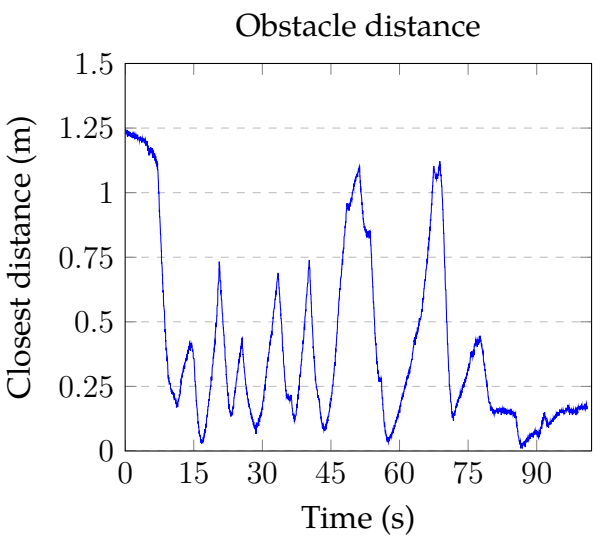


(d) Direction changed over time.

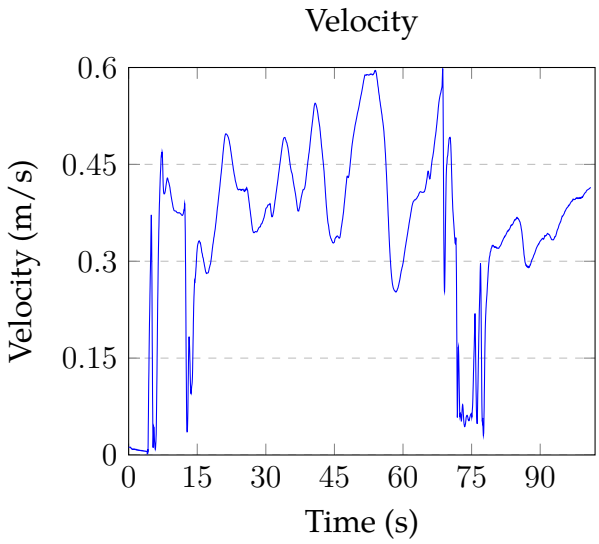
Figure A.64: Results for operator 6 from the user study with configuration 2.

Configuration 3

Figure A.65: The path for operator 6 with configuration 3 in the user study. Red is obstacles detected by sensor and blue is the UAV.



(a) The closest obstacle distance from sensor measurements over time.

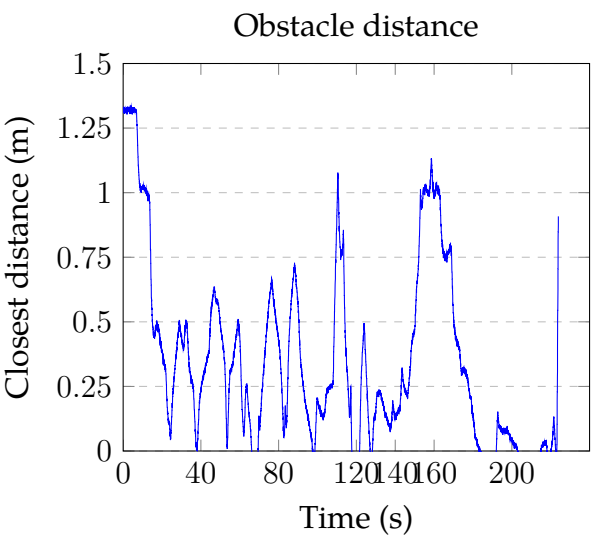


(b) The velocity over time.

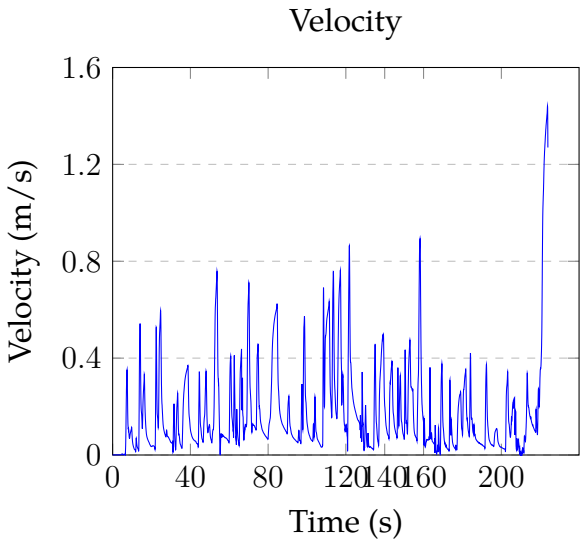
Figure A.66: Results for operator 6 from the user study with configuration 3.

Configuration 4

Figure A.67: The path for operator 6 with configuration 4 in the user study. Red is obstacles detected by sensor, blue is the UAV, and pink indicates collision.



(a) The closest obstacle distance from sensor measurements over time.

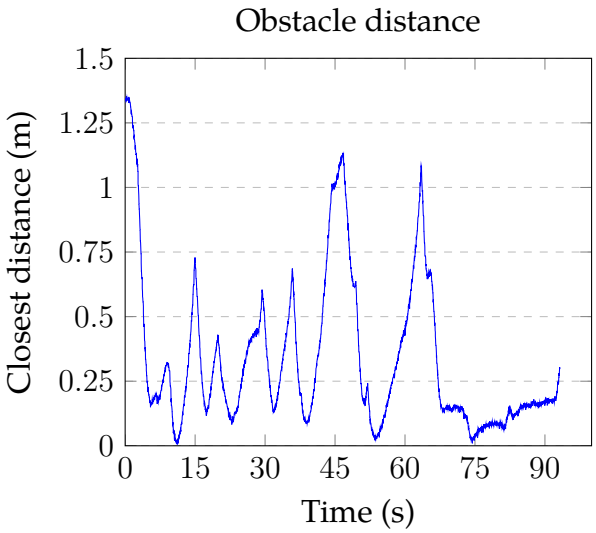


(b) The velocity over time.

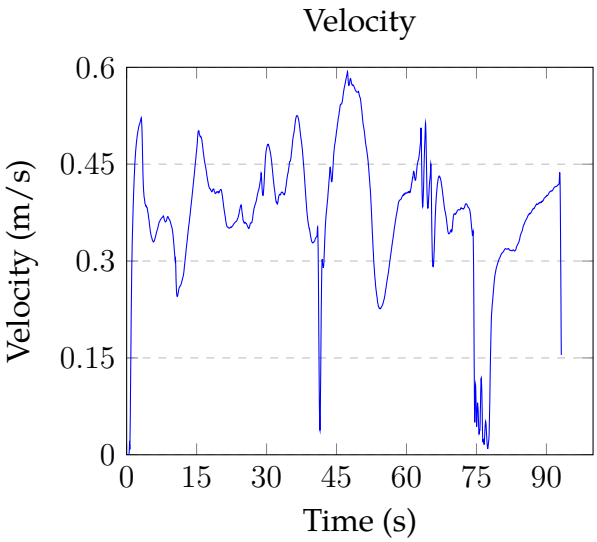
Figure A.68: Results for operator 6 from the user study with configuration 4.

Configuration 5

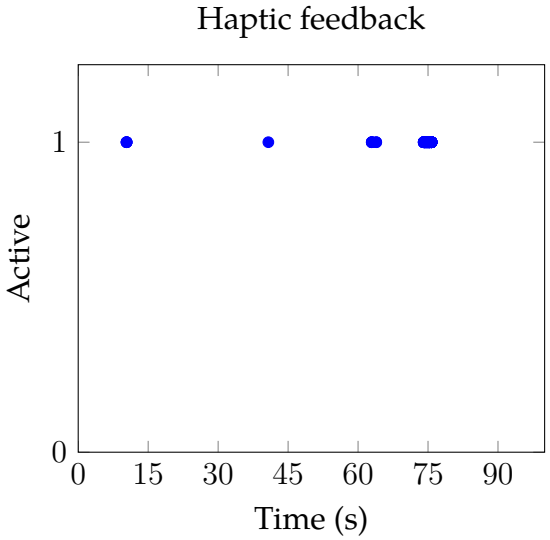
Figure A.69: The path for operator 6 with configuration 5 in the user study. Red is obstacles detected by sensor, blue is the UAV, and green is when haptic feedback is activated.



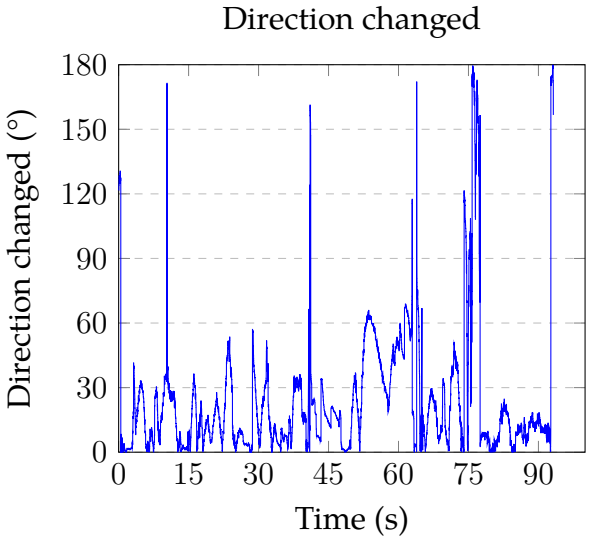
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.



(d) Direction changed over time.

Figure A.70: Results for operator 6 from the user study with configuration 5.

Configuration 6

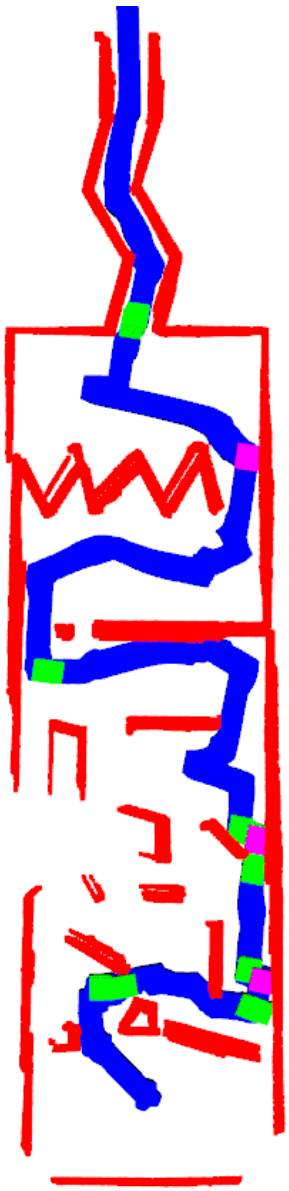
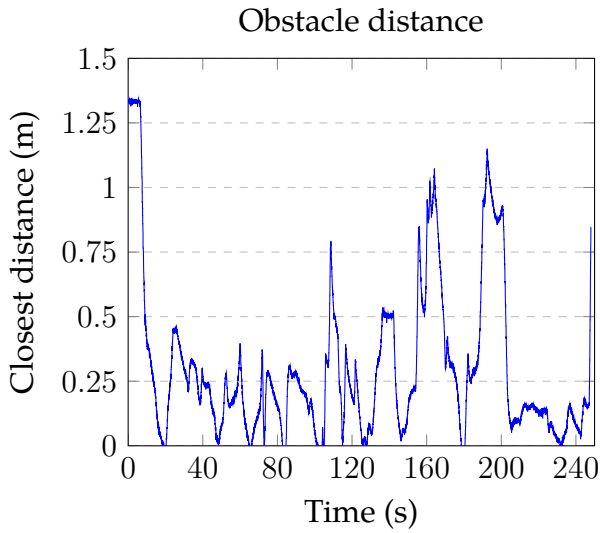
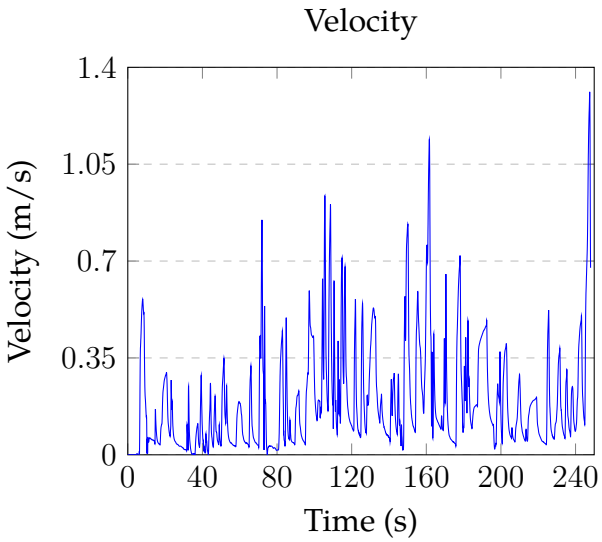


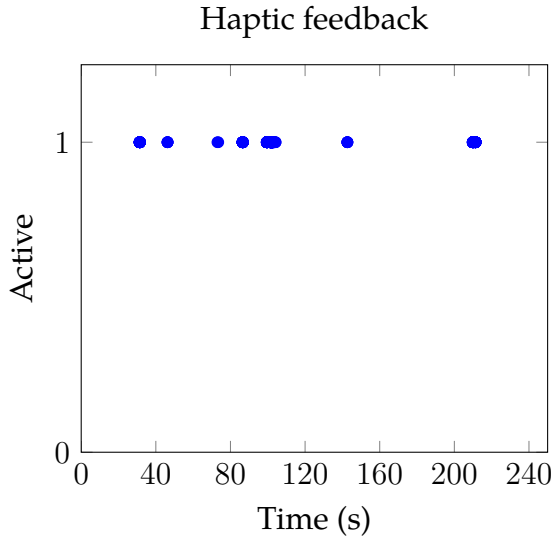
Figure A.71: The path for operator 6 with configuration 6 in the user study. Red is obstacles detected by sensor, blue is the UAV, green is when haptic feedback is activated, and pink indicates collision.



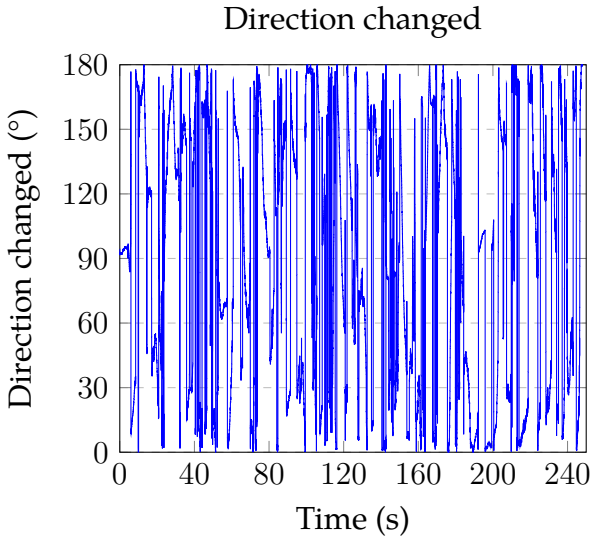
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.



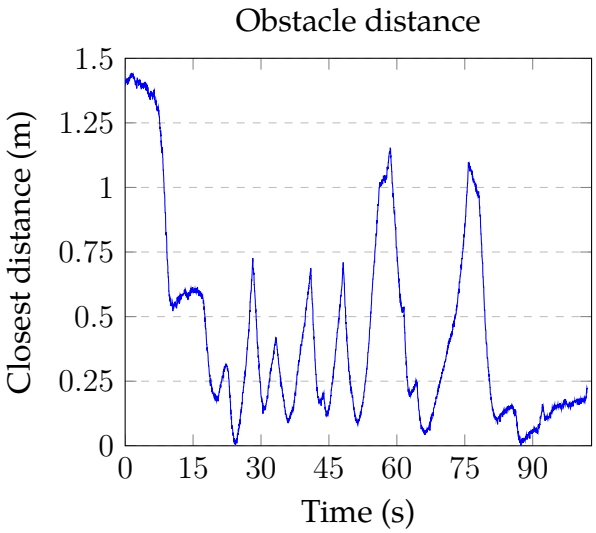
(d) Direction changed over time.

Figure A.72: Results for operator 6 from the user study with configuration 6.

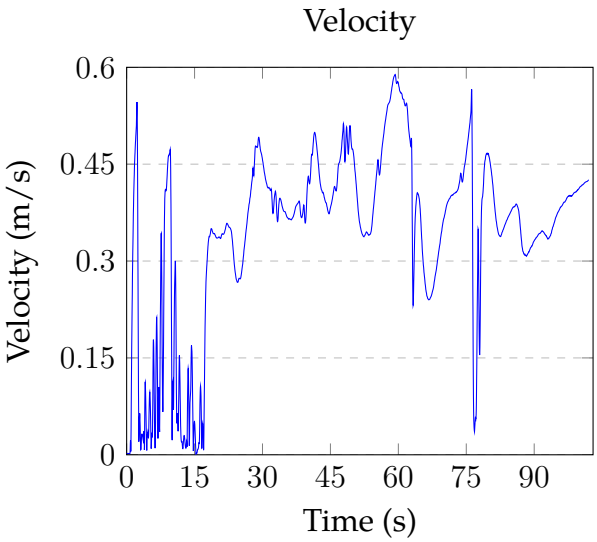
A.7 Operator 7

Configuration 1

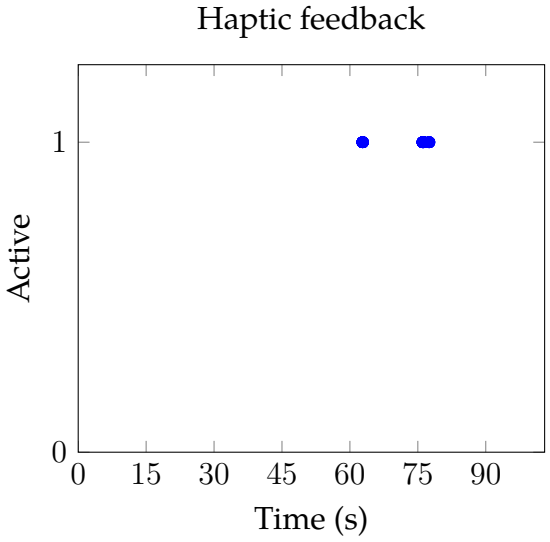
Figure A.73: The path for operator 7 with configuration 1 in the user study. Red is obstacles detected by sensor, blue is the UAV, and green is when haptic feedback is activated.



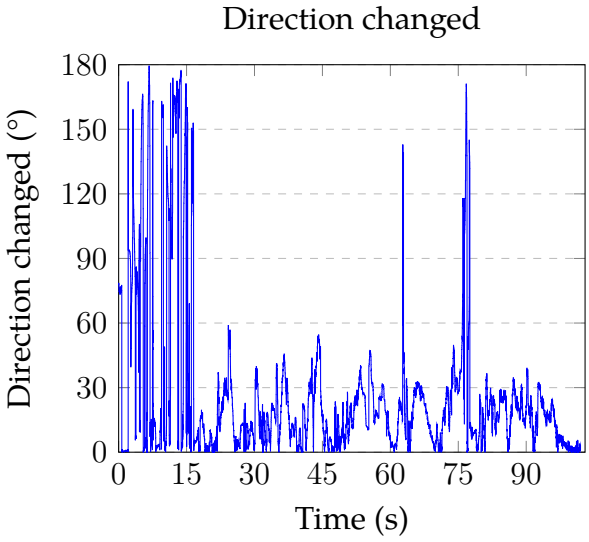
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.

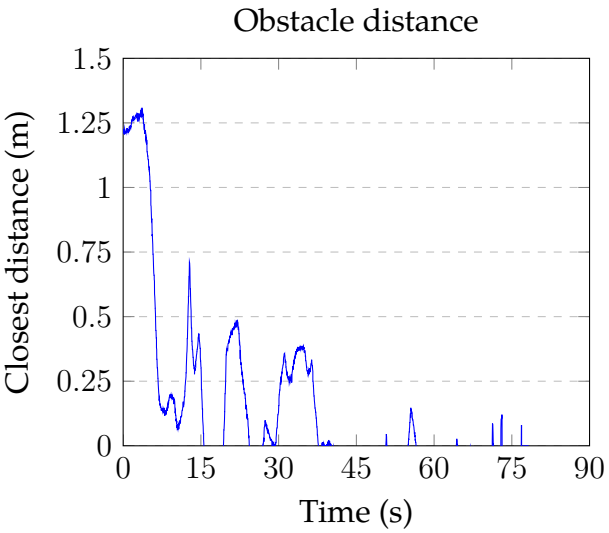


(d) Direction changed over time.

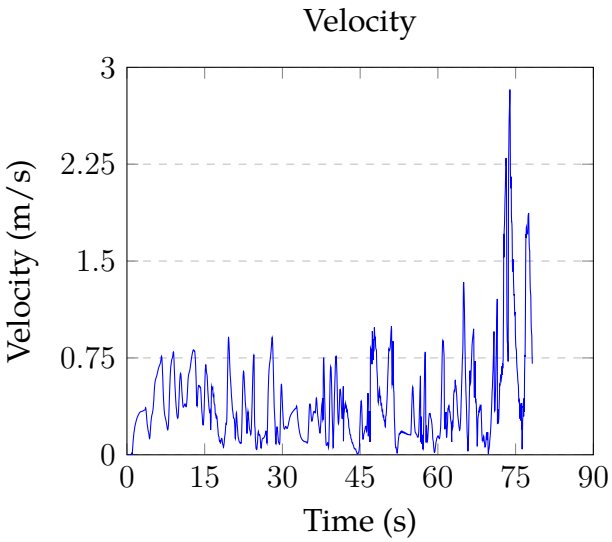
Figure A.74: Results for operator 7 from the user study with configuration 1.

Configuration 2

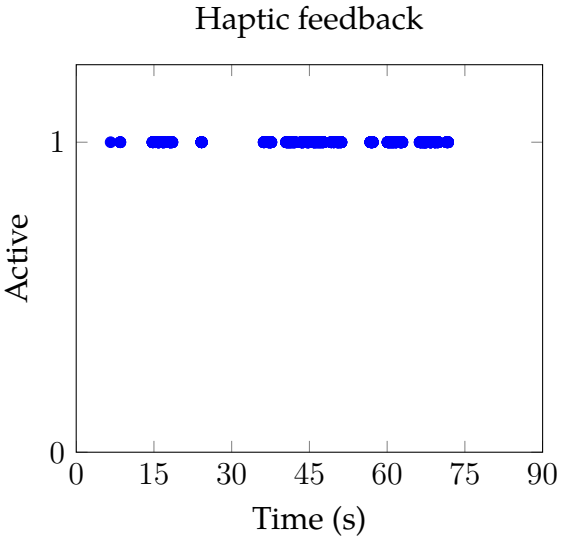
Figure A.75: The path for operator 7 with configuration 2 in the user study. Red is obstacles detected by sensor, blue is the UAV, green is when haptic feedback is activated, and pink indicates collision.



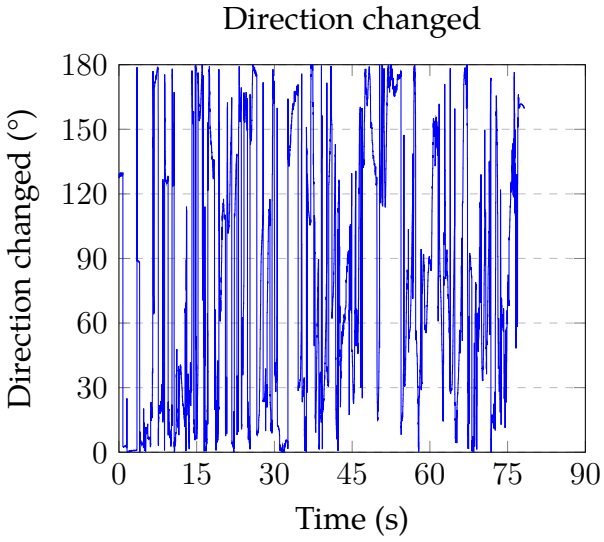
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.

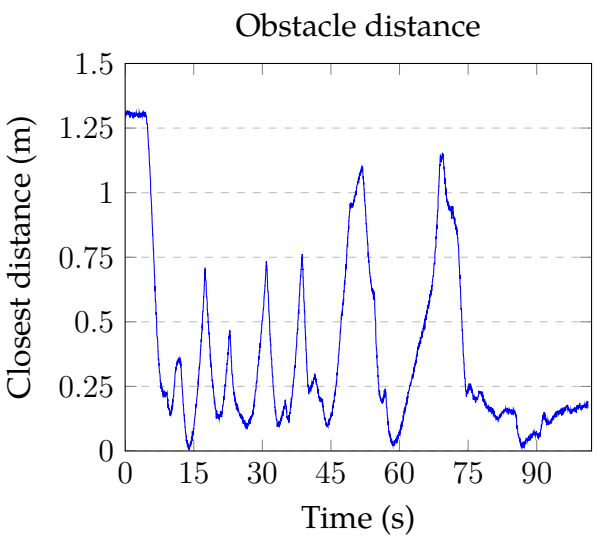


(d) Direction changed over time.

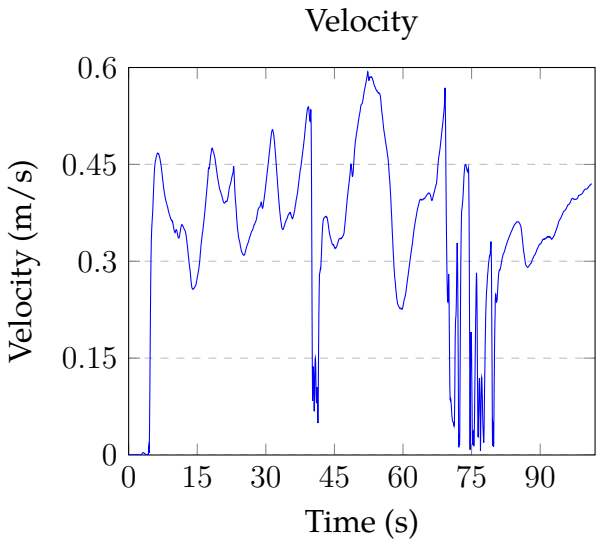
Figure A.76: Results for operator 7 from the user study with configuration 2.

Configuration 3

Figure A.77: The path for operator 7 with configuration 3 in the user study. Red is obstacles detected by sensor and blue is the UAV.



(a) The closest obstacle distance from sensor measurements over time.

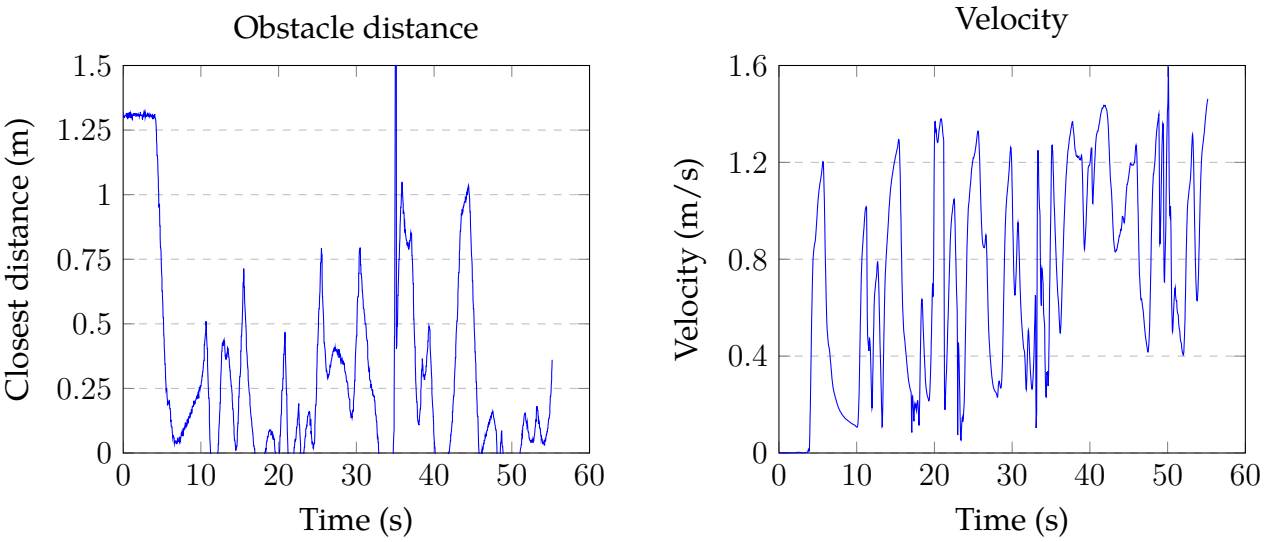


(b) The velocity over time.

Figure A.78: Results for operator 7 from the user study with configuration 3.

Configuration 4

Figure A.79: The path for operator 7 with configuration 4 in the user study. Red is obstacles detected by sensor, blue is the UAV, and pink indicates collision.



(a) The closest obstacle distance from sensor measurements over time.

(b) The velocity over time.

Figure A.80: Results for operator 7 from the user study with configuration 4.

Configuration 5

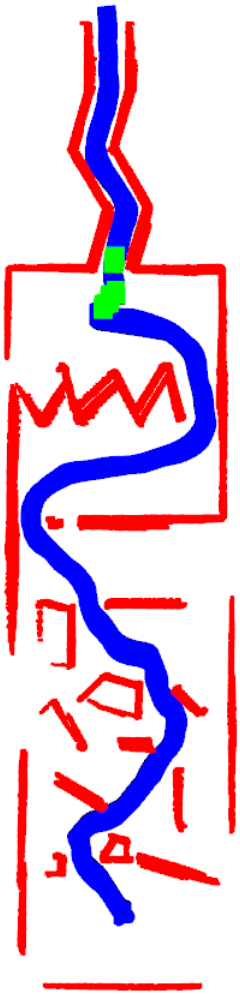
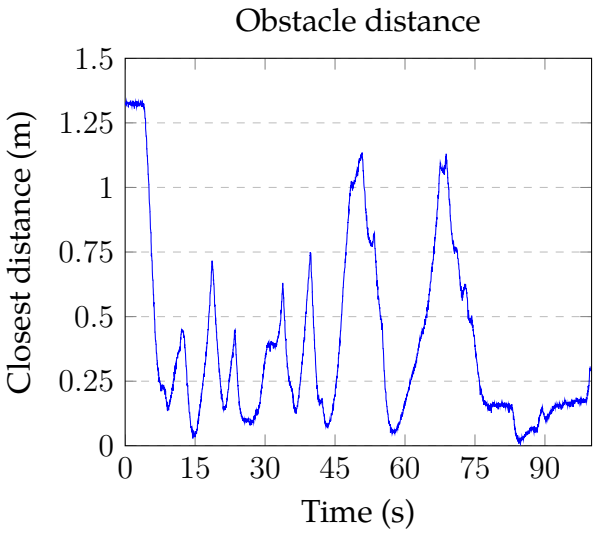
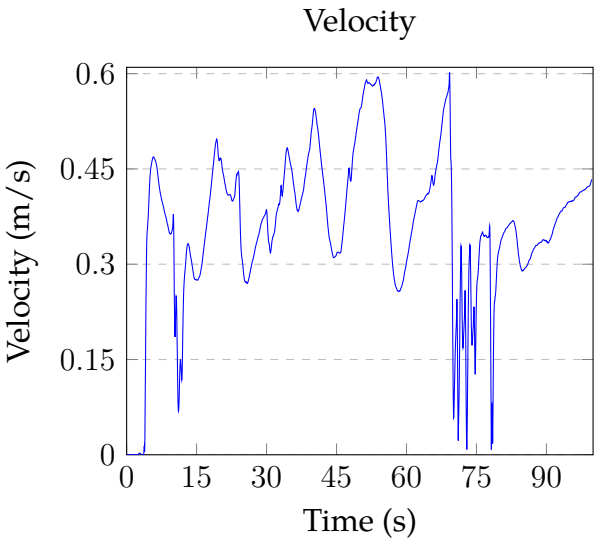


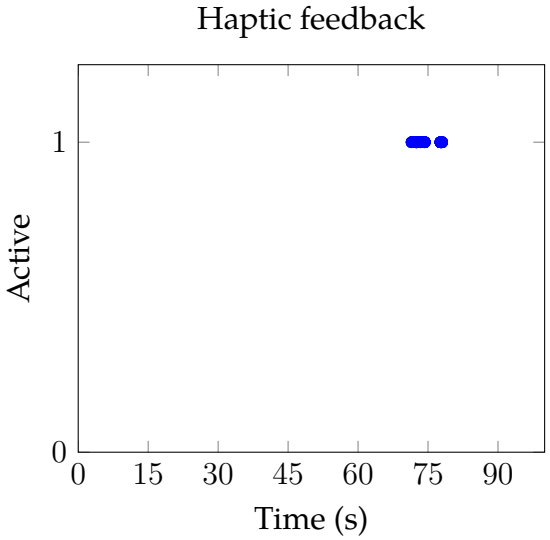
Figure A.81: The path for operator 7 with configuration 5 in the user study. Red is obstacles detected by sensor, blue is the UAV, and green is when haptic feedback is activated.



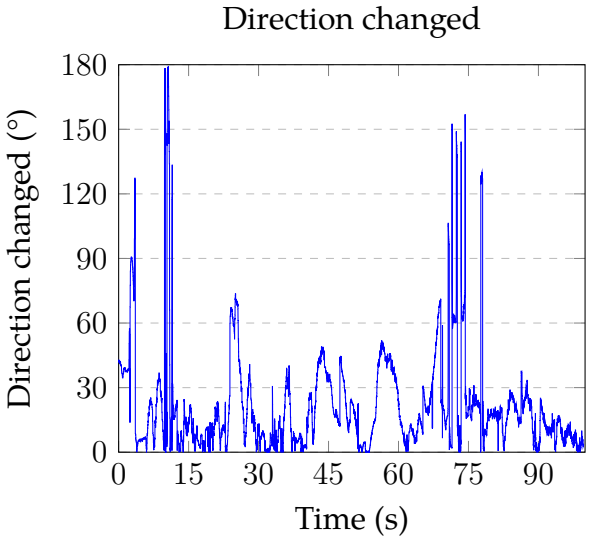
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.

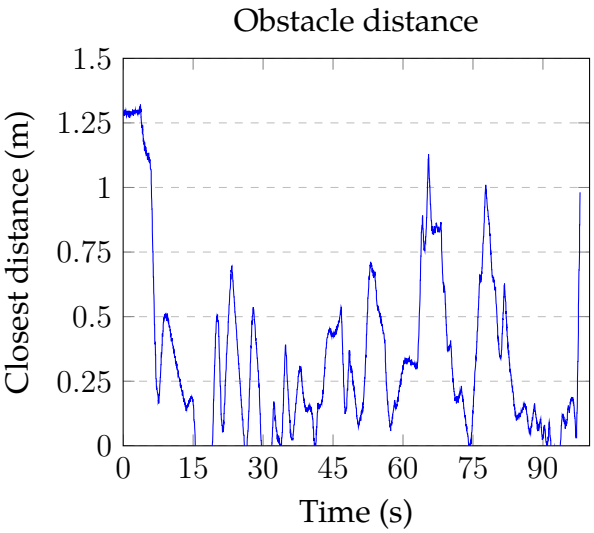


(d) Direction changed over time.

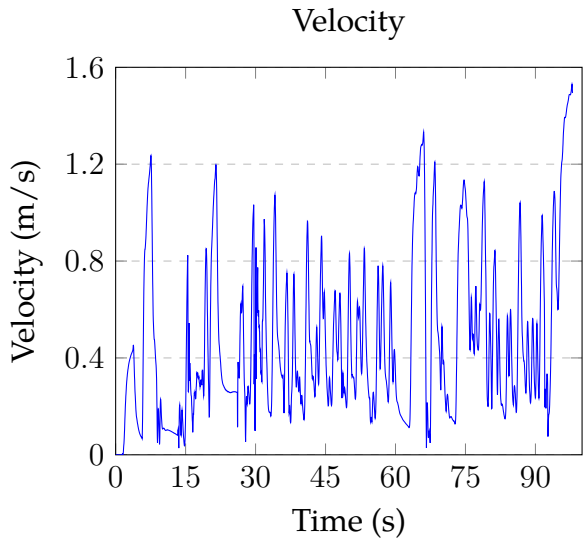
Figure A.82: Results for operator 7 from the user study with configuration 5.

Configuration 6

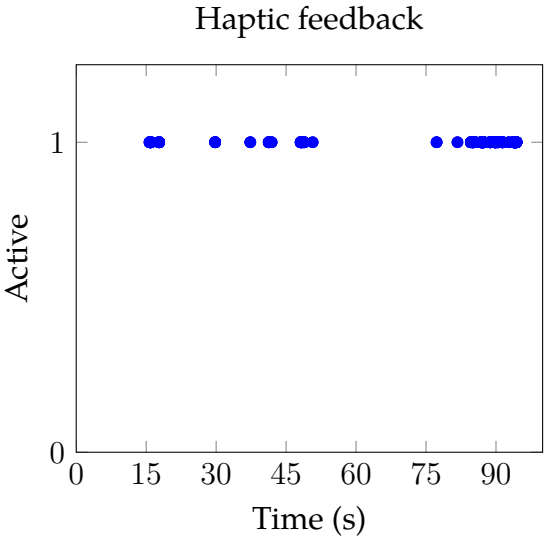
Figure A.83: The path for operator 7 with configuration 6 in the user study. Red is obstacles detected by sensor, blue is the UAV, green is when haptic feedback is activated, and pink indicates collision.



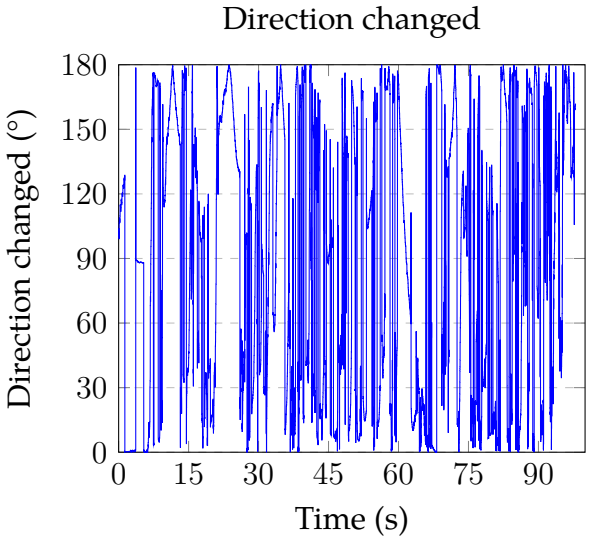
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.



(d) Direction changed over time.

Figure A.84: Results for operator 7 from the user study with configuration 6.

A.8 Operator 8

Configuration 1

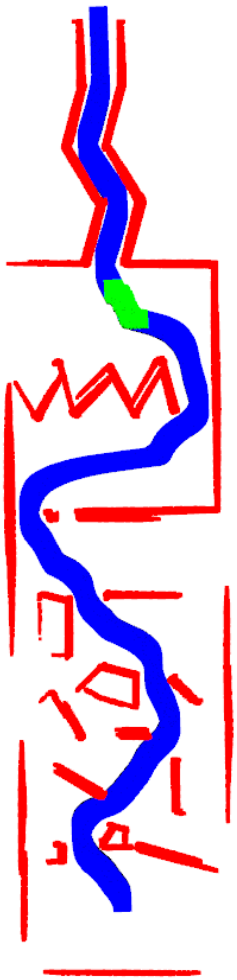
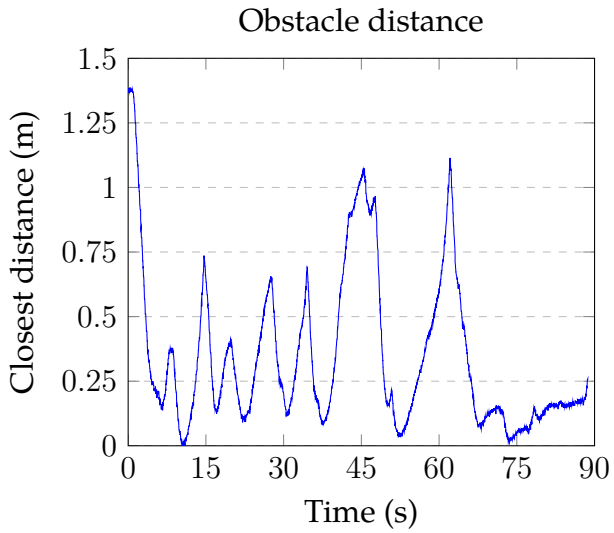
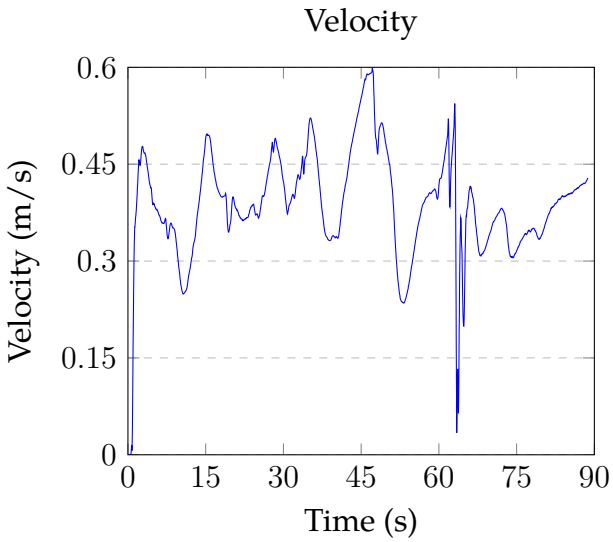


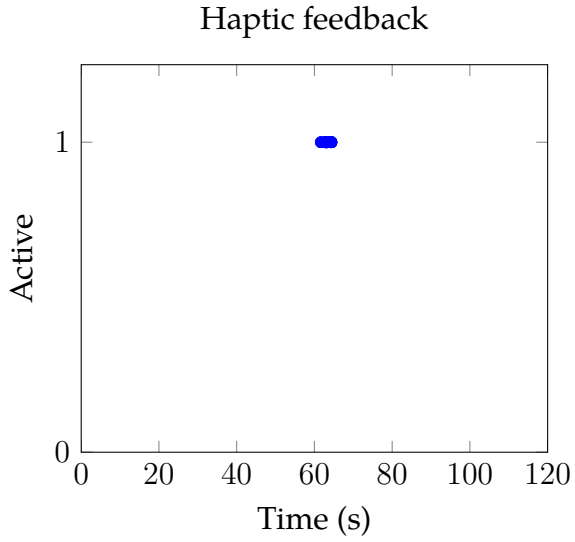
Figure A.85: The path for operator 8 with configuration 1 in the user study. Red is obstacles detected by sensor, blue is the UAV, and green is when haptic feedback is activated.



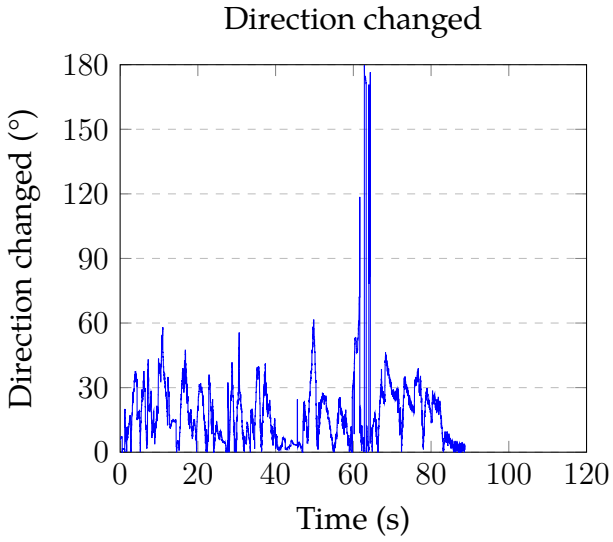
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.



(d) Direction changed over time.

Figure A.86: Results for operator 8 from the user study with configuration 1.

Configuration 2

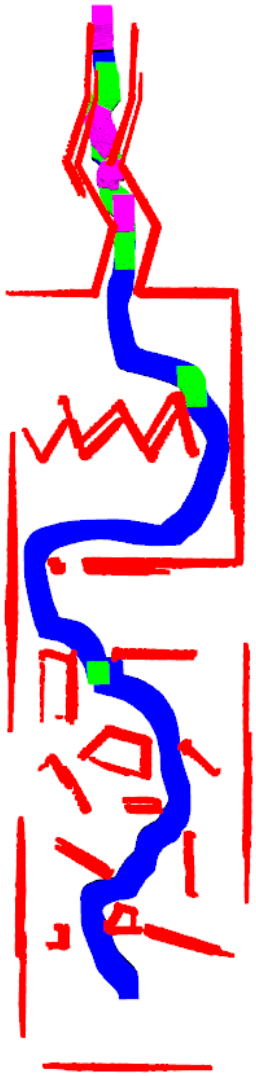
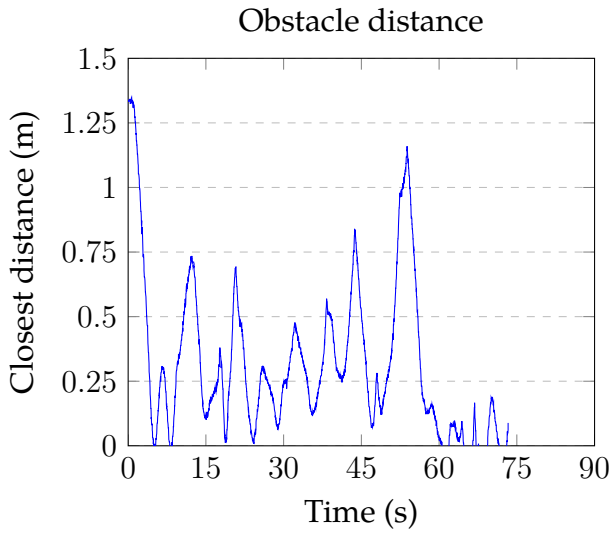
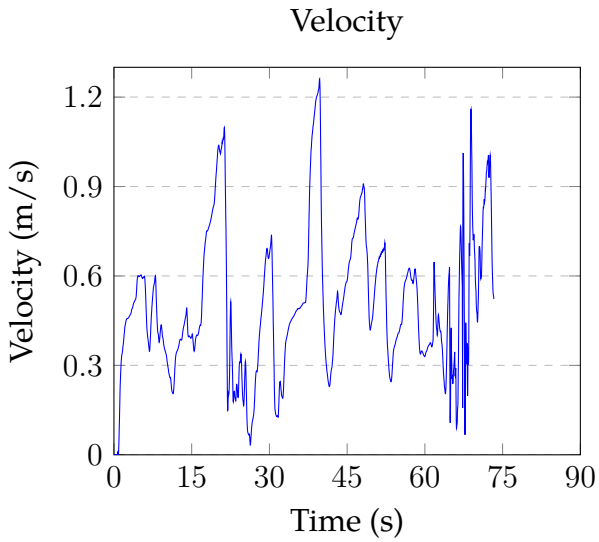


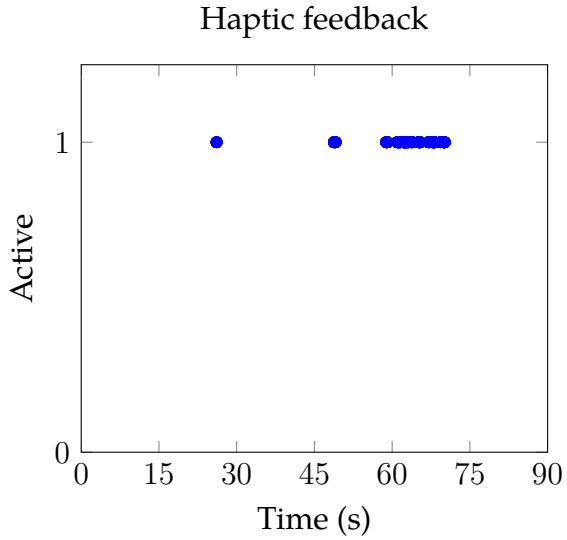
Figure A.87: The path for operator 8 with configuration 2 in the user study. Red is obstacles detected by sensor, blue is the UAV, green is when haptic feedback is activated, and pink indicates collision.



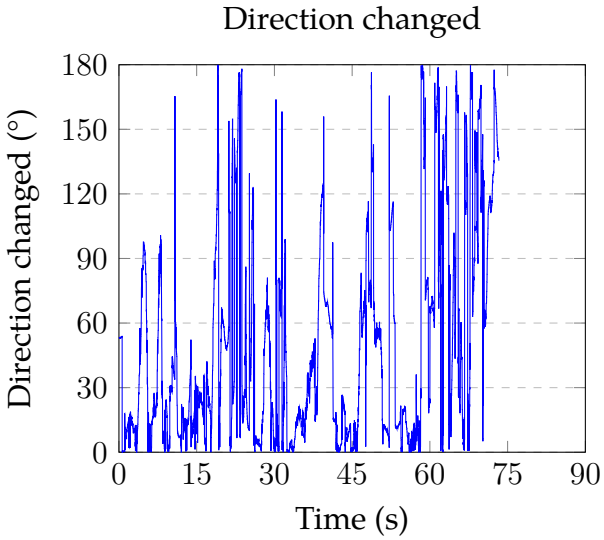
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.

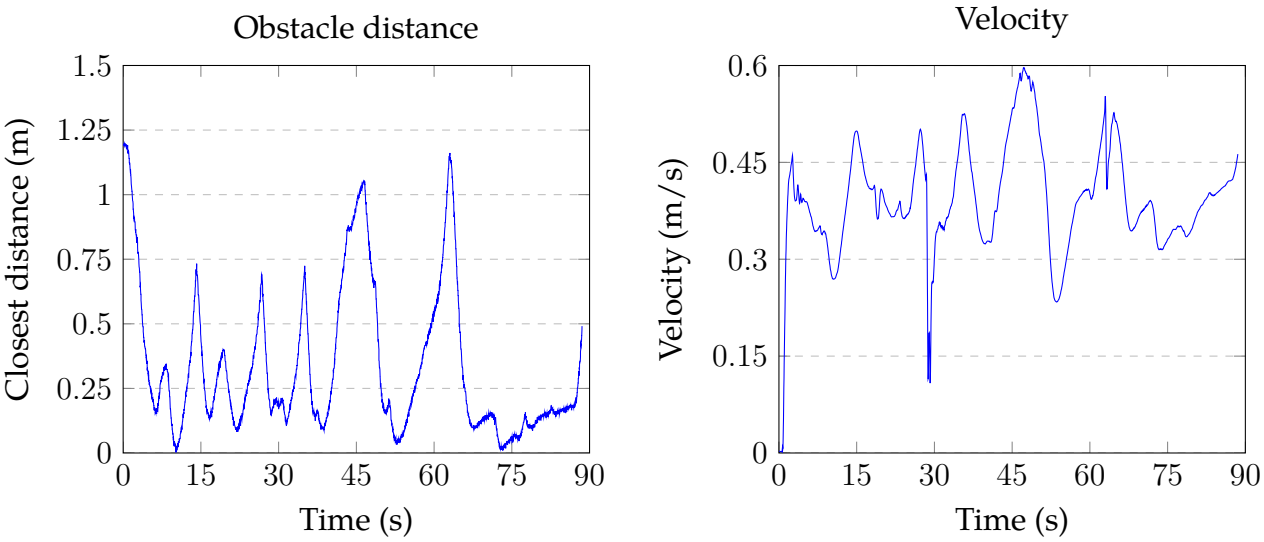


(d) Direction changed over time.

Figure A.88: Results for operator 8 from the user study with configuration 2.

Configuration 3

Figure A.89: The path for operator 8 with configuration 3 in the user study. Red is obstacles detected by sensor and blue is the UAV.



(a) The closest obstacle distance from sensor measurements over time.

(b) The velocity over time.

Figure A.90: Results for operator 8 from the user study with configuration 3.

Configuration 4

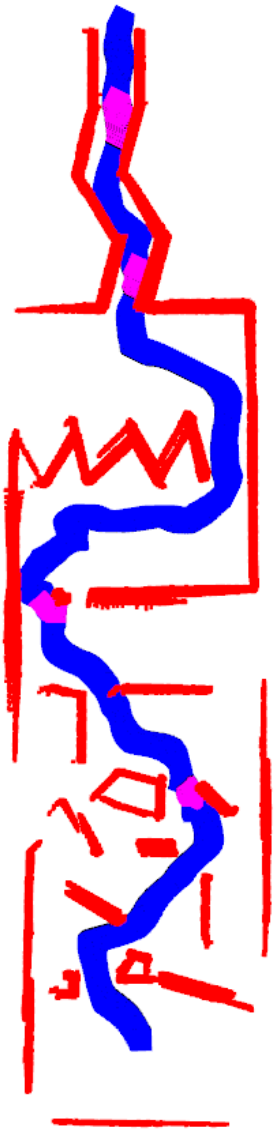
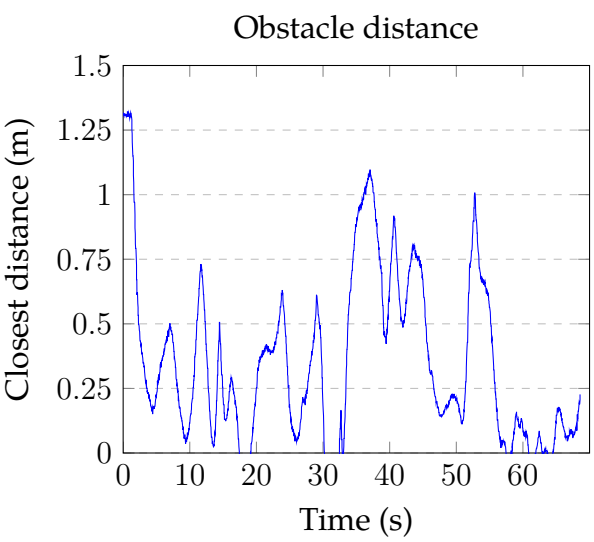
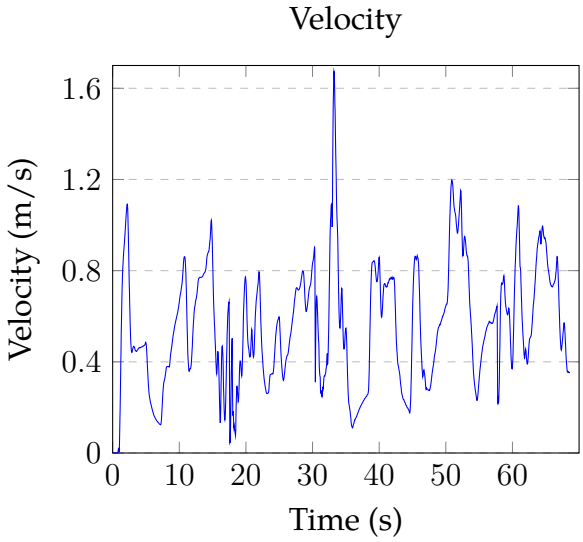


Figure A.91: The path for operator 8 with configuration 4 in the user study. Red is obstacles detected by sensor, blue is the UAV, and pink indicates collision.



(a) The closest obstacle distance from sensor measurements over time.

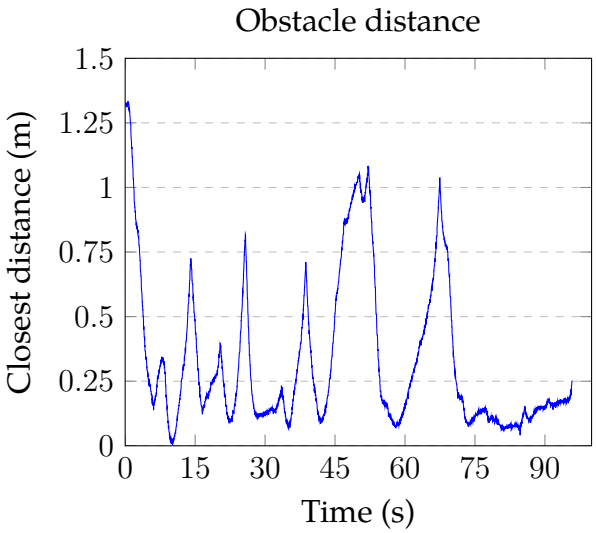


(b) The velocity over time.

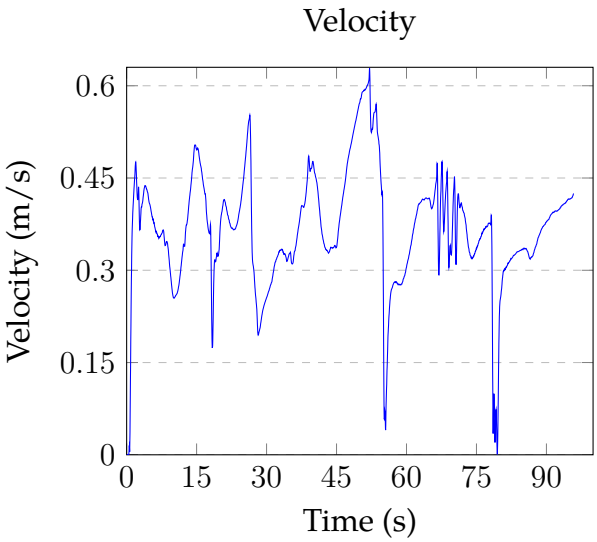
Figure A.92: Results for operator 8 from the user study with configuration 4.

Configuration 5

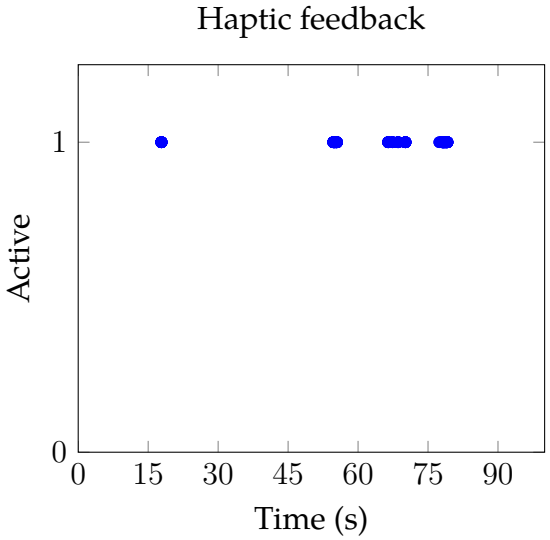
Figure A.93: The path for operator 8 with configuration 5 in the user study. Red is obstacles detected by sensor, blue is the UAV, and green is when haptic feedback is activated.



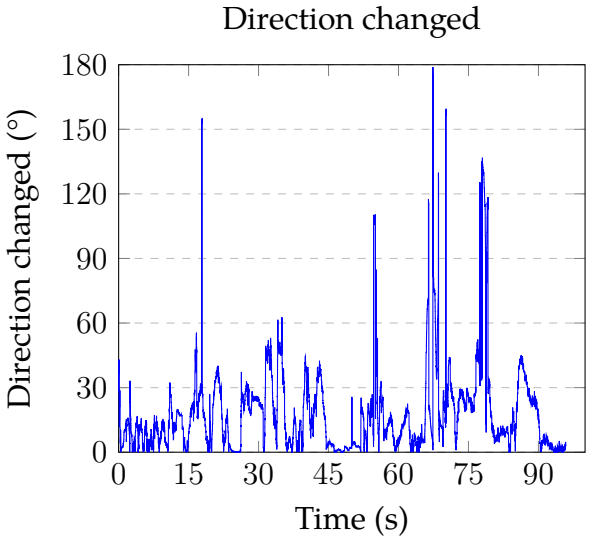
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.



(d) Direction changed over time.

Figure A.94: Results for operator 8 from the user study with configuration 5.

Configuration 6

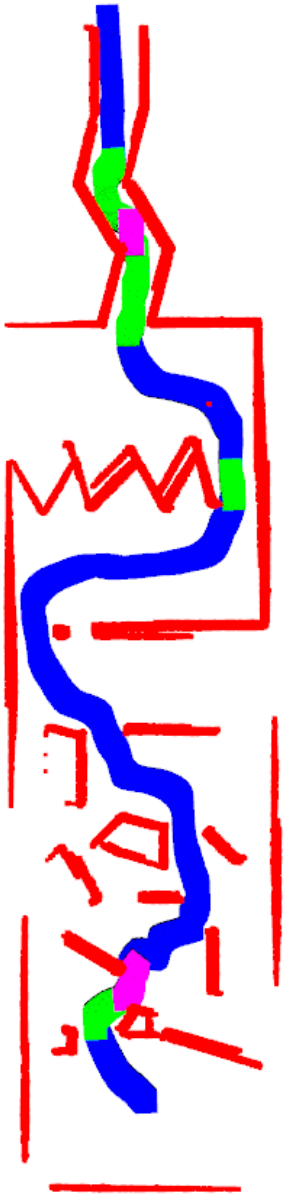
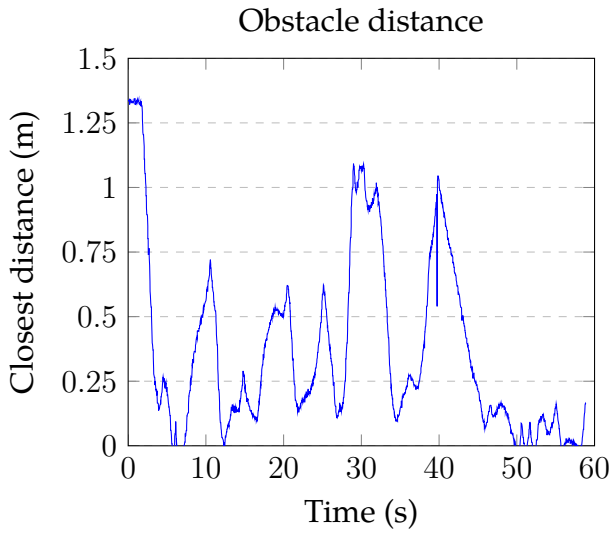
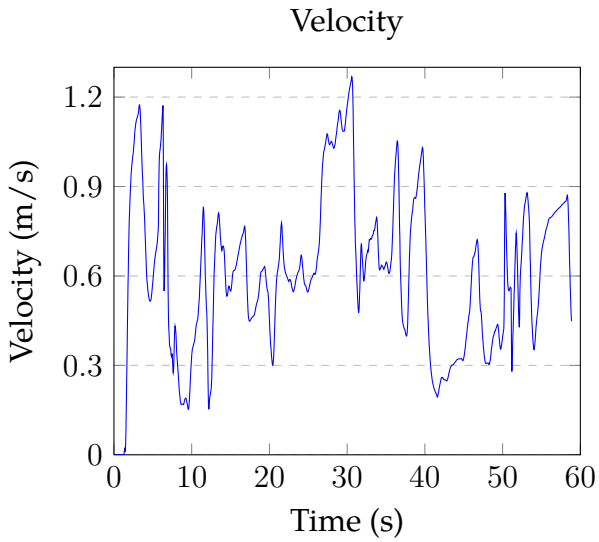


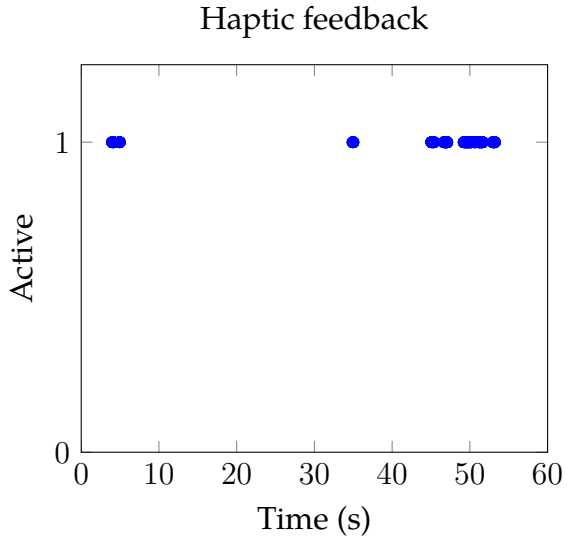
Figure A.95: The path for operator 8 with configuration 6 in the user study. Red is obstacles detected by sensor, blue is the UAV, green is when haptic feedback is activated, and pink indicates collision.



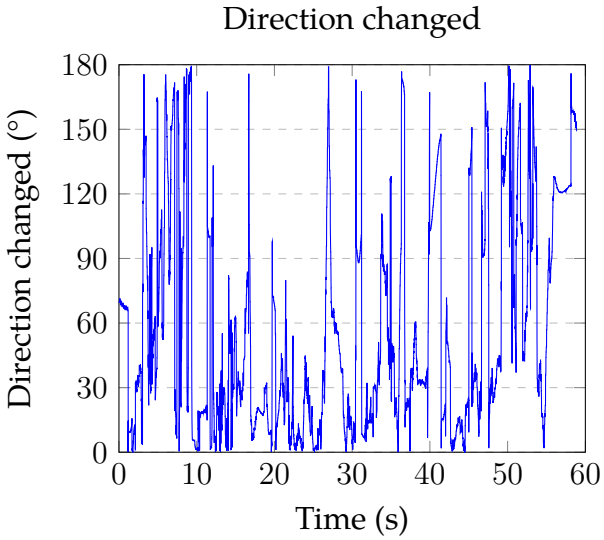
(a) The closest obstacle distance from sensor measurements over time.



(b) The velocity over time.



(c) Haptic Feedback over time. A value of 1 means that haptic feedback is present, 0 that it is not present.



(d) Direction changed over time.

Figure A.96: Results for operator 8 from the user study with configuration 6.

Appendix B

Social Aspects

B.1 Sustainability

UAVs can have a positive effect with regard to environmental sustainability. They can be used for renewable energy maintenance, such as wind turbines and solar panels, where they monitor for damages. UAVs can also be monitoring nature, to hinder poachers, and do aerial mapping of rain forests. Environmental disasters can be prevented by inspected an affected area and lead people away from the danger. Large areas such as farms and agricultural land can be monitored by UAVs instead of ground vehicles, reducing emission.

Replacing humans with robots in a factory can be economically sustainable since robots does not need wages.

B.2 Ethics

As mentioned in Section 2.2 robots were used in a search and rescue mission at the World Trade Center. This is a good example of a dangerous situation for people that can be performed by a robot instead. The robots were tele-operated and the lack of feedback to the operator (only a video feed) led to the robots getting stuck. By making the robots more autonomous, like having automatic collision avoidance, and providing more feedback to the operator the situation could have been different and more people could have been found.

There are also ethical issues regarding works connected to autonomous systems, like this project. Autonomous UAVs could be used for surveil-

lance, espionage and in warfare.

B.3 Social Impact

There are a number of tedious tasks that are well suited to be done by UAVs, such as monitoring inventory, supply and progress at construction sites or mapping the interior of a building. Using UAVs for such tasks allows people to instead spend time on more important things. Replacing people for these tedious tasks can however lead to less job opportunities.