Faculty of Aerospace Engineering

Autonomous Collision Avoidance for Swarms of MAVs

Based solely on RSSI measurements

T. Szabó

August 21, 2015

TU Delft Delft University of Technology

Challenge the future

Autonomous Collision Avoidance for Swarms of MAVs

Based solely on RSSI measurements

Master of Science Thesis

For obtaining the degree of Master of Science in Aerospace Engineering
at Delft University of Technology

T. Szabó

August 21, 2015

Faculty of Aerospace Engineering · Delft University of Technology



Delft University of Technology

Copyright © T. Szabó
All rights reserved.

Delft University Of Technology
Department Of
Control and Simulation

The undersigned hereby certify that they have read and recommend to the Faculty of Aerospace Engineering for acceptance a thesis entitled **"Autonomous Collision Avoidance for Swarms of MAVs"** by **T. Szabó** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated: August 21, 2015

Readers:

Prof. Dr. Ir. J.A. Mulder

Dr. G.C.H.E. de Croon

Dr. Ir. C.C. de Visser

Ir. K.Y.W. Scheper

Dr. Ir. C.J.M. Verhoeven

Abstract

Swarming is a promising solution for extending the flight time and payload carrying capabilities of Micro Aerial Vehicles (MAVs), where recent years have brought many advancements. These allow MAVs to operate ever more autonomously by tackling problems such as obstacle avoidance and autonomous navigation. A major challenge that still remains, however, is to ensure collision avoidance within the swarm itself. Avoiding collisions with other members of the swarm requires knowledge of their relative positions - typically requiring additional sensors to be carried on-board. Using the signal strength of the MAVs' communication link provides an alternative method for estimating relative distances between the members of the swarm without requiring need for any additional sensors.

Signal strength based distance estimation has challenges of its own, since the measurements may be severely impacted by signal noise and interference. Additionally, non-ideal transmitter/receiver antennas introduce significant biases for the obtained distance estimates. The design of an effective, non-trivial collision avoidance strategy based on such estimates is a challenging task.

Evolutionary Robotics is a promising framework for the automatic development of successful behaviors based on limited information. The results obtained with evolution often result in the exploitation of sensory-motor coordination for deriving additional knowledge from the limited information that is available. Considering the limitations of signal strength based distance estimation - both in terms of quantity and quality, the evolutionary robotics framework is a promising tool for developing successful intra-swarm collision avoidance behaviors based solely on signal strength measurements.

The research question posed in this thesis can be formulated as follows:

How can a swarm of MAVs achieve intra-swarm collision avoidance based solely on Received Signal Strength Indicator (RSSI) measurements, with minimal impact on the MAV's primary task performance?

To answer the question above, two collision avoidance behaviors are developed within this thesis - one designed manually, based on logic and human reasoning, while the other one is the result of an evolutionary optimization. Simulation experiments of 300 seconds show

that, three MAVs in a square room with varying size, using either of the proposed avoidance behaviors extend the flight time by more than 100% when compared to not having active collision avoidance within the swarm. The behavior obtained with evolution has a probability of resolving a conflict in the range of $88 - 96\%$ - successfully resolving 15-22 conflicts in a simulation trial. Results of the manually designed avoidance behavior have a success rate in the range of $78 - 94\%$, resolving between 7 and 16 conflicts in each trial. Both of these strategies have, however, characteristic failure modes, when the avoidance behavior is typically unable to resolve a conflict. In terms of task performance, resolving a conflict with the evolved avoidance behavior takes longer time. This, in turn, results in the MAVs using their task behavior for somewhat shorter periods of time. On the other hand, the avoidance behavior of the evolved algorithm has a more optimal exit angle, resulting is larger area coverage on smaller areas.

Real-world experiments, performed with three ARDrones in a flight space of 6×6 meters show good correspondence to the simulation results in terms of the algorithm's success rate, resolving, on average, 13 and 14 conflicts per trial, resulting in a success rate of 86% and 88% for the manual and evolved avoidance behaviors respectively. The other considered performance metrics, however, show significant differences compared to the simulation results, where the average trial times were 37.46 and 38 seconds for the avoidance behaviors. This is due to the increased number of conflicts the ARDrones encountered, indicating a significant reality gap between the simulation environment and reality.

The results of this thesis indicate, that a collision avoidance algorithm based on distance estimates obtained solely from RSSI measurements can not guarantee collision avoidance. In reality, estimates show large variations and periodic biases which severely degrade the performance of the avoidance behaviors. Nevertheless, the algorithms proposed in this thesis can resolve a number of conflicts based on such unreliable measurements. Therefore, RSSI based collision avoidance can be an attractive alternative for small groups of MAVs with stringent payload limitations, operating in a large environment.

Acronyms

AI	Artificial Intelligence
ANN	Artificial Neural Network
BT	Behavior Tree
DUT	Delft University of Technology
EA	Evolutionary Algorithm
EKF	Extended Kalman Filter
ER	Evolutionary Robotics
FSL	Free Space Loss
FSM	Finite State Machine
GA	Genetic Algorithm
GCC	Generalized Cross-Correlation
GCS	Ground Control Station
GP	Genetic Programming
GPS	Global Positioning System
I/O	Input/Output
IMU	Inertial Measurement Unit
IR	Infrared
KF	Kalman Filter
LIDAR	LIght Detection And Ranging
LOS	Line Of Sight
MAV	Micro Aerial Vehicle
MOGA	Multi-Objective Genetic Algorithm
NED	North, East, Down
NSGA-II	Non-dominated Sorting Genetic Algorithm II
PDF	Probability Density Function
RADAR	RAdio Detection And Ranging
RF	Radio Frequency

RSS	Received Signal Strength
RSSI	Received Signal Strength Indicator
SDK	Software Development Kit
SLAM	Simultaneous Localization and Mapping
SPEA	Strength Pareto Evolutionary Algorithm
SRP	Steered Response Power
TDOA	Time Difference Of Arrival
TOA	Time Of Arrival
TOF	Time Of Flight
UKF	Unscented Kalman Filter
VOR	Very high frequency Omni-directional Range

Contents

Acronyms **vii**

1 Introduction **1**

2 Scientific Paper **5**

3 Literature Review **21**
3-1 Estimating Distance from Signal Strength 21
3-2 Evolutionary Robotics 23
3-2-1 Genetic Algorithms 24
3-2-2 Multi-Objective Genetic Algorithms 26
3-3 Artificial Intelligence Framework 26

4 Experimental Setup **29**
4-1 The Flight Platform 29
4-1-1 The Parrot ARDrone2 & Paparazzi UAV 30
4-1-2 The BLED112 Bluetooth USB dongle 30
4-1-3 The CyberZoo 30
4-2 Simulation Models 31
4-2-1 The MAV Dynamics Model 32
4-2-2 The RSSI Sensor Model 33
4-2-3 The Virtual Flight Arena 34
4-3 Controller Structure 35
4-3-1 Distance Estimation Filters 36
4-3-2 Incorporating the Avoidance Behavior 37

5 Evolving an Avoidance Behavior **39**
5-1 The NSGA-II Algorithm . . . 39
5-2 Genetic Operators for Behavior Trees . . . 40
5-3 Cost Functions . . . 43
5-4 Evolution Parameters . . . 44
5-5 Evolution Results . . . 45

6 Collision Avoidance Strategy **49**
6-1 A Collision Avoidance Strategy Based On Closure Rate . . . 49
6-2 Manually Designed Avoidance Behavior . . . 52
6-3 Evolved Avoidance Behavior . . . 54

7 Results **57**
7-1 RSSI Based Distance Estimation . . . 57
7-2 Performance of the Avoidance Behavior . . . 62
7-2-1 Performance Metrics . . . 62
7-2-2 Simulation Results . . . 63
7-2-3 Real World Results . . . 66

8 Conclusions **71**

9 Recommendations **73**

A Literature Review on Relative Localization **75**
A-1 Time of Arrival . . . 76
A-2 Time Delay of Arrival . . . 77
A-3 Received Signal Strength . . . 78
A-4 Direct Bearing Measurements . . . 79
A-5 Comparison of Methods . . . 79

B The Behavior Tree Implementation **81**

Bibliography **85**

List of Figures

4-1 Implemented wall avoidance behavior for the MAVs 31
4-2 Commanded, measured and modeled flight velocities of the ARDrone2 32
4-3 The RSSI model of the BLED112 Bluetooth USB dongle. 34
4-4 Signal Strength as a function of distance 35
4-5 Illustration of the antenna gain effects on the distance estimates 37

5-1 Illustration showing the definition of the crowding distance 40
5-2 Illustration of the boundary of the common region between two BTs 42
5-3 Progress of the different cross functions during evolution 45
5-4 The Pareto front after the evolution 47

6-1 An avoidance behavior based on high level information, guaranteeing collision avoidance between two MAVs 52
6-2 Illustration of a worst case conflict with multiple MAVs 53
6-3 The manually developed avoidance behavior 53
6-4 Examples of simulation trials with the manually designed avoidance behavior . . 54
6-5 The evolved avoidance behavior 55
6-6 Examples of simulation trials with the evolved avoidance behavior 55

7-1 RSSI measurements and off-line model predictions with two ARDrones 58
7-2 True, on-board and off-board distance estimates with two ARDrones 60
7-3 RSSI measurements and off-line model predictions with three ARDrones 61
7-4 Results of the simulation experiments 65

7-5 Example trajectories from the real-world experiments 68
7-6 Distance estimates corresponding to the trajectories indicated on Figure 7-5. . . 69

Chapter 1

Introduction

In recent years the miniaturization of electronics has allowed the development of Micro Aerial Vehicles (MAVs), weighing 100 grams or less. (K. Ma, P. Chirarattananon, S. Fuller, 2013; Wagter, Tijmons, Remes, & Croon, 2014; Kushleyev, Mellinger, Powers, & Kumar, 2013) Due to their small size and weight, these vehicles are suitable to be used in narrow, indoor spaces or close to humans without endangering people or property. Another consequence of their size is that they have limited flight time, stringent payload limitations and tight computational constraints - features that make MAVs unattractive to be used on their own. In order to exploit the benefits of MAVs and comply with their limitations, their use in swarms is envisaged (Dorigo et al., 2004).

Swarm Robotics is defined as *"the study of how large numbers of relatively simple physically embodied agents can be designed such that a desired collective behavior emerges from the local interactions among agents and between the agents and the environment."* (Sahin, 2005)

The use of swarms has been proposed to numerous fields of interest, like search and rescue missions (Basiri, Schill, Lima, & Floreano, 2012), target localization and tracking (Uzol, Yavrucuk, & Sezer-Uzol, 2008), communication relay systems (Hauert, Winkler, Zufferey, & Floreano, 2008), navigation and exploration (Stirling, Roberts, Zufferey, & Floreano, 2012) or even commercial crop pollination (Dantu, Kate, Waterman, Bailis, & Welsh, 2011). The main benefits of using swarms over a single, more complex entity are the redundancy, flexibility and scalability that a swarm provides (Sahin, 2005). The main challenges of swarming, especially for small sized MAVs are navigation, obstacle avoidance and action coordination. Some of these challenges, namely navigation (Forster, Pizzoli, & Scaramuzza, 2014) and the avoidance of obstacles in the environment (Wagter et al., 2014) have already been studied. Considering the limitations of small MAVs, intra-swarm collision avoidance remains an important question to be answered.

Studies considering collision avoidance for MAV swarms are typically based on the relative distance and bearing estimates to the nearby MAVs. In environments where Global Positioning System (GPS) is available, such measurements can easily be obtained if the MAVs communicate their absolute position to their neighbors. In indoor environments, where the

benefits of MAVs can truly be exploited, GPS is not available. Because of this, the absolute position of the MAVs can only be obtained from either a motion tracking system (Kushleyev et al., 2013) or by having a centralized map of the area with a common reference frame that all MAVs have access to (Dunkley, Engel, & Cremers, 2014). The use of a motion tracking system is only applicable in limited and, generally, known environments, while having a common map of the area is out of reach for current, small sized MAVs due to the computational and communications requirements that this would impose.

In GPS denied environments, the relative position of MAVs with respect to each other has to be measured on-board. Typically, in order to measure the distance and bearing to near-by MAVs additional sensors such as an on-board camera (Cutler, Michini, & How, 2013; Faigl, Krajnik, & Chudoba, 2013), Infrared (IR) sensors (Roberts, Stirling, Zufferey, & Floreano, 2012) or an array or microphones (Basiri & Schill, 2013) are used. While these approaches provide both relative distance and bearing to the neighboring MAVs, they all rely on specific sensors and typically require the processing large amounts of data that does not scale down well for small sized MAVs. Appendix A offers a more extensive review of the available relative localization techniques.

Typically MAVs have a wireless connection used for sending data to a ground station and/or receiving commands from there. Furthermore, these communication devices typically measure the Received Signal Strength Indicator (RSSI) of this wireless connection in order to determine the reliability and quality of the communication link. Such RSSI measurements can also be used to estimate the relative distance between the transmitter and receiver antennas, hence, the distance between two MAVs, *without the need of any additional sensors* to be carried on-board.

Although distances can be estimated from RSSI measurements, information about the relative bearing of the MAVs, required by most of the action coordination strategies available today (Brambilla, Ferrante, Birattari, & Dorigo, 2013), can not be derived from a single signal strength measurement. A couple of exceptions using RSSI for coordinating swarm behavior exist, however, they either rely on additional sensors for collision avoidance (Ludwig & Gini, 2006; Ugur, Turgut, & Sahin, 2007) or use a predetermined height separation (Hauert, Zufferey, & Floreano, 2008).

Avoiding collisions based only on distance estimates with limited accuracy, without knowing from what direction the MAVs are approaching from, poses a complex problem to be solved. Evolutionary Robotics (ER) is a method intended to optimize the controller and/or body form of a robot by making use of Evolutionary Algorithms (EAs) (Nolfi & Floreano, 2000; Bongard, 2013). Inspired by natural evolution, the solutions provided by EAs often result in less conventional methods, exploiting sensory-motor coordination for achieving the task at hand. This often result in better solutions for complex problems that are difficult to tackle for human a designer. Therefore, in this thesis the ER framework is used to develop, or in other words *evolve*, a suitable collision avoidance behavior.

The goal of this thesis can be formulated as the following research question:

How can a swarm of MAVs achieve intra-swarm collision avoidance based solely on RSSI measurements, with minimal impact on the MAV's primary task performance?

The above specified research question can be answered by answering the following sub-

questions:

1. How can a swarm of Micro Aerial Vehicles achieve successful intra-swarm collision avoidance by solely relying on Received Signal Strength Indicator measurements?
   1.1. How can distance be estimated from RSSI measurements?
   1.2. Given the distance to another MAV, how can a potential collision be avoided in a way that has minimal impact on the performance of the MAV's primary task?
2. How can the performance of the collision avoidance behavior be improved by using the Evolutionary Robotics framework?
   2.1. What are the appropriate performance metrics to be used by the EA?
   2.2. Which are the appropriate parameters for the algorithm to convergence?
   2.3. How does the resulting behavior compare to a manually designed one both in simulations and in reality?

The answers to these questions are summarized in the form of a scientific paper in Chapter 2 and discussed in more detail in the remainder of this thesis. Chapter 3 presents a review of recent scientific literature that served as a basis for this thesis, while in Chapter 4 the experimental setup, including the real world setup and the simulation environment constructed based on it, is discussed. In Chapter 5, the EA implementation, including the genetic operators, implemented cost functions and algorithm parameters are detailed, as well as a discussion on the evolutionary results in terms of the EA's progress and convergence.

In Chapter 6 two avoidance behaviors are presented - one of them having been designed manually, while the other one is the behavior resulting from the evolutionary process. Their performance, in terms of collision avoidance and task behavior, are analyzed in Chapter 7. Finally, in Chapters 8 and 9, the conclusions of this thesis and recommendations for future research are presented.

Chapter 2

Scientific Paper

This page is intentionally left blank.

This page is intentionally left blank.

Intra-swarm collision avoidance based solely on distance estimates

T. Szabó, E.J.J. Smeur, K.Y.W Scheper, C. De Wagter, B.D.W. Remes, C.C. de Visser, and G.C.H.E. de Croon[1]

***Abstract*— This article presents two collision avoidance strategies that allow Micro Aerial Vehicles (MAVs) to avoid each other based solely on distance estimates from Received Signal Strength Indicator (RSSI) measurements. The main advantage of these strategies is that they require *no* additional sensors beside the communication equipment already available on-board - an important factor when considering swarms of MAVs with stringent payload limitations. We make use of Evolutionary Robotics in the design of one of the algorithms. Simulation experiments show that in a worst-case scenario, when the MAVs fly at the same height, using either of the avoidance behaviors results in an increase in flight time of over 100% when compared to a scenario without active collision avoidance, in a 6×6m space, while successfully avoiding 78% and 88% of the conflicts with the manually designed and evolved behaviors respectively for MAVs with a diameter of 0.8m. The experimental results obtained with three Parrot ARDrones, equipped with BLED112 Bluetooth USB dongles, confirm the success rate of the avoidance algorithms observed in the simulations albeit severe sensor noise, however, they also show a significant drop in all other metrics considered. This leads to the conclusion that while RSSI based distance estimates can not ensure a successful collision avoidance behavior in urban environments, they could be useful alternatives for small groups of MAVs in large environments.**

***Index Terms*— RSSI, MAV, Swarming, Collision Avoidance, Behavior Tree, Evolutionary Robotics**

I. INTRODUCTION

The miniaturization of electronics has recently led to the construction of Micro Air Vehicles (MAVs), weighing in the order of 80 grams or less (e.g., [1], [2], [3], [4]). Examples range from 73g micro quad rotors studied in [3] to the 80mg Robobee [2]. Such light-weight MAVs are suitable for navigation in narrow spaces and are safe enough to operate close to humans. Of course, the light weight also implies stringent restrictions in terms of the energy, sensors, and processing that can be carried on board.

A single MAV has a limited flight time and payload capability. Therefore, it is envisaged that such MAVs will operate in *swarms* (cf. [5]), so that they can survey or explore large areas efficiently and distribute the relevant payload (such as sensors) over the different swarm members.

Swarming of MAVs represents a major challenge. The individual MAVs should be able to navigate [6], avoid obstacles [4] and coordinate their behavior. An important first step in such coordination is that the MAVs can avoid intra-swarm collisions. This is a difficult problem, since other MAVs can fly in from any 3D-direction.

[1]Micro Air Vehicle Laboratory, Control and Simulation, Faculty of Aerospace Engineering, TU Delft Robotics Institute, Delft University of Technology, The Netherlands. email

Most studies in the field of swarm robotics base their coordinating behavioral algorithms on the local sensing of the relative direction and distance to other swarm members [7], [8]. Outdoors, such a sensor measurement can be simulated by communicating GPS-coordinates [9]. Indoors, GPS is not available, so that the communication of coordinates would either require the use of a motion tracking system [3] or the construction of a centralized map with a common reference frame. While the first strategy is only applicable to rather limited, known environments, the second strategy is currently out of reach for smaller MAVs (cf. [10]).

For coordination and collision avoidance, it would be valuable to directly measure the 3D-direction and distances to other drones in the swarm. Previously, active strategies have been proposed in which MAVs used infrared leds and multiple IR sensors [11], or actively emitted sound chirps with multiple microphones [12]. Such strategies do provide both a relative bearing and distance, but do not fit well with the size and energy restrictions of smaller MAVs.

An alternative sensing scheme that does have the potential to scale down to tiny drones is to use the wireless connection itself to coordinate swarm behavior. For instance, in [13] the number of MAVs in communication range is used to aggregate a swarm of robots, even if they move through an obstacle field. In [14], swarming algorithms were evolved in which the number of MAVs in communication range was combined with the hops from the MAV to a base and user station. The resulting local behaviors led to successful search for a user station with an unknown location. There have also been studies focusing on swarm dispersion in order to create a large but connected wireless network [15], [16]. In one of these studies, only the distance measurement from the wireless connection was used to achieve a large and still connected wireless network.

While the studies mentioned above have achieved good results on higher level tasks, none of them has addressed the use of the wireless connection for the avoidance of intra-swarm collisions. They either assume additional sensors for collision avoidance [15], [16] or a predetermined height separation [14].

In this article, we make use of Evolutionary Robotics (ER)[17], [18] to develop an intra-swarm collision avoidance strategy that relies only on the strength of a wireless connection. The main advantage of this strategy is that it scales to small MAVs, which are typically already equipped with small RF transceivers [19]. In addition, it is suitable for indoor scenarios in which the MAVs do not have a shared global reference frame.

The remainder of the article is organized as follows. First, in Section II, the process of estimating distances from

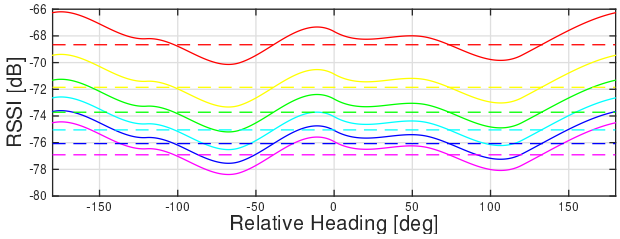


Fig. 1. Radiation pattern measured for the BLED112 USB Dongle. The solid lines show the RSSI given by Equation 6. The dashed lines show the RSSI values with the assumption of an omnidirectional antenna. The different colors red to magenta correspond to transmitter-receiver distances of 0.5 to 3 meters in steps of 0.5m respectively

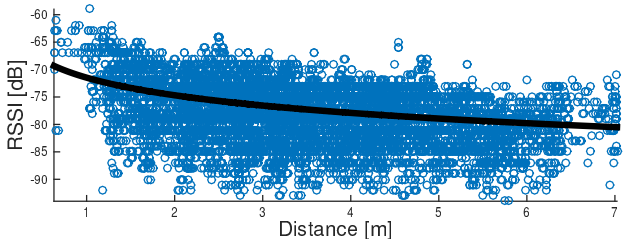


Fig. 2. In-flight RSSI measurements vs the distance with the BLED122 USB Dongle used for modeling the signal strength-distance relationship. The data points indicate the measured signal strength, while the black line represents the data fit of Equation 6 with antenna gain terms set to 0 dB. The standard deviation of the measurements is estimated to be equal to 3.2 when taking the antenna gains into account as well.

RSSI measurements is explained. Section IV discusses the evolutionary setup and the desired structure of the resulting controller, while in Section V-A the two avoidance strategies proposed in this article are presented. Subsequently, Sections VI-A and VI-B present our experimental results from simulations and from flying MAVs respectively. Finally, our conclusions regarding the proposed strategies are detailed in Section VII.

II. ESTIMATING DISTANCES FROM RSSI

To estimate the distance between a broadcasting and a receiving antenna, we must first develop a model for the signal propagation. Here, we adapt the derivation from [20, Ch. 3] by J. Seybold. The absorption of RF signals when traversing a medium can be modeled using a Free Space Propagation model. This is based on the inverse-square law, where the power flow per unit area (w) of a signal broadcast with power P_t using an antenna with gain G_t, at distance d is defined as

$$w = \frac{P_t G_t(\Psi_{tr})}{4\pi d^2} \tag{1}$$

where the antenna gain is not assumed isotropic and is a function of the relative angle Ψ_{tr} from the transmitter to the receiver. Given λ, the carrier signal's wavelength and A_e, the antenna's effective area or aperture, the power available at the receiving device can be determined as:

$$P_r = \frac{P_t G_t(\Psi_{tr})}{4\pi d^2} A_e(\Psi_{rt}) = \left(\frac{\lambda}{4\pi d}\right)^2 P_t G_t(\Psi_{tr}) G_r(\Psi_{rt}) \tag{2}$$

where angle Ψ with subscript rt is the relative angle from receiver to transmitter, and

$$G_r(\Psi_{rt}) = \frac{4\pi A_e(\Psi_{rt})}{\lambda^2} \tag{3}$$

These equations are based on idealized free-space models. In cluttered environments the path loss exponent may differ, therefore it is replaced by the variable n. Define the loss of the signal over the traversed distance as $L = \frac{P_r}{P_t}$. This results in the Friis free-space loss equation given as:

$$L = \left(\frac{\lambda}{4\pi d}\right)^n G_t(\Psi_{tr}) G_r(\Psi_{rt}) \tag{4}$$

Which can be expressed in log form as

$$L = 10nlog\left(\frac{\lambda}{4\pi d}\right) + G_{t_{dB}}(\Psi_{tr}) + G_{r_{dB}}(\Psi_{rt}) \tag{5}$$

We can add the Received Signal Strength Indicator (RSSI) as measured, e.g., with typical Bluetooth sensors. The free space loss is then replaced with the measured RSSI, using a reference measurement L_0 at 1m.

$$RSSI = L_0 - 10nlog(d) + G_{t_{dB}}(\Psi_{tr}) + G_{r_{dB}}(\Psi_{rt}) \tag{6}$$

In this paper we utilize the BLED112 USB Bluetooth Dongle, which has a transmit power of $+0dBm$ and a receiver sensitivity of $-93dBm$ [21]. The radiation pattern of most antennas is not uniform and should be identified.

Multiple tests were performed using two Parrot AR drones equipped with BLED112 Bluetooth modules flying around a flight hall equipped with a motion capture system. The recorded signal strength and ground-truth distances and directions were used to determine the parameters of Eq. 6.

Figure 1 shows the RSSI variations as a function of relative heading θ. It shows that it has $\sim$ 4dB 'lobes'. A consequence of this, turning one MAV results in the changing of the observed RSSI value even if the distance between the transmitter and receiver remains constant.

Figure 2 shows the noise present in the RSSI readings for different distances, where the circles indicate the measured signal strength at different distances and relative angles. The black line indicates the model fit of Equation 6, where the antenna gain terms, $G_{t_{dB}}(\Psi_{tr})$ and $G_{r_{dB}}(\Psi_{rt})$, are assumed to be zero. Such a model corresponds to the information available at the receiver for estimating the distance d between the transmitter and receiver. The noise at a given distance is in the range of $[-15, +10]dB$. This raw data must first be filtered before it can be used by the control system.

For estimating the distances between the MAVs, an Extended Kalman Filter (EKF) was implemented, of which the only state is the estimated inter-MAV distance. Since the motion of the MAVs and the variation of the inter-MAV distance are unknown, the state prediction step of the filter assumes the inter-MAV distance to remain constant. The filter's output prediction corresponds to Equation 6 with the antenna gain terms assumed to be 0 dB. While this assumption is necessary, as there is no information available about the relative positions of the MAVs, it has significant consequences on the resulting distance estimates. Assuming the antenna gain terms to be zero, together with the signal's variations due to sensor noise, ground/obstacle reflections, RF interference, etc. results in an inaccurate estimate that

either over- or under-estimates the inter-MAV distances, depending on the MAVs' relative position.

In order to limit these effects, furthermore to ensure that the distances are more likely to be under- rather than over- estimated, resulting in false evasive maneuvers rather than collisions, a maximum-value prefilter is applied to the measured RSSI values before the Kalman filtering. This prefilter maintains the highest RSSI reading over a period of half a second, limiting the effects of interference at the expense of biasing the estimate towards smaller distances.

III. BEHAVIOR TREES FOR MAV CONTROL

In this article we make use of Behavior Trees (BTs) [22] as a means to encode the avoidance behavior. BTs are rooted tree structures that recursively decompose the task to be performed into smaller sub-tasks.

The BT implementation used in this article makes use of a Blackboard, a common place to store the BT's inputs, outputs and internal parameters that can be accessed by all nodes. In addition to that, four types of BT nodes are implemented, all of which can either succeed or fail. *Select* nodes marked with ('?') *tick*, or in other words trigger, their children until one of them succeeds or all of them fail, returning success or failure respectively. *Sequence* nodes ('$\rightarrow$') represent the reciprocal of a Select node, where they tick their children until either one of them fails or all of them succeed, returning the respective state. *Condition* nodes (green boxes) succeed if the comparison of a given Blackboard variable and a constant value evaluate to true, otherwise they return failure. Set nodes, (red boxes) assign a constant value to one of the Blackboard parameters and always return success. Examples of two BT structures can be seen on Figure 3.

The BTs discussed in this article have a single input, the distance estimated to the closest MAV based on RSSI measurements, d. Their outputs are the commanded forward velocity, V, turn rate, $\dot{\Psi}$ and a heading increment Ψ_A. Of these, the turn rate command is intended for, long smooth turns, while the heading increment is intended as a means for sudden avoidance maneuvers.

IV. EVOLVING AN AVOIDANCE BEHAVIOR

Evolutionary Robotics (ER) employs Genetic Algorithms (GAs), a meta-heuristic search method inspired by natural evolution, to optimize the behavior of robots [17]. It relies on a population of candidate solutions, at first generated randomly, which are evolved (improved) through a number of generations. In each generation the individuals' fitness is evaluated based on predefined cost functions, then a new population is created from the genetic material of the well performing individuals via the *crossover* and *mutation* genetic operators. This cycle is continued until some stop condition is met or N_{gen} generations have passed.

Simple GAs [23, Ch. 2] require the different objectives to be weighed and combined into a single objective function. Multi-objective GAs attempt to identify the Pareto front, the region of the solution space where no one objective can be further improved without degrading the performance of another one. In this article we use the Non-dominated Sorting Genetic Algorithm - II (NSGA-II) [24], an elitist, multi-objective GA that attempts to spread the solutions evenly on the Pareto front.

A. Genetic Operators for Behavior Trees

BTs have been previously used with Evolutionary Algorithms (EAs) via grammatical evolution [25] and GAs [26]. The genetic operators used in this article are inspired by the ones used in [26].

1) Initialization: The N_{Pop} members of the first generation of the GA are initialized using the *grow* method [27], where each composite node of the tree is initialized with its maximum number of children (N_{Child}), and there is a fixed limit on the maximum tree depth (N_{Depth}).

2) Evaluation: In each generation, the population's members need to be evaluated based on the selected cost functions. Within ER this is typically done via simulation runs - though there are examples in literature where the population was evaluated on a real robot [28]. Simulations allow faster than real time evaluation of the members, which is a critical factor in ER, as thousands of simulation runs need to be evaluated until the desired solution is found. The simulation setup and cost functions used in this work are discussed in Sections IV-B and IV-C respectively. Since the performance of an individual is not independent of the initial conditions of a simulation run, in order to estimate a generalized performance of the individuals on the different cost functions, the average of N_{Sim} simulation trials with different initial conditions are used.

3) Selection: A solution is said to *dominate* another if it performs better on at least one objective while not performing worse on any other one. According to this principle, after evaluating all individuals of the population, they are sorted to different *non-domination fronts* based on the number of other individuals that dominate them.

For each domination front, the *crowding distance* of its members are calculated. The crowding distance is defined to be proportional to the perimeter of n-dimensional cuboid in the solution space, constructed around the other members of the front on either side of the respective individual along the different objectives. It serves as an indication of the population's density in a certain region of the solution space.

The candidate solutions for the reproduction step are selected by a version of the Tournament selection [29], where the individual with the best performance is chosen from N_T randomly selected members of the population.

Different individuals of the population are compared with the *Crowded comparison* operator [24]. An individual is said to be better than another one if it either dominates it, or if neither of them dominates the other, has a higher crowding distance, increasing the likelihood of selecting members from less populated regions of the solution space.

4) Crossover: The crossover operator is used to combine the genetic material of two parents from the previous generation and create two children for the current one. In this article we make use of a version of the *uniform crossover*

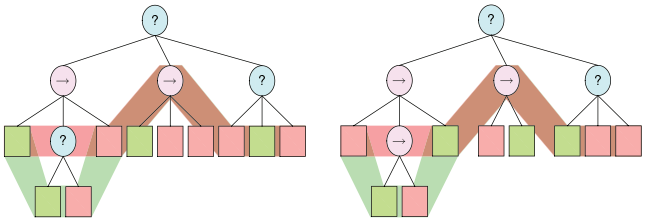


Fig. 3. Two BTs with the boundary of their common region highlighted between them. A node is said to belong to the common region between two trees if: 1. It has the same arity (number of children) in one tree as in the other and 2. If a more restrictive definition is used, has the same type in both trees. All leaf nodes of which the parent belongs to the common region belong to the boundary. Here, the green region indicates the less stringent definition, only considering point 1, and the red region indicates the more restrictive definition of the common region.

[30], which exchanges complete branches of the BT along the boundary of their *common region*.

A given node in a BT is considered to be part of the common region of the two trees if: 1. It has the same number of children as the node in the other tree at the corresponding location; 2. It has the same node type. The latter of these conditions is not enforced in this article, corresponding to a less stringent definition of the common region. All children of a node in the common region either belong to the common region as well, or to the boundary of the common region, along which the nodes are exchanged during crossover. All leaf nodes of which the parents are in the common region, belong to the boundary of the common regions. Figure 3 shows an example of two BTs with the boundary of the common region highlighted between them. The region highlighted in green corresponds to the definition of the boundary region used in this article, considering only the node's child count, while the red region also considers the node types.

The crossover operator used in this article works by first creating copies of the parent BTs as children, determining the boundary of the common region between the two resulting BTs, than exchanging each node/branch in the boundary regions between the two trees with a probability of P_c. This allows the exchange of complete behavior blocks between the two trees resulting in a higher, more directed flow of genetic material from the parents to the children.

5) Mutation: Mutations are used to introduce changes into the genetic material in an attempt to optimize the node parameters for exploiting a local optima, or introduce large changes to the genetic material, allowing the algorithm to explore the solution space. Two kinds of mutations are used in this article. Micro-mutations alter the genetic material of existing nodes, while Macro-mutations, also called *Headless Chicken Crossover* [31], replace nodes by randomly generated tree branches.

Micro-mutations are implemented for leaf nodes by changing the constant parameter of the nodes by a random variation of maximum 10% of the valid parameter range. When such a mutation would results in a parameter outside of its valid range, the direction of mutation is reversed. For composite nodes, one of the following actions is selected randomly: two children are swapped, a node is deleted or a random node is added to the tree. All mutations are applied on a per-node basis, with a probability P_m and P_{HCC} respectively. A micro-mutation, however, is only applied if the node was not macro-mutated.

B. Simulation Setup

The simulated environment is a closed square area with a side length varying uniformly between 6 and 12 meters, varying the rate at which the MAVs come into conflict with each other. The MAVs are assumed to be able to avoid the walls by making a sharp turn to avoid them. This behavior is implemented independently of the avoidance behavior and executed whenever the MAVs approach the environment's boundaries, irrespectively of what the avoidance behavior commands. The MAVs are initialized in the environment at random locations, at least two meters from each other. Their initial heading is set to point towards the centroid of a polygon with its vertices at the initial locations of the MAVs, putting them on an collision course from the start.

The maximum simulation time is set to T_{sim}, however a simulation is stopped at the moment where a collision occurs between two MAVs. A collision is defined as two MAVs closer than their diameter in the horizontal plane. Furthermore, in order to help the EA's convergence, the MAVs' initializations are maintained across a number of generations. These are changed when the number of individual domination fronts reduces to less than 10, or a given set of initializations was used for 10 times.

The real-world experiments presented in this article are performed with Parrot ARDrones, running the Paparazzi open-source autopilot [32]. In order to limit the reality gap [26] between the evolved algorithm's performance, the simulated MAV's dynamics is modeled based on flight tests performed with the ARDrones. The MAVs themselves are modeled as flying spheres with linear, decoupled 2nd order velocity dynamics ($\omega = 2.74$, $\zeta = 0.56$) in the world coordinates, which is a good approximation of the MAV's flight dynamics under normal conditions. The state-space representation of this model is given in Equation 7 along the *X* world axis - an identical model is used along the *Y* axis as well. A drawback of modeling the MAVs in such a way is that the axis coupling during aggressive maneuvers is not properly captured by the model. The MAVs are commanded to fly at the same height in order to investigate a worst case scenario, in which many conflicts occur.

$$\begin{bmatrix} \dddot{X} \\ \ddot{X} \\ \dot{X} \end{bmatrix} = \begin{bmatrix} -2\zeta\omega & -\omega^2 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \ddot{X} \\ \dot{X} \\ X \end{bmatrix} + \begin{bmatrix} \omega^2 \\ 0 \\ 0 \end{bmatrix} V_{X_{cmd}} \quad (7)$$

The flight direction of the MAVs is restricted to the forward direction. Therefore, the inputs of the dynamics are defined as the commanded forward velocity and the commanded MAV heading, whereas the outputs of the avoidance behavior correspond to the commanded forward velocity, turn rate and heading increments. The last two of these are combined by an internal state of the MAV's controller module to form a commanded MAV heading.

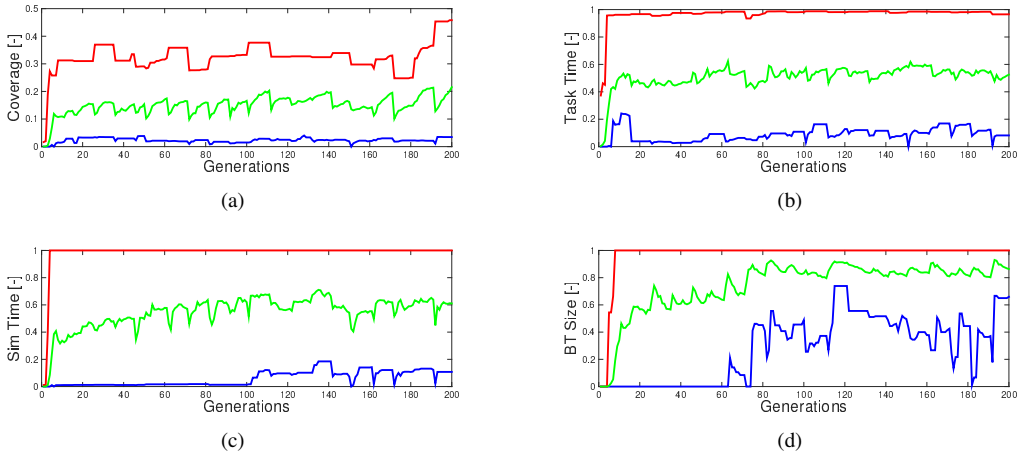


Fig. 4. Progress of the evolutionary process. The red line indicates the best performance measured in a generation, the green line the mean, while the blue line the worst performance. **Top Left**: (Area) Coverage; **Top Right**: Task time; **Bottom Left**: (Task) Simulation Time; **Bottom Right**: Size of the BT

The input of the avoidance behavior, the distance estimate to the closest neighbor, is calculated by filtering simulated RSSI measurements. The simulated sensor readings approximate the real ones as described in Section II. In particular, Eq. 5 is used, where $G_{r_{dB}}(\theta)$ is fitted to the measured data with a 3rd order spline (1st order continuity). Subsequently, normally distributed noise is added $\sim \mathcal{N}(0, 3.2)$ dB.

C. Cost Functions

The primary goal of our work is to provide a means to avoid collisions between drones performing an arbitrary mission. The primary task of the MAVs is defined as exploring an unknown environment by flying straight until the MAV reaches a wall, turning and continuing flying straight.

In order to give some flexibility in performing other, similar missions with the same avoidance behavior, a certain structure for the resulting BT is enforced, where the MAVs have to perform their task with the last branch of the BT that is appended manually to the tree before evaluating the population members. The wall avoidance behavior - though not implemented in the form of a BT branch, precedes in priority all other tasks, which could also be represented as appending an obstacle avoidance branch to the front of the BT. Such a structure allows the resulting behavior to be easily adapted for other tasks.

Based on this setup, the following cost functions are defined for the GA:

1) Area coverage with the last branch of the BT: This is a direct measure of the task performance of the MAVs. The area to be explored is divided into small, 5×5 cm sectors, which are initially marked as uncovered. At each time step of the simulation if an MAV uses its last branch, the location of its centroid gets marked on the map as covered. The value of this cost function corresponds to the ratio of sectors visited and the total number of sectors.

2) Task Time of the MAVs: Our exploration strategy is not ideal in the sense that it often results in visiting previously explored areas. Because of this, if the MAVs enter an already visited region, using the last branch of their BT is not beneficial anymore - which is not necessarily the case if their task is implemented differently. Therefore, this cost function indicates how much the default behavior of the MAVs is used, where its value is the number of times the last branch of the BT is ticked divided by the root node's tick count.

3) Simulation Time: Since a simulation trial is immediately stopped as soon as a collision between two MAVs occurred, maximizing the simulation time implies preventing, or at least delaying an inter-MAV collision. The value of this cost function is defined as the simulation time of a trial divided by the maximum simulation time T_{Sim}.

4) The BT's size: The difficulty for the human designer to understand the encoded behavior in a BT is proportional to the BT's size. Because of this, a small BT that possesses the desired traits of the avoidance behavior is preferred over a larger BT that does a similar job. The value of this cost function is equal to 1 if the BT has less than 10 nodes, equal to 0 if it has 100 or more nodes and varies linearly between them. Note, that the appended branch containing the task behavior of the MAVs is not included when evaluating this cost function.

D. Results of the Evolutionary Optimization

The parameters used for the evolution are given in Table I, while the statistical results of the optimization are shown in Figure 4.

One of the things to notice is that the mean score of the population increases for a while and then drops sharply to a lower value. These drops correspond to the instances when initial conditions of the MAVs are changed, furthermore their magnitude is typically proportional to the number of generations where a set of initial conditions were used, indicating that the behavior has started to converge to a local optimum.

Another important thing to notice is that for all cost functions, with the exception of the area coverage, in the first few generation a (trivial) solution is found that maximizes the respective cost function. These solutions typically maximize only some of the cost functions while they have poor performance on the other ones. Typically solutions maximizing the

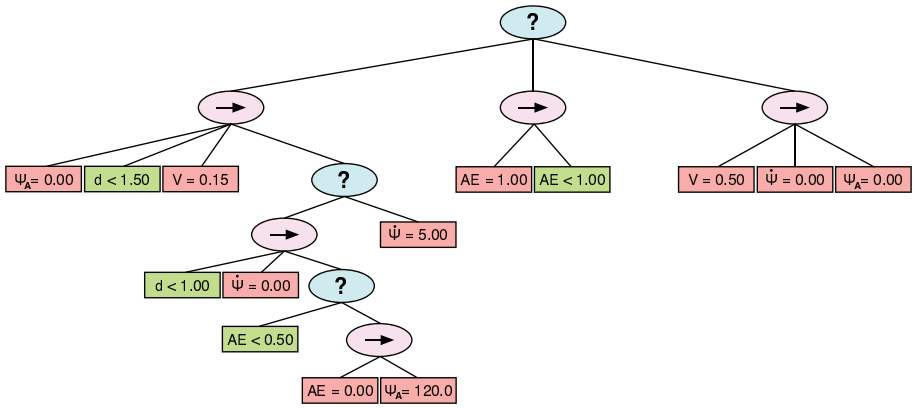


Fig. 5. The BT implementation of our manually designed avoidance strategy. The left branch represents the avoidance behavior, making the MAVs slow down and perform their avoidance turns when the distance between them becomes critical and the turn is enabled. The central branch re-enables the avoidance turn when the distance has grown sufficiently large. The last branch represents the MAVs task behavior.

Task Time or *Sim Time* objectives also maximize the *Tree Size* objective. These solutions correspond to only performing the task without avoiding the other MAVs, maximizing the task time, or simply hovering in place, maximizing the simulation time.

The population's mean performance typically lies around half way between the best and worst performance, except for the *Tree Size* objective. This indicates diversity amongst the solutions, as a solution performing well on one objective typically has worse performance on a different one.

The *Tree Size* objective has a different behavior, as it tends to converge to its maximum value. This is an indication that the form in which the different solutions are encoded is continuously simplified by removing redundant nodes.

Another observation that can be made regarding this objective is that the population's mean seems to settle at a score around 0.8. This is likely due to the crossover operator used in this work, as it has a limited ability to remove unused nodes from the trees that perform well. This limitation is due to exchanging nodes only along the boundary region of the trees, where the operator does not attempt to exchange nodes from the common region.

TABLE I

THE PARAMETERS OF OUR EVOLUTIONARY ALGORITHM

	Parameter	Value	
N_{Gen}	Number of generations	200	[-]
N_{Pop}	Population size	100	[-]
N_{Child}	Maximum number of children	5	[-]
N_{Depth}	Maximum tree depth	5	[-]
N_T	Tournament size	10	[-]
N_{Sim}	Number of simulations	5	[-]
T_{Sim}	Simulation time	300	[s]
P_C	Crossover probability	70	[%]
P_m	Micro-mutation probability	20	[%]
P_{HCC}	Macro-mutation probability	0.1	[%]

V. INTRA-SWARM COLLISION AVOIDANCE STRATEGY

If the RSSI measurements do not depend on the relative orientation of the MAVs, a straightforward strategy suffices to ensure collision avoidance. Successful avoidance can be achieved by simply reversing the flight direction when the distance reaches a threshold ($d \geq T_d$) and the MAVs are getting closer to each other ($\dot{d} < 0$).

However, as Figure 1 shows, the RSSI measurements (and hence distance estimates) vary significantly with the relative positions of the MAVs, further corrupted by signal interference. This implies that no meaningful closure rate $\dot{d}$ can be established, and that strategies such as the one explained above will fail. This leads to a much more challenging problem.

A. Human Designed Avoidance Behavior

Here we present a simple collision avoidance behavior, of which the BT representation is shown in Figure 5. As discussed in Section IV-B, the sole input of the BT is the distance d to the closest neighbor, while its outputs are the commanded velocity in the direction of the MAVs current heading (V), the one-time heading setpoint change (Ψ_A, relative to the current heading) and a commanded turn rate ($\dot{\Psi}$). The parameter AE is used as an internal flag limiting the instances when the MAV is allowed to perform avoidance maneuvers, preventing 'chattering' behavior.

The proposed strategy can then be explained as follows. If another MAV gets closer than 1.50m, the MAV slows down and starts turning smoothly with $\dot{\psi} = 5°$/s. If the other MAV gets closer than 1.00m, the MAV will stop turning smoothly and will make a more sudden, large turn of $120°$. The MAV will fly straight in the new direction. The variable AE prevents the MAV from continuously making such large turns while still being close to the other MAV. If the other MAV is further away again, the MAV will first start turning smoothly (for $d \in [1.00, 1.50\rangle$) and then go straight forward again, while restoring the internal AE variable ($d \geq 1.50$m).

B. Evolved Avoidance Behavior

The evolutionary process resulted in a number of solutions that optimize only one of the performance metrics. The behavior presented in Figure 6 was selected as the best trade-off between MAVs performing their task and avoiding

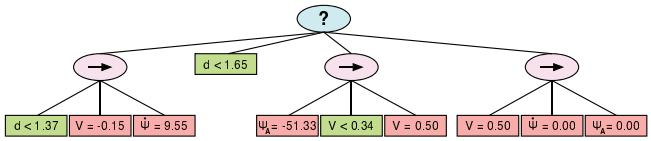


Fig. 6. The resulting strategy of the evolutionary process. The left branch represent the main avoidance behavior where the MAVs fly straight backwards. The central branch the turning behavior that ensures that the consecutive conflicts are different. The last branch represents the MAV's task behavior.

collisions. Initially, there were 29 nodes present in the selected tree, from which 19 nodes were redundant, therefore they were removed manually.

The resulting behavior can be separated into four parts. The first branch of the tree is responsible for reversing the flight direction of the MAVs, while performing a slow right turn, when two MAVs are within 1.37 meters from each other. The second, condition node under the root is responsible for an action hold behavior, which prevents the BT from changing the control command before the MAVs are at least 1.65 meters apart. The third branch of the tree represents a one-time avoidance turn maneuver, performed only if the MAVs have just resolved a conflict, while the last branch represents the default task behavior of the MAVs.

This behavior shows significant resemblance to the manually designed one in the sense that it attempts to put the MAVs on a diverging flight path when a conflict is detected. The main difference between them is that with the evolved behavior the MAVs fly backwards for a while before performing the avoidance turn. This allows the BT to repeat the maneuver if, after the avoidance turn, the resulting MAV headings still result in a collision course.

Both proposed avoidance behaviors rely on an action hold behavior, where with the manual behavior it is ensured by the AE parameter, while the evolved behavior achieves this with the second child of its root node.

VI. Results

The performance of our avoidance behaviors is evaluated primarily in simulations. This allows the consideration scenarios that are difficult to achieve in reality due to physical limitations of our real-world setup, such as considering the MAV and area sizes . A number or real world experiments have also been conducted to evaluate the accuracy of our simulation results.

A. Simulation Results

MAVs having a smaller diameter have a higher chance of missing each other, even if the avoidance behavior does not prevent them from getting on a collision course. Furthermore, the arena's size has a direct effect on the "density" of the MAVs, hence on the frequency with which they get into conflict with each other. Because of the influence of these factors, the avoidance behaviors proposed in this article were evaluated for a number of MAV sizes (0.1 - 0.8 m diameters) and a number of arena sizes (6 - 12 m side length). Furthermore, a more realistic scenario was considered in these simulations than what was used during evolution, where the MAVs were initialized with random orientations. All conditions were evaluated 1000 times while measuring the following performance metrics:

1) Mean trial time
2) Success rate of the avoidance algorithm
3) The survival probability of the MAVs
4) The time spent performing the task
5) Coverage of the flight arena.

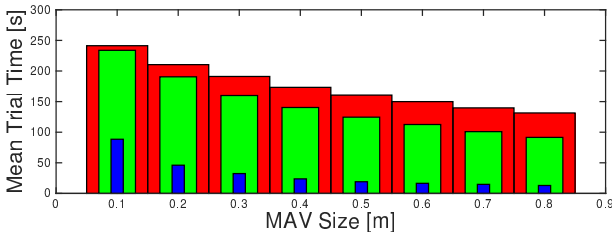
The results of the simulation experiments are shown in Figure 7. The trials presented here concerning the avoidance behavior's success were performed in a 6×6 meter arena with varying MAV sizes, while the trials considering the time during which the MAVs perform their task were performed with 0.8 meter diameter MAVs on varying sized simulation arenas. All simulation times are limited to a maximum of 300 seconds, or until a collision occurs. A number of flight trajectories from different trials are shown in Figure 8.

Figure 7(a) shows the mean time until a collision occurs on 1000 trials. From the figure, it can be seen that by employing either of the avoidance behaviors presented in this article, the expected simulation time is more than doubled when compared to not having an active collision avoidance behavior. Furthermore, the evolved avoidance behavior is more effective than the manually designed one, especially for large MAVs.

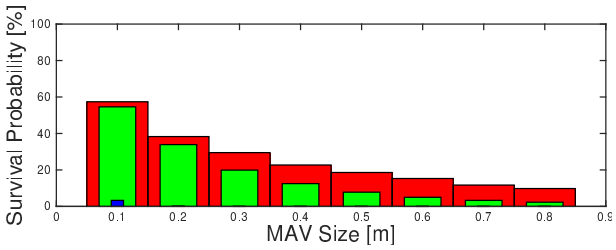
This results is further emphasized by Figure 7(b), showing the probability of ending a trial without a collision. It shows that the chances avoiding all collision during a trial start at about 55% for MAVs of 0.1 meters in diameter using the manually developed avoidance behavior and drop as their size increases to almost zero. For the evolved behavior it starts close to 60% and drop gradually to around 10% for MAVs of 0.8 meters.

Figure 7(c) indicates the likelihood of a successful outcome, given a conflict between two MAVs as a function of their size. The results of this figure indicate that the probability of a conflict being successfully resolved vary between 88.25 and 96.3% as a function of MAV size with the evolved avoidance behavior, while it is between 78.1 and 94.8% if the manual avoidance behavior is used. Without active collision avoidance behavior, these values range from 31% for large MAVs to 85.2% for MAVs of 0.1 meters in diameter.

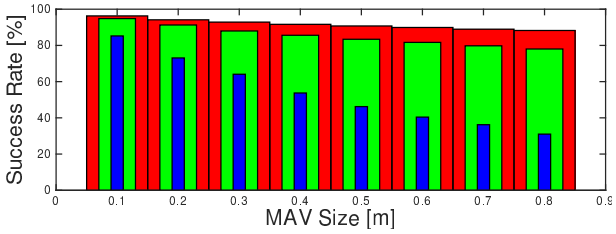
Figure 7(d) shows the average time that the MAVs spend performing their task in flight spaces of different sizes. In these trials the simulated MAVs have a diameter of 0.8 meters. A clear correlation between the time the MAVs spend performing their task and the arena's size can be observed on this figure, which is due to the reduced number of conflicts they encounter with increasing arena sizes. The manual behavior is more efficient in terms of task time for low MAV densities, while the evolved behavior performs slightly better when there are more conflicts. This is due to the manual avoidance behavior resuming the task performance sooner, but the evolved behavior having longer trial times with higher MAV densities.



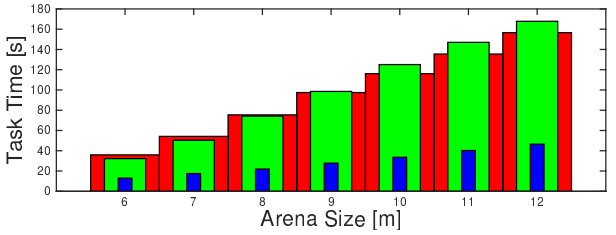
(a) The mean time until a collision occurs. By using either of the avoidance behaviors, this time more than doubles when compared to not actively avoiding each other.



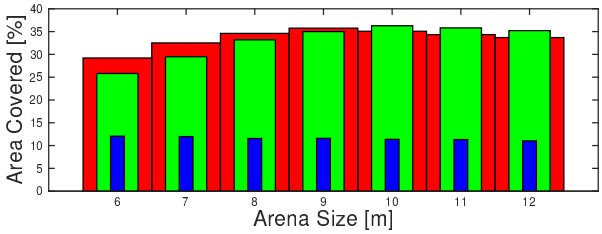
(b) The probability of no MAVs crashing in a given trial. Not avoiding each other has almost zero probability of survival. The evolved behavior shows a significant increase in survival rate when compared to the manually designed one.



(c) The success rate of the avoidance behavior - corresponding to the number of resolved conflicts divided by the total number of encountered conflicts



(d) The average time the MAVs spend perforing their task. The task time of the MAVs is increased when actively avoiding each other and it is increasing with decreasing MAV density. The evolved behavior has a lower rate of increase than the manually developed one.



(e) The average rate of area coverage as a function of area size. Even though the task time of the evolved behavior is lower than that of the manual one, the area coverage rate is higher for smaller areas due to a more optimal avoidance turn angle, Ψ_A.

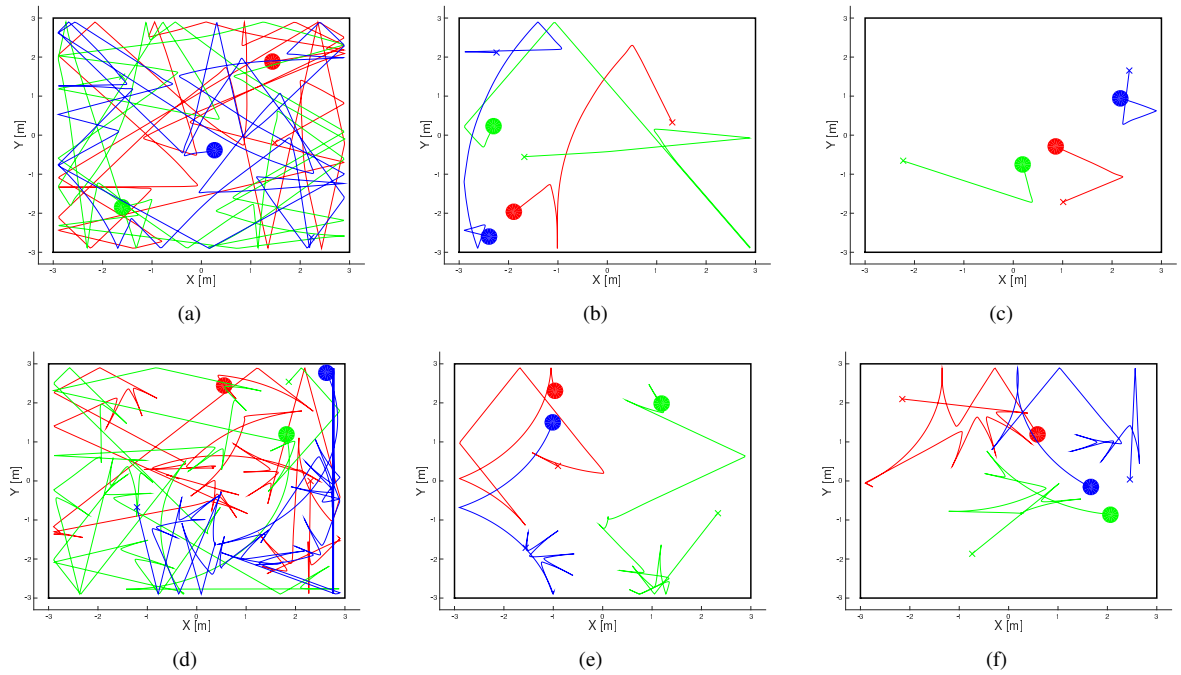
Fig. 7. Results of our simulation experiments evaluated on 1000 runs. The blue bars represent the MAVs with no collision avoidance, the green bars represent our manually designed avoidance behavior, while the red bars indicate the results of the evolved behavior.

Figure 7(e) shows the average area coverage of the MAVs. A similar trend can be noticed on this figure as on Figure 7(d). The evolved behavior has better performance, covering between $\approx 30 - 36\%$ of the area on smaller areas, where more conflicts occur. The manually designed behavior results in an area coverage in the range of $\approx 35 - 36\%$ and it has a better performance on larger areas. The decreasing area coverage as the size of the flight space gets larger is due to the quadratic increase in the number of cells the MAVs need to visit in a fixed trial time.

Figures 8(a) and 8(d) present successful trials with the manual and evolved avoidance behaviors respectively. Here we can see that in terms of area coverage the performance of the two behaviors is similar.

Figures 8(b) and 8(e) present the failure case where the arena's bounds interfere with the avoidance behavior, resulting in a collision. On Figure 8(b) blue MAV can be seen turning away from the southern wall of the arena when it gets into conflict with the red MAV. When they both perform their avoidance turn, the blue MAV is forced to turn again to avoid the western wall, resulting in a collision. Figure 8(e) shows the blue and green MAVs resolving a conflict, at which point the red MAV gets into conflict with the blue one. As they both start their avoidance maneuver, they end up on a parallel path. When the red MAV is forced to turn away from the northern wall, the two MAVs collide.

Figure 8(c) present another failure case of the manually designed behavior, when two MAVs are in the progress of resolving a conflict and a third MAV interferes, which results in a collision. Such a failure mode is not present in the evolved behavior since there is no explicit action blocking present. Multi-way conflicts nevertheless cause difficulties for the evolved behavior as well, as it can be seen on both Figures 8(e) and 8(f), where a collision could be avoided, but the resolution of the multi-way conflict, combined with a wall avoidance maneuver result in situations from which the MAVs can not recover anymore.



(a) (b) (c) (d) (e) (f)

Fig. 8. Flight trajectories from our simulation experiments. In these figures **x** marks the initial positions of the MAVs, while their final position is indicated by the filled circles. **(a)**, **(d)**: a successful trials with the manually developed and evolved behaviors, where all conflicts are resolved. **(b)**: The blue and red MAVs perform their avoidance maneuvers close to the southern wall, when the blue MAV turns away from the western wall, ending up on a collision course with the red MAV. **(c)**: The blue and red MAVs resolve a conflict, but shortly after the red and green MAVs approach each other. The green MAV performs its avoidance turn, but this behavior is still disabled for the red MAV, resulting in a collision. **(e)**: The blue and red MAVs are trying to avoid each other, but as the red MAV turns away from the northern wall, they collide. **(f)**: An example of a multi-way conflict resolution close to the northern wall of the arena, where the consecutive conflicts of the blue MAV with the red one, combined with a wall avoidance, result in a collision between the blue and green MAVs.

TABLE II

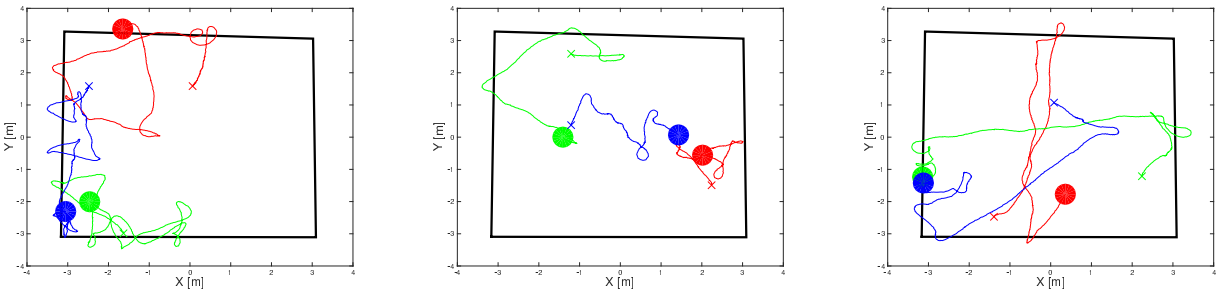
SUMMARY OF RESULTS FOR THE REAL-WORLD EXPERIMENTS

	Trial Time	Success Rate	Task Time	Area Coverage (%)
2 ARDrones - Evolved	92.84 s	83.12 % - 9.85 conflicts	81.89 s (88.20 %)	10.73 %
2 ARDrones - Manual	86.17 s	87.67 % - 12.80 conflicts	71.74 s (83.25 %)	10.98 %
3 ARDrones - Evolved	38.00 s	87.62 % - 13.80 conflicts	24.70 s (65.01 %)	6.04 %
3 ARDrones - Manual	37.46 s	86.16 % - 13.05 conflicts	25.66 s (68.50 %)	6.13 %

B. Results and Discussions of the Real-World Experiments

The real-world experiments presented here were performed using ARDrones, running the Paparazzi open-source autopilot [32]. In the design of the behavior it was assumed that the ARDrones are capable of accurately tracking velocity commands, e.g. by utilizing optical flow. For our experiments, this was simulated through the used of an Optitrack system, which provides real-time feedback to the drones about their position and velocity within the flight arena, based on infrared reflective markers placed on the drones. The measurements of the Optitrack system were forwarded by the ground station to the drones via a Wi-Fi link, where all the computations were performed. The position information of the Optitrack system was solely used to ensure that the ARDrones return to the 6×6 m region in the center of the $10 \times 10 \times 7$ m flight arena, based on the same wall avoidance algorithm as in the simulation experiments.

During these experiments, six experimental runs were performed with two ARDrones (diameter ≈ 0.8 m) and nine runs with three. Since the drones were equipped with indoor hoods, typically the collisions between two drones resulted only in a bump between them, therefore they were allowed to continue flying. In order to maintain our definition of a collision, during the analysis of the results, each trial is considered to end when the distance between the drones in the horizontal plane is reduced to 0.8 meters. In case a collision did not end in a physical crash, the time period until all MAVs were at least 2.5 m apart was discarded and the remainder used as a new experimental trial, corresponding to a new set of initial conditions. The results of the experiments are summarized in Table II with the exception of the survival probability of the MAVs, as no single trial lasted for 300 seconds. Comparing our experimental results with the corresponding simulation results on Figure 7, a significant drop in performance can be observed in all of the metrics with the exception of the success rate, which is very close to that observed in simulations. The similar success rate of the simulation and experimental trials indicates that the performance of the avoidance behavior is comparable to that observed in simulations. During the real experiments, however, more conflicts occurred than in the simulation



(a) The blue and green drones repeatedly avoid each other in a corner, until they collide

(b) The distances between the blue and red drones is severely underestimated, blocking their avoidance maneuvers - cf. Figure 10(b)

(c) Due to measurement noise, the blue MAV performs two avoidance turns close to the arena's center, resulting in a collisions course with the green one - c.f. Figure 10(c)

(d) The red and green drones are stuck together in a similar way as shown in Figure 8(e), resulting in a collision

(e) The red and blue drones collide because their distances are overestimated - cf. Figure 10(e)

(f) All three drones are in a corner repeatedly avoiding each other until the blue drone turns away from a wall and collides with the red one.

Fig. 9. Examples of flight trajectories with three ARDrones. (a) - (c): examples of trajectories with the manual avoidance behavior. The corresponding distance estimates are shown in Figure 10. (d) - (f): examples of trajectories with the evolved behavior. **(a)**: the blue and red drones are stuck in a corner, repeatedly avoiding each other and the arena boundaries, eventually leading to a collision. **(b)**: The distances between the red and blue drones are severely underestimated, blocking the avoidance maneuver from being executed, causing a collision between the red and blue drones. **(c)**: The blue MAV performs two avoidance turns close to the arena's center due to measurement noise, resulting in a collision course with the green one. The green MAV also performs its avoidance turn, but has to turn back due to arena bounds, resulting in a collision. **(d)**: the red and green drones perform their avoidance maneuvers but they get stuck together like on Figure 8(e). **(e)**: the distance estimate between the red and blue drones is severely overestimated, resulting in a collision. **(f)**: all three drones converge to the south-west corner of the flight arena, where they iteratively try to avoid each other. Due to the wall avoidance, the blue drone ends up colliding with the red drone.

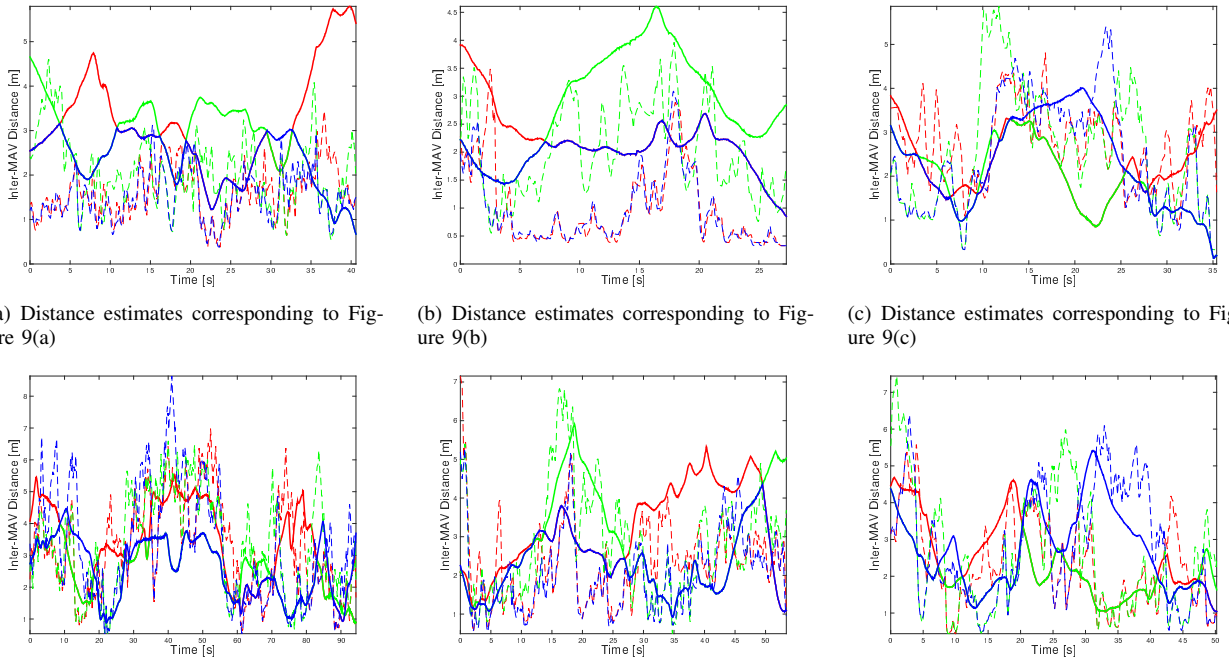
environment, which can be attributed to a significant reality gap.

Examples of the flight trajectories with the manual and evolved avoidance behaviors are shown in Figure 9. Looking at these figures, two things can be noticed. The trajectories of the drones never really follow a straight line, showing characteristic, oscillating lateral motion, especially when the drones fly straight for a short periods. Furthermore, in many cases, whenever the drones perform a sharp turn, a significant lateral displacement can be observed. Both of these are clearly visible on the flight trajectories of the blue and red drones shown in Figure 9(b).

Such a behavior can be explained by the axis coupling of the drones during aggressive maneuvers. The sharp turns made by the drones induce a sway motion, resulting in the turn being performed along a circle, as shown in Figure 9(b), followed by lateral oscillations as the inner-loop controllers cancel the displacement caused by the sway motion. As a consequence of this, the flight trajectories become different from the ones observed in simulations. Another consequence of it is that the variations in the relative positions between the drones aggravate the antenna gain effects, resulting in less reliable distance estimates between them.

Another noteworthy difference between the simulation environment and the behavior of the drones observed during the experiments is due to the aerodynamic effects of the propellers. During the experiments performed, it was observed that whenever the drones approach each other at low speeds, they had a tendency to push each other away. On the other hand, if the distance between them dropped below a certain point, they seemed to attract each other. A likely explanation for this is the effects of the propeller upwash acting at a certain distance from the drones. Due to this propeller upwash, as the drones approach each other, the lift on the propellers closer to each other increases, tilting the drones away. If the distance between them reduces even further, this upwash acts on the propellers on the other side of the intruding drone, tilting them towards each other.

Although these aerodynamic effects were quickly canceled by the inner-loop controller of the drones, they happen at potentially critical time instances, increasing the likelihood of a collision. Considering the flight trajectories shown in Figure 9(d), it can be observed that the red and green MAVs are flying together shortly before they collide. In simulations, for this failure mode of the avoidance behavior to result in a collision, external interference was typically required, for



(a) Distance estimates corresponding to Figure 9(a)

(b) Distance estimates corresponding to Figure 9(b)

(c) Distance estimates corresponding to Figure 9(c)

(d) Distance estimates corresponding to Figure 9(d)

(e) Distance estimates corresponding to Figure 9(e)

(f) Distance estimates corresponding to Figure 9(f)

Fig. 10. Trajectories and corresponding distance estimates for three experimental trials with three ARDrones using the evolved avoidance behavior.

instance having to avoid the arena bounds. In reality, the aerodynamic effects described above have likely contributed to the collisions occurring in such situations.

Figure 10 shows the true inter-MAV distances and the on-board estimates corresponding to the flight trajectories shown in Figure 9. The two main things to note on these figures are the high variance of the estimated distances and the periodic, sometimes consistent, over- and under- estimation of the inter-MAV distances.

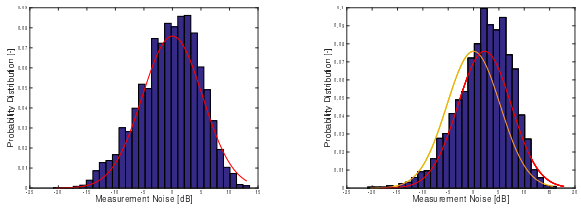
The variance of the distance estimates make it difficult for the avoidance behavior to determined when the drones are truly in conflict with each other, resulting in significant chattering between the drones trying to avoid each other and performing their task. Furthermore, this also results in the drones often repeatedly performing their avoidance turns, resulting in different, typically sub-optimal, flight trajectories than observed in simulations. A number of examples for such behavior can be observed on Figure 9. For instance on Figure 9(a), the red MAV performs two consecutive avoidance turns close to the center of the arena, while on Figure 9(e) both the red and blue MAVs perform their avoidance turn multiple times close to the arena's center.

In addition to the significant variance of the distance estimates, consistent, sometimes severe, over- or under-estimation of the inter-MAV distances that can be observed on Figures 10(a), 10(d), but especially Figure 10(b). These effects typically result in the drones either getting stuck in a state when they try to avoid another drone that is further away from them, locking their avoidance behavior, or not detect an impending collision, resulting in a collision. On all of these figures it can be observed that these estimation biases are not consistent across an entire trial, but only for certain time periods. This can be clearly seen when looking at the distance estimates shown in Figures 10(b) and 10(c). These two trials were part of the same dataset collected during a single flight. Looking at Figure 10(b), a severe underestimation of the distance between the blue and red drones can be observed, while the estimated distances shown in Figure 10(c) show a much better correspondence to the true distances.

In the simulation environment, the combined effect of sensor noise, interference and multi-path propagation was approximated as zero-mean Gaussian noise with a standard deviation $\sigma = 3.2$. From the data collected during the experiments presented here, the standard deviation of the measurement noise was estimated to be $\sigma = 5.27$, likely due to an increased use of the 2.4 GHz frequency band in the neighborhood of the flight arena compared to the time when sensor model was estimated.

An additional factor contributing both to the bias and the variance of the distance estimates was identified to be a mismatch between the Gaussian noise model used in the simulation environment and the noise distribution observed during the experiments. The parameters of the model presented in Section II and all calibration flights, due to space constraints in the flight arena, were performed with two ARDrones. Figure 11(a) shows the estimated Probability Density Function (PDF) of the measurement noise during one of the experiments performed with two ARDrones, together with a zero-mean Gaussian approximation with $\sigma = 5.27$. On this figure a good correspondence between the noise PDF and its Gaussian approximate can be observed.

Figure 11(b), shows the noise distribution with three ARDrones, where model parameters are the ones estimated from the dataset shown in Figure 11(a). The orange line



(a) Estimated PDF of the noise distribution with two ARDrones.

(b) Estimated PDF of the noise distribution with three ARDrones.

Fig. 11. The estimated PDF of the measurement noise with two and three ARDrones. The red line on the figures indicates a Gaussian distribution with $\sigma = 5.27$. On Figure (b) the orange line corresponds to a zero-mean Gaussian noise approximation, while the red line corresponds to this distribution shifted to the estimated PDF's mean.

indicates a zero-mean Gaussian assumption with $\sigma = 5.27$, while the red one corresponds to the Gaussian approximation shifted to the mean of the distribution. The mean of the estimated noise PDF is $\approx +2$ dB, while a mismatch between the PDF and its Gaussian approximation can also be observed, as there is a significantly higher probability mass in the 0 to $+6$ dB range than indicated by the Gaussian approximation.

The bias of the PDF observed on Figure 11(b) was removed before the start of the experiments by performing a test flight with three ARDrones employing one of the avoidance behaviors, based on the initial estimates obtained with two ARDrones. It is difficult to estimate the model parameters from such flights as the trajectory of the drones is a function of their initial conditions and the accuracy of the initial model parameters. Because of this, it often happens that the collected dataset does not properly cover the relevant distance and relative angle ranges, therefore the estimated model parameters are often incorrect, requiring the repetition of the identification flights. The datasets obtained from these flights were not used in the analysis of which the results are presented in Table II.

A probable cause of the observed noise bias is the interference caused by the Wi-Fi based data link between the third ARDrone and the ground station. Significant variations in the model parameters, especially for the reference signal strength A, were observed as a function of the Wi-Fi traffic in the environment. Increasing Wi-Fi traffic in the close environment typically resulted in an increase in the value of the parameter A. This, in turn, resulted in severe underestimation of the inter-MAV distances if the model parameters were not re-estimated. Such an event is the likely explanation of the severe distance estimation errors shown in Figure 10(e).

This observation is substantiated by the fact that estimated values for the model parameter A were -69.2 and -67.9 dB for the experiments with the manually developed and evolved avoidance behavior, where the latter of these were performed earlier in the day when there were more people working in the surrounding environment.

VII. CONCLUSIONS

In this article we have proposed two collision avoidance algorithms that rely solely on the signal strength measurements of an on-board receiver. The simulation experiments performed show that by utilizing either of the proposed avoidance strategies, the expected flight time of the MAVs doubles with respect to the case when no active collision avoidance is present. Furthermore, by using the avoidance behavior resulting from the EA, the probability of the MAVs surviving a trial more than doubles with respect to the alternative avoidance strategy, for MAVs larger than 0.4 meters in diameter.

Our real-world experiments indicate the success rate of the avoidance behaviors is close to the ones observed in simulations, resolving on average between $13 - 14$ conflicts per trial. The flight time of the drones, however, is reduced by a factor of ≈ 3.5 compared to the flight times observed in simulations, indicating a large reality gap. A number of sources were identified for this reality gap, the most important of which is the quality of the distance estimates obtained during the experiments, which is significantly lower than that observed in simulations. It is noteworthy that due to the presence of the avoidance behaviors, many of the scenarios that we classify as a collision in this article resulted only in a gentle bump, without causing a physical crash of the drones.

Based on our results, we conclude that ensuring intra-swarm collision avoidance based solely on distances estimated from RSSI measurements is difficult, if not impossible, in urban environments, due to the high noise variance and varying model parameters. Nevertheless, the collision avoidance algorithms proposed in this article were able to resolve a large number of conflicts, albeit the unreliable distance estimates, without requiring any additional sensors. Because of this, an RSSI based avoidance algorithm is a noteworthy alternative for MAVs with stringent payload limitations that operate in small groups and large spaces. In such cases, employing either of the proposed avoidance behaviors, though can not guarantee collision avoidance, can reduce the probability of a collision significantly.

Additionally, the quality of the distance estimates obtained from RSSI measurements could be significantly enhanced by using a different frequency band from the 2.4Ghz, eliminating a major part of the signal interference caused by electronic equipment commonly used in urban environments.

REFERENCES

[1] M. Keennon, K. Klingebiel, H. Won, and A. Andriukov, "Development of the nano hummingbird: A tailless flapping wing micro air vehicle." in *50th AIAA Aerospace Science Meeting*, 2012, pp. 6–12.

[2] K. Ma, P. Chirarattananon, S. Fuller, and R. Wood, "Controlled flight of a biologically inspired, insect-scale robot." *Science*, vol. 340, no. 6132, pp. 603–607, 2013.

[3] A. Kushleyev, D. Mellinger, C. Powers, and V. Kumar, "Towards a swarm of agile micro quadrotors." *Autonomous Robots*, pp. 287–300, 2013.

[4] C. De Wagter, S. Tijmons, B. Remes, and G. de Croon, "Autonomous flight of a 20-gram flapping wing MAV with a 4-gram onboard stereo vision system," in *ICRA*, 2014.

[5] M. Dorigo, V. Trianni, E. Şahin, R. Groß, T. H. Labella, G. Baldassarre, S. Nolfi, J.-L. Deneubourg, F. Mondada, D. Floreano, *et al.*, "Evolving self-organizing behaviors for a swarm-bot," *Autonomous Robots*, vol. 17, no. 2-3, pp. 223–245, 2004.

[6] C. Forster, M. Pizzoli, and D. Scaramuzza, "Svo: Fast semi-direct monocular visual odometry," in *ICRA*, 2014.

[7] Craig W. Reynolds, "Flocks, Herds, and Schools: A Distributed Behavioral Model," in *Computer Graphics, 21(4)*, 1987, pp. 25–34.

[8] T. Vicsek and A. Zafeiris, "Collective motion," *Physics Reports*, vol. 517, no. 3-4, pp. 71–140, 2012.

[9] G. et al. Vásárhelyi, "Outdoor flocking and formation flight with autonomous aerial robots. ," in *IROS*, 2014.

[10] O. Dunkley, J. Engel, J. Sturm, and D. Cremers, "Visual-inertial navigation for a camera-equipped 25g nano-quadrotor," in *Technical University, Munich, IROS 2014 Aerial Open Source Robotics Workshop.*, 2014.

[11] J. Roberts, T. Stirling, J.-C. Zufferey, and D. Floreano, "3-d Relative Positioning Sensor for Indoor Flying Robots," *Autonomous Robots*, vol. 33, no. 1-2, pp. 5–20, 2012. [Online]. Available: http://lis.epfl.ch/eyebot

[12] M. Basiri, F. S. Schill, D. Floreano, and P. Lima, "Audio-based Localization for Swarms of Micro Air Vehicles," in *IEEE International Conference on Robotics and Automation (ICRA 2014)*, 2014, pp. 4729 – 4734.

[13] J. Nembrini, A. Winfield, and C. Melhuish, "Minimalist coherent swarming of wireless networked autonomous mobile robots." in *From Animals to Animats 7: Proceedings of the Seventh International Conference on Simulation of Adaptive Behavior*. MIT Press, 2002, pp. 373–382.

[14] S. Hauert, J. C. Zufferey, and D. Floreano, "Evolved swarming without positioning information: An application in aerial communication relay," *Autonomous Robots*, vol. 26, pp. 21–32, 2009.

[15] L. Ludwig and M. Gini, "Robotic swarm dispersion using wireless intensity signals," *Distributed Autonomous Robotic Systems 7*, pp. 135–144, 2006.

[16] E. Ugur E., Turgut A. E. & Sahin, "Dispersion of a swarm of robots based on realistic wireless intensity signals. ," in *22nd international symposium on Computer and information sciences,*. IEEE, 2007, pp. 1–6.

[17] S. Nolfi and D. Floreano, *Evolutionary robotics*. Mit Press Cambridge, 2000.

[18] J. C. Bongard, "Evolutionary Robotics," *Commun. ACM*, vol. 56, no. 8, pp. 74–83, Aug. 2013.

[19] B. Remes, P. Esden-Tempski, F. van Tienen, E. Smeur, C. De Wagter, and G. de Croon, "Lisa-s 2.8g autopilot for GPS-based flight of MAVs," in *IMAV 2014*, 2014.

[20] J.S.Seybold, *Introduction to RF Propagation.* John Wiley & Sons, Inc., 2005.

[21] B. T. Ltd., "BLED112 bluetooth smart dongle." https://www.bluegiga.com/en-US/products/bled112-bluetooth-smart-dongle/ (accessed: 20/02/2015).

[22] A. Champandard, "Behavior trees for next-gen game AI." in *Game Developers Conference Europe 07*, 2007.

[23] X. Yu and G. Mitsuo, *Introduction to evolutionary algorithms*. Springer Science & Business Media, 2010.

[24] K. Deb, S. Pratab, S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm: NGSA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, 2002.

[25] D. Perez and M. Nicolau, "Reactiveness and navigation in computer games: Different needs, different approaches," *... Games (CIG), 2011 IEEE ...*, 2011.

[26] K. Scheper, "Behaviour Trees for Evolutionary Robotics." Master's thesis, Delft University of Technology, the Netherlands, 2014. [Online]. Available: http://repository.tudelft.nl/assets/uuid:dde8d42e-590a-465d-abaf-760ec304760f/kscheper_thesis.pdf

[27] J. Koza, "Genetic programming as a means for programming computers by natural selection," *Stat. Comput.*, 1994.

[28] D. Floreano and F. Mondada, "Automatic creation of an autonomous agent: Genetic evolution of a neural-network driven robot," *From Anim. to Animat.*, pp. 421–430, 1994.

[29] B. Miller and D. Goldberg, "Genetic algorithms, tournament selection, and the effects of noise," *Complex Syst.*, vol. 9, pp. 193–212, 1995.

[30] R. Poli and W. Langdon, "On the Search Properties of Different Crossover Operators in Genetic Programming," 1995.

[31] P. Angeline, "Subtree crossover: Building block engine or macromutation," *Genet. Program.*, 1997.

[32] B. Remes, D. Hensen, F. van Tienen, C. De Wagter, E. van der Horst, and G. de Croon, "Paparazzi: how to make a swarm of parrot AR drones fly autonomously based on GPS." in *IMAV 2013*, 2013.

Chapter 3

Literature Review

This chapter is presents a literature review on a number of topics required for developing an intra-swarm collision avoidance algorithm within the ER framework, that relies solely on RSSI measurements. The main questions considered in this review are:

- How can distance be estimated solely from RSSI measurements?
- What tools from ER should be used in the development of such a collision avoidance algorithm?
- What is an appropriate Artificial Intelligence (AI) framework to encode the resulting behavior in?

3-1 Estimating Distance from Signal Strength

In terms of achieving collision avoidance behavior based on RSSI measurements, the first, but perhaps also the most important question to answer is: *how can distance estimates be obtained from signal strength measurements*? Depending on the environment where the communication takes place, there are a number of deterministic and statistical models available. In ideal environments, assuming omni-directional antennas and no interference, the loss in the signal's power can be expressed by the Free Space Loss (FSL) propagation model (Seybold, 2005, p. 68, in non-logarithmic form) as given in Equation 3-1, where λ is the wavelength and d is the distance between the transmitter and receiver antennas. The FSL equation describes the power loss due to the transmitted signal power spreading over the surface of a sphere as it radiates in an ideal medium. In non-ideal environments the constant *2* in front of the equation is replaced by a parameter n, incorporating the power loss due to the non-ideal medium itself. There are two major factors that are not considered by the FSL model and have a significant effect on the RSSI measurements. These are the antenna gains and the effects of multi-path propagation and signal interference.

$$FSL[dB] = -2 \cdot 10log_{10}(\frac{\lambda}{4\pi d}) \tag{3-1}$$

A model similar to the one shown in Equation 3-1, taking the antenna gains into account, is the Friis Free-Space Loss model (Seybold, 2005, p. 67) as given in Equation 3-2. The additional model parameters compared to Equation 3-1 are G_{Tx} and G_{Rx}, which correspond to the transmitter and receiver antenna gains. Since the MAVs that make up the swarm are identical in principle, the gain terms are in fact only a function of the relative directions of the two MAVs, Ψ_{1-2} and Ψ_{2-1}.

$$FSL_{Friis}[dB] = -2 \cdot 10log_{10}(\frac{\lambda}{4\pi r}) + G_{Tx}(\Psi_{1-2}) + G_{Rx}(\Psi_{2-1}) \tag{3-2}$$

Since typically Radio Frequency (RF) transmitters radiate power in multiple directions, part of the transmitted signal gets reflected by the obstacles in the environment, such as the ground, walls or objects surrounding the transmitter. Part of these reflected signals is also received by the RF receiver, resulting in differences in the measured signal strength of which the magnitude depends on the phase offset between the Line Of Sight (LOS) and reflected signals (Seybold, 2005, p 164-172). The obstacle from which the reflected signal's effects are considered to be the most dominant is the ground plane. Assuming the ground plane to be perfectly flat, the distance d_x at which the ground reflected signal's effects start to dominate the measured signal strength is given by the crossover point, as shown in Equation 3-3. Here, h_t and h_r are the transmitter and receiver antenna's height above the ground, d_x the crossover distance and λ the signal's wavelength. At distances below the crossover point, the FSL model provides a suitable approximate, however at distances beyond $d_{crossover}$, the effects of the reflected signal need to be taken into account.

$$d_{crossover} = \frac{4\pi h_t h_r}{\lambda} \tag{3-3}$$

Accurately modeling the effects of multi-path propagation and RF interference requires the consideration of all obstacles and RF transmitters in the environment. As this approach would result in a location specific, statistical model and it is considered impractical within the scope of this research. A common approach found in literature to model the effects of interference, on a model general level, is via zero-mean Gaussian noise. One such model is the log-distance path loss model (Seybold, 2005, p. 214), as shown in Equation 3-4. This model considers the power loss between a transmitter and receiver pair in a similar fashion to the FSL model, and the effects of interference via Gaussian noise. The parameters of this model are: $PL(d_0)$ - a reference signal loss value at a distance of d_0 from the transmitter, n - the path loss exponent due to a non-ideal medium, d - the true distance and X_s - a zero-mean Gaussian random variable with variance σ^2.

$$L_{total}[dB] = PL(d_0) + n \cdot log_{10}(\frac{d}{d_0}) + X_s \tag{3-4}$$

By combining the Log-Distance Path Loss and the Friis Free Space Loss models discussed in Section 3-1, the model in Equation 3-5 can be constructed, which incorporates the most important factors of the signal's power loss. Here, $A(d_0)$ is a reference signal strength measured at a distance of d_0 meters from the transmitter, n is the path loss exponent, d is the true distance between the two antennas, $N(0, \sigma^2)$ is zero-mean Gaussian noise with variance σ,

$G(\Psi)$ is an antenna gain model. Ψ_{1-2} and Ψ_{2-1} are the relative angles between the transmitting and receiving antennas taking into account their true heading angles and their relative positions with respect to each other.

It should be noted that the parameters $PL(d_0)$ in Equation 3-4 and $A(d_0)$ in Equation 3-5 represent different things. $PL(d_0)$ represents purely a signal loss at a distance d_0, while A_0 represents a reference RSSI measurement at a distance d_0 from the transmitter, corresponding to the summation of $PL(d_0)$ and the transmitted signal power. Furthermore, since RSSI is an *indication* of the received signal's strength, the difference between the true signal strength value and the value of its indication is also incorporated into $A(d_0)$.

$$RSSI[dB] = A(d_0) - n \cdot 10log_{10}\Big(\frac{d}{d_0}\Big) + G(\Psi_{1-2}) + G(\Psi_{2-1}) + \mathcal{N}(0, \sigma^2) \tag{3-5}$$

In order to obtain a distance estimate from the measured signal strength, the model given in Equation 3-5 needs to be inverted. The main issue with inverting such a model is that the relative bearing angles Ψ_{1-2} and Ψ_{2-1} are not available for the MAV performing the estimation. Therefore, the assumption of constant gain (omnidirectional) antennas needs to be made. Typically, for linear applications with Gaussian noise, Kalman Filters (KFs)(Kalman, 1960) are used. However, since the parameter of interest lies inside a logarithmic function, due to inversion, the noise term becomes non-Gaussian. A popular nonlinear extension of the KF is the Extended Kalman Filter (EKF), which uses a linearized version of the nonlinear dynamics and/or output equations around the filter's state. Another, more recent extension is the Unscented Kalman Filter (UKF)(Julier & Uhlmann, 2004), where the mean and variance of the process are propagated through the dynamics and output equations by a number of Sigma points(Julier, 2002).

Both the EKF and UKF are able to handle nonlinear dynamics or measurement transformation, however they both maintain the assumption that the additive noise is Gaussian. The Gaussian noise assumption can be removed by the use of a Particle Filter (Thrun, 2002), which determines the most likely state, given a set of measurements, via Monte Carlo sampling. Particle filters allow the designer to specify not only a state transition and system output models, but also models for the state disturbance and measurement noise as well. The increased flexibility of the particle filter, however, comes at the cost of increased computational load as in many cases a large number of particles (potential solutions) may be required for the filter to converge.

3-2 Evolutionary Robotics

Evolutionary Robotics (ER) (Nolfi & Floreano, 2000; Bongard, 2013) takes inspiration from natural evolution, and uses biologically inspired methods for the automatic development (evolution) of robot structures (Auerbach & Bongard, 2012) and/or robot controllers (Maniezzo, 1994; Dorigo et al., 2006; Hauert, Zufferey, & Floreano, 2008; Floreano, Dürr, & Mattiussi, 2008; Petrovi, 2008; Pintér-Bartha, Sobe, & Elmenreich, 2012; Scheper, 2014). ER relies on EAs (Yu & Mitsuo, 2010) to perform a meta-heuristic search through the solution space, in order to optimize a set of performance metrics that define the desired characteristics of the robot's controller.

3-2-1 Genetic Algorithms

One of the most common forms of EAs are Genetic Algorithms (GAs), taking inspiration from the evolution of genetic material found in living organisms. They rely on maintaining a *population* of *genomes* or, in other words, *candidate solutions* for the problem that the GA is attempting to solve. The algorithm evaluates the *fitness* of these genomes, selects suitable parents and creates a new generation of genomes from these parents with the aim of improving their genetic material. This cycle is iterated until either the desired performance is obtained, or the algorithm reached some maximum number of generations. The four major steps that constitute most GAs are given below.

1. *Initialization*: an initial, typically random population is created
2. *Evaluation*: all potential solutions of the current population are evaluated based on the predefined cost functions
3. *Selection*: members of the population are selected for reproduction
4. *Reproduction*: a new generation is formed via recombination of the selected members

Initialization

In the initialization step, a population of typically random genomes is created. Having a large variety of genomes is beneficial, as it introduces a large pool of genetic material that the algorithm can use. Furthermore, the large spread in performance and solution strategies of the initial population allows the GA to have an initial idea of the likely region(s) of the solution space where the optimal solution may lie.

Evaluation

In order to identify the population's genomes with desirable traits, these are evaluated based on some performance metrics, or *cost functions*. These performance metrics define the desired properties of the final solution. Simpler GAs (Yu & Mitsuo, 2010, p 11-37) make use of a single cost function that incorporates all the relevant performance metrics, while Multi-Objective Genetic Algorithms (MOGAs) (Sbalzarini, Müller, & Koumoutsakos, 2000; Deb, Pratap, Agarwal, & Meyarivan, 2002) measure all of these performance metrics independently. Considering the fact that as an end result of the evolution one wants to obtain a single solution to the problem at hand, the main difference between the simpler, single objective GAs and their multi-objective counterparts is where the trade-off between the different objectives has to be made. Using a single objective function requires the different objectives to be weighted based on their level of importance before the evolutionary process begins. Multi-objective GAs can converge to the *Pareto front*, the region of the solution space where no one objective can be further improved without sacrificing performance on a different one.

Selection

In the selection step, different combinations of *parents* are selected from the population based on their performance obtained in the evaluation step. It is important to not only select the best performing genomes for reproduction as this would result in discarding a significant part of the genetic material and it may result in the premature convergence of the algorithm to a suboptimal solution. Two commonly used selection algorithms are the Roulette Wheel Selection (Yu & Mitsuo, 2010, p 19-20) and Tournament Selection (Miller & Goldberg, 1995). In the Roulette Wheel Selection algorithm each candidate solution is considered to be a slice of a roulette wheel proportional in size to its relative fitness (with respect to the other individuals of the generation). By taking a uniform random number in the [0,1] range as "throwing the ball" the individual corresponding to the sector where the "ball" ends up is selected, hence the relative fitness of an individual is proportional to the chances of being selected. Tournament Selection is an algorithm where the best candidate is selected from N randomly selected candidates. The tournament size N determines the selection pressure, controlling the convergence rate of the EA, as it determines the ratio between exploration of the solution space and the exploitation of the closest local optima.

Reproduction

The last step of the evolutionary cycle is the creation of a new population from the selected parents. This is done by making use of the crossover and mutation genetic operators. The crossover operator recombines the genetic material of two parents, forming two children for the next generation of the algorithm, allowing the exploration of the solution space. Mutations are used as a means to introduce random variations to the genetic material. Depending on the amount and type of modified genetic material, mutations can either facilitate exploration when large changes are introduced or local exploitation via small changes. The proper balance between exploitation and exploration is an important factor for GAs, since an algorithm that only exploits the available genetic material could potentially result in premature convergence to a sub-optimal solution, while an algorithm that only explores may never really converge to a solution.

The exact operation of the genetic operator depend on the encoding scheme used for the EA's solution. Traditionally, GAs use bit-strings or real valued vectors for encoding behavior. For such a structure a simple crossover operator can be implemented by selecting a random point in the genomes, with respect to which part of the genetic material comes from one of the parents, while the remainder from the other parent. Mutations can also easily be implemented by the random offset of some randomly selected parameters (Yu & Mitsuo, 2010). More advanced genetic operators for these genome types are discussed in (Floreano et al., 2008).

For optimizing genomes encoded in a tree structure, a simple crossover operator is the sub-tree crossover (Koza, 1994), where two randomly selected sub-trees are swapped. A number of alternative crossover operators for tree structures, inspired by the operators used for bit-string genomes, such as the one point crossover or uniform crossover operators are discussed in (Poli & Langdon, 1995). Mutations can be performed with the Headless Chicken Crossover (Angeline, 1997) for large mutations, or by random changes in the parameters of the different branches for smaller mutations.

3-2-2 Multi-Objective Genetic Algorithms

There are a number of different MOGAs available in literature (Sbalzarini et al., 2000; Deb et al., 2002). One of the most significant differences between different GAs is the way two genomes are compared with each other. With a single objective to optimize, the preference of one genome over another can be decided by comparing their performance on that one objective. Such a comparison is not always possible when multiple objectives are involved as one may perform better than another on one objective while performing worse on a different one. A common way amongst MOGAs is to rank the genomes based on whether one *dominates* the other. A genome is said to *dominate* another genome, when it performs better on at least one objective while not performing worse on any of the other objectives. This allows the genomes to be assigned to *domination fronts*, where no one genome is better or worse than any other.

The main difference between different MOGAs is the way they rank the genomes within one domination front. The Strength Pareto Evolutionary Algorithm (SPEA) (Sbalzarini et al., 2000) ranks the genomes within the same domination front based on their strength, which is proportional to the number of genomes that they dominate if these are on the first (non-dominated) front, or to the sum of strengths of the genomes that dominate them plus one if they lie on any other front.

The Non-dominated Sorting Genetic Algorithm II (NSGA-II) (Deb et al., 2002) takes a different approach and attempts to equalize the density of the solutions in the different regions of the domination fronts. In order to do that, it introduces the *crowding distance*, which is proportional to the perimeter of an n dimensional cuboid in the solution space of which the sides are equal to the distance between the genome's neighbors on either side along each objective. This will be further explained in Section 5-1.

NSGA-II, SPEA and a number of other MOGAs are compared on a set of test problems in (Deb et al., 2002). The results presented in their work shows that the NSGA-II they developed has a higher performance on most of the implemented test problems. This is likely due to the fact that NSGA-II attempts to spread the solutions on the non-dominated front, introducing more genetic material from which better offsprings can be created.

3-3 Artificial Intelligence Framework

One of the first steps in the design of any behavior is to determine the most appropriate framework in which to encode it in. Considering that within this research Evolutionary Robotics is to be used, the literature review presented in this section is restricted to AI frameworks that are have previously been used within ER. The two areas of focus in this review are the expressive power of the different AI frameworks and the ease with which the designer can understand the resulting behavior. The latter is necessary as within ER the behavior is typically evolved in a simulation environment, which does not provide a perfect representation of reality. Because of this, the manual adaptation of the resulting behavior is often required in order to to bridge the reality gap (Scheper, 2014).

Biologically inspired Artificial Neural Networks (ANNs) intend to mimic the way a brain works. Their output depends on the interconnection of a number of artificial neurons. Due to their structure, ANNs are capable of approximating virtually any input-output relationship,

given that there are sufficient neurons present in the network. Because of their simplicity, flexibility and approximating power, ANNs have been extensively used in ER (Maniezzo, 1994; Dorigo et al., 2006; Hauert, Zufferey, & Floreano, 2008; Floreano et al., 2008). The black-box encoding mechanism of the ANNs, however, results in a structure that is difficult for humans to understand or adapt manually. Even though there are methods to analyze the evolved or trained ANN, typically a complete re-evolution or re-training of the network is required for adapting the network.

One of the most common methods for encoding manually designed autonomous behaviors is via Finite State Machines (FSMs). FSMs encode autonomous behavior by considering all possible states in which the system may be in and list all possible state transitions that may occur. This results in a structure that is easily understandable for humans as long as the number of states/state transitions is small. For more complex systems, the *state explosion problem* (Valmari, 1998) appears, as the number of possible state transitions grows exponentially as more states are added to the system. While less common, FSMs have also been successfully used in ER (Petrovi, 2008; Pintér-Bartha et al., 2012; Francesca, Brambilla, Brutschy, Trianni, & Birattari, 2014). In terms of understandability FSMs outperform ANN, however, the advantages of FSMs quickly fade away as the number of required system states grows, furthermore, in terms of performance evolved ANNs tend to outperform evolved FSMs (Pintér-Bartha et al., 2012).

A more recent method of encoding autonomous behavior is via Behavior Trees (BTs). Originally developed as a formal way of specifying system requirements (Dromey, 2003), BTs were adapted by the computer-gaming industry for encoding the behavior of non-playing characters (Champandard, 2007). Next to their use in computer games, in the past years BTs have been proposed in robotics applications as well (Marzinotto, Colledanchise, Smith, & Ogren, 2014; Colledanchise, Marzinotto, & Ogren, 2014; Scheper, 2014). BTs operate based on subsumption architectures (Brooks, 1990), recursively decomposing the desired behavior into sub-behaviors and execute them in a sequential order, which implicitly enables BTs to prioritize actions. Furthermore, the tree-structure of the BTs, together with the presence of the state-returns from the executed branches, implicitly encodes information about the state in which the robot is in (Ogren, 2012), allowing BTs to remain compact and intelligible for humans, without suffering from the state explosion problem.

The most common behavior encoding framework for ER is ANNs, which are flexible, expressive and have well developed genetic operators (Floreano et al., 2008). Their main drawback is the difficulty in understanding the resulting behavior and the necessity of their re-evolution even for the smallest changes. FSMs offer an easy to understand representation for the underlying behavior, however their complexity for larger problems make them difficult to manage. Using BTs maintains and facilitates the human understandability of the resulting behavior even for larger systems. Even though the expressive power of BTs is different and perhaps a bit more restrictive compared to ANNs when it comes to approximating continuous Input/Output (I/O) relationships, they have achieved good results within ER, therefore they are considered as the appropriate AI framework to be used within this thesis.

Chapter 4

Experimental Setup

The research question defined in Chapter 1 does not define the task that the MAVs are required to perform. The main reason behind is the attempt to develop a behavior capable of avoiding inter-MAV collisions while the MAVs themselves perform an *arbitrary* task. Because of this, throughout this thesis the task of the MAVs is defined as:

The MAVs shall explore an unknown area that is constrained in size, by using a specific BT branch that makes them fly straight at high velocity, while turning away from the bounds of the area.

The remainder of this chapter is intended to describe the experimental setup used within this thesis. In Section 4-1 the MAV platform used to evaluate the real-world performance of the avoidance behavior is presented, while Section 4-2 discusses the simulation environment that was constructed based on a simplified model of the sensor and flight platform's characteristics. Finally, in Section 4-3 MAV's controller module is presented, which includes the RSSI based distance estimation filters and serves as an interface between the BT and the MAV's autopilot.

4-1 The Flight Platform

The flight platform selected for this thesis is the Parrot ARDrone2 quadcopters running the Paparazzi UAV open-source autopilot (Remes et al., 2013). Even though these quadcopters are already equipped with WiFi transmitters that are used as datalink between the autopilot and the ground station, the long-term goal of the Delft University of Technology (DUT)'s MAVLab is to have a swarm of nano quadcopters based on low energy Bluetooth BLE connections. Therefore, the sensor chosen to be used within this thesis is the Bluegiga BLED112 low energy USB dongle (Bluegiga Technologies, 2014).

4-1-1 The Parrot ARDrone2 & Paparazzi UAV

The Parrot ARDrone2 is a commercially available, low-cost quadcopter that has been widely used as a research platform in the scientific community. (Krajník, Vonásek, Fišer, & Faigl, 2011; Remes et al., 2013) It features an array of on-board sensors, such as a high resolution, large field of view front facing camera (720p, 92°), a bottom camera, sonar based altimeter, 3-axis accelerometer, 3-axis high precision gyroscope, 3-axis magnetometer and a pressure sensor. Furthermore, the ARDrone2 features a 1 GHz ARM Cortex A8 processor with 1 GB DDR2 RAM, on-board Wi-Fi and a USB 2.0 port. (Parrot, 2015)

In terms of software, the ARDrone2 comes with a preinstalled autopilot that provides communication and control for the drone. The feature set provided by this autopilot is extendible via a Software Development Kit (SDK). In this thesis, however, the preinstalled autopilot is completely replaced by the Paparazzi UAV firmware in order to gain more control over the flight behavior of the platform.

Paparazzi UAV is an open-source project aimed at developing autopilot software for hobby and professional use. (Brisset, Drouin, Gorraz, Huard, & Tyler, 2006) It features configurable, modular approach to the development process, where the autopilot uploaded to the flight platform is compiled based on an XML based configuration (airframe) file. With this approach, modules implementing different features can be loaded based on the design requirements, enabling code reusability and flexibility.

As the MAV's inner-loop control is beyond the scope of this thesis, an existing controller from within Paparazzi is used. The input variables of this controller are the commanded velocities in the X and Y coordinates in the North, East, Down (NED) coordinate frame, together with the flight altitude and the MAV's heading. In addition to these inputs, the inner-loop controller relies on attitude angle estimates from the on-board Inertial Measurement Unit (IMU) and velocity estimates from, for example GPS or the bottom facing camera.

4-1-2 The BLED112 Bluetooth USB dongle

Considering the long term goals of the DUT's MAVLab of having a swarm of nano quadcopters, the BLED112 low energy Bluetooth module was chosen as the device of which to measure the RSSI. Since within this thesis, the MAVs are not intended to have active communication amongst them and the data link between the MAVs and the Ground Control Station (GCS) has already been implemented via a Wi-Fi link, the MAVs periodically broadcast dummy transmissions over the attached BLED112 modules, of which the RSSI are measured by the other MAVs.

4-1-3 The CyberZoo

The real-world tests used to validate the performance of the avoidance behavior were conducted at the DUT's CyberZoo, a 10x10x7 meter flight arena equipped with an Optitrack system. The Optitrack system allows the measurement of the position, velocity and heading

of objects via a set of reflective IR markers. These measurements are, on one hand, used to construct a fake GPS signal for the ARDrone2's inner-loop controllers, while on the other hand, they are used to analyze the performance of the avoidance behavior. At this point, it should be noted that the velocity information derived from the fake GPS signal provided by the Optitrack system is used solely by the inner-loop controllers for tracking the commanded velocities, while the position information is only used to ensure that the ARDrones remain within the designated virtual flight arena, as described below.

The validation experiments are performed at an altitude of 0.5 meters above ground inside a 6x6 meter virtual flight arena, which the MAVs are not allowed to leave. When they approach the boundaries of this virtual arena, the MAVs turn away from it at the same angle as their incidence angle, as shown on Figure 4-1. During the collision avoidance behavior's design, this behavior is assumed to be already available for the MAVs, therefore it is considered as part of the designed behavior.

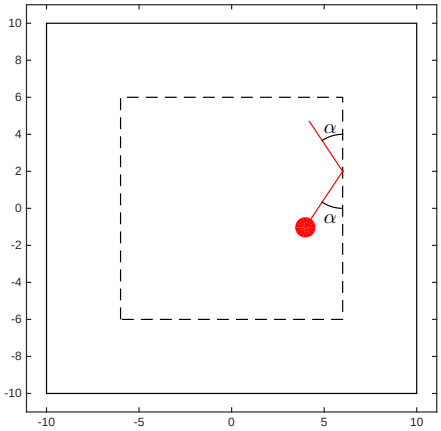


Figure 4-1: An MAV turns away from a wall of the virtual flight arena. The incidence and departure angles of the MAV to/from the wall segment are equal.

4-2 Simulation Models

During the avoidance behavior's development, a simulation environment was constructed based on the models of the selected flight platform described in the previous section. In this section, these models and their implementation in the simulator are discussed.

Considering that in part of this thesis EAs are used, the primary concern during the simulation environment's development was the computational time that a given simulation run takes. In order to minimize the computational resources required, thereby decreasing the simulation time, the simulator was developed in C++, based on the simplified MAV model discussed Sections 4-2-1 - 4-2-3.

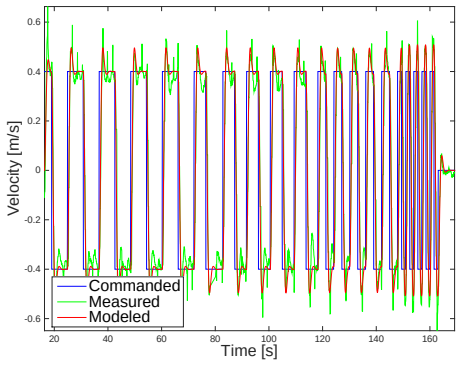


Figure 4-2: Commanded, measured and modeled flight velocities of the ARDrone2

4-2-1 The MAV Dynamics Model

The dynamics of a quadcopter has six degrees of freedom and four inputs, where the different degrees of freedom are the rotations and translations along the three axis, while the inputs are the rotational speeds of the four motors. This results in coupled, nonlinear dynamics of which the control is beyond the scope of this thesis. If, however, the dynamics of the MAV is modeled together with the inner-loop controllers, a decoupled linear model is a good approximation for the dynamics of a quadcopter in its normal flight regime.

In order to minimize the computational requirements of the MAV's dynamics model while at the same time maintain sufficient accuracy such that the simulation results remain representative, a linear, 2nd order velocity model was implemented in both the X and Y directions, without axis coupling. The parameters of this model, ω and ζ, were estimated from flight tests with an ARDrone2. In order to identify the parameters of the MAV's dynamics, an ARDrone2 was commanded to fly forward with a velocity of 0.4 m/s, than fly back with the same velocity. The time between these two states was incrementally decreased, resulting in a frequency sweep. For the recorded dataset, the model parameters were estimated to be 2.74 rad/s and 0.56 for ω and ζ respectively. The commanded, true and modeled velocities of one of the identification trials are shown in Figure 4-2.

The resulting MAV dynamics model consists of seven states: accelerations, velocities and positions in the X and Y world coordinates and the altitude at which the MAV is at. The altitude dynamics is modeled as a 1st order transfer function with a time constant $\tau = 0.3s$ - however, since there is no active control over the flight altitude, this state stabilizes and remains constant at 0.5 meters shortly after the start of a simulation.

The MAVs are restricted to flying only along their longitudinal axis, therefore the inputs of the dynamics model are defined to be the commanded flight heading Ψ_c, commanded velocity V_c and the commanded (constant) altitude h_c. The resulting state-transition model is given in Equation 4-1.

$$\begin{bmatrix} \dddot{X} \\ \ddot{X} \\ \dddot{Y} \\ \ddot{Y} \\ \dot{X} \\ \dot{Y} \\ h \end{bmatrix} = \begin{bmatrix} -2\zeta\omega & -\omega^2 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -2\zeta\omega & -\omega^2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{\tau} \end{bmatrix} \begin{bmatrix} \ddot{X} \\ \dot{X} \\ \ddot{Y} \\ \dot{Y} \\ X \\ Y \\ h \end{bmatrix} + \begin{bmatrix} \omega^2 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & \omega^2 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \frac{1}{\tau} \end{bmatrix} \begin{bmatrix} V_c \cdot cos(\Psi_c) \\ V_c \cdot sin(\Psi_c) \\ h_c \end{bmatrix} \quad (4\text{-}1)$$

Two important differences were observed between the real and modeled dynamic behavior of the ARDrones. The model described in this section neglects the axis coupling that can be observed during aggressive turn maneuvers, which result in non-negligible sway motion on the real flight platform. The effects of this are most dominant when observing the behavior of the real flight platform during a wall avoidance maneuver, when the drone approaches the wall at a steep incidence angle, requiring it to perform a sharp turn. This maneuver results in smooth, small radius turns in the simulator, but significant sway motion can be observed on the real flight platform, often resulting in a considerable displacement when reentering the designated flight region.

Another important difference is that the simulation model developed in this section does not consider the aerodynamic effects of the propellers. These effects are not relevant during the flight of a single drone, or multiple drones at higher altitudes, but become important if two of them are close to each other at low altitudes while flying at low speeds. In these cases it was observed that in a certain distance region, the two drones tend to push each other away, while at a different, closer distance, these effects tend to prevent the drones from separating from each other.

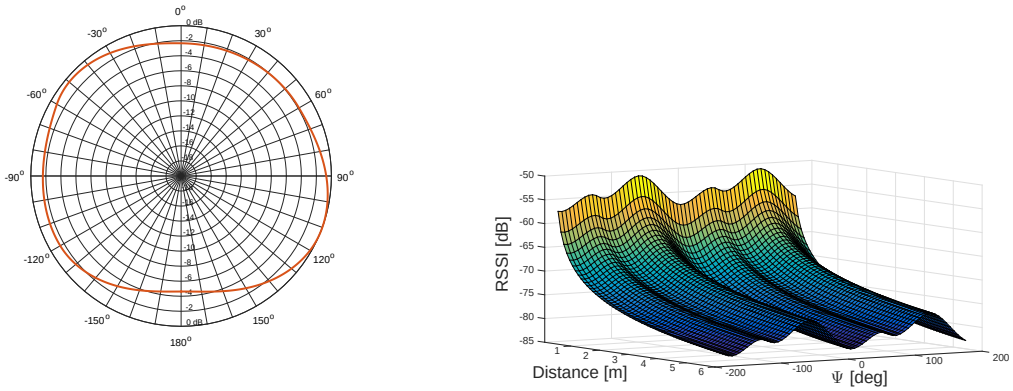
A possible explanation for these effects is the upwash of the propellers acting at a certain distance from the drones. As two drones approach each other, the upwash increases the lift of the propellers that closer to each other, tilting the drones away. On the other hand, if the drones are closer to each other than the distance where the propeller upwash is acting, the outer propellers experience the increased lift, tilting the drones towards each other, making it harder for them to separate. Eventually the inner-loop controller of the drones eliminates these effects, however, they are significant deviations from the simulation model at potentially critical time instances.

4-2-2 The RSSI Sensor Model

Based on the discussion of Section 3-1, the RSSI of the BLED112 Bluetooth USB dongle is modeled as given in Equation 3-5, repeated here as Equation 4-2. The parameters of this model are: A - a reference signal strength measurement at a distance of one meter, n - the path loss exponent, $G(\Psi)$ - the antenna gain function and σ^2 - the noise variance. The filters and assumptions used to eventually invert this model are discussed in Section 4-3-1.

$$RSSI = A - 10n \cdot log_{10}(d) + G(\Psi_{1-2}) + G(\Psi_{2-1}) + \mathcal{N}(0, \sigma^2) \quad (4\text{-}2)$$

In order to estimate the parameters of this model in representative environments, a number of test flights were performed with two ARDrones equipped with the BLED112 modules. In



(a) The antenna gain as a function of the relative angle Ψ between the transmitter and receiver

(b) The RSSI as a function of the relative angle Ψ and distance d between the transmitter and receiver

Figure 4-3: The RSSI model of the BLED112 Bluetooth USB dongle.

order to obtain a representative dataset for the parameter estimation, the drones were required to fly along concentric circles, with one and three meter radii respectively, with the same velocity. This allows the MAVs to go through all possible relative angles (Ψ_{1-2} and Ψ_{2-1}) and most of the relevant distance range as well.

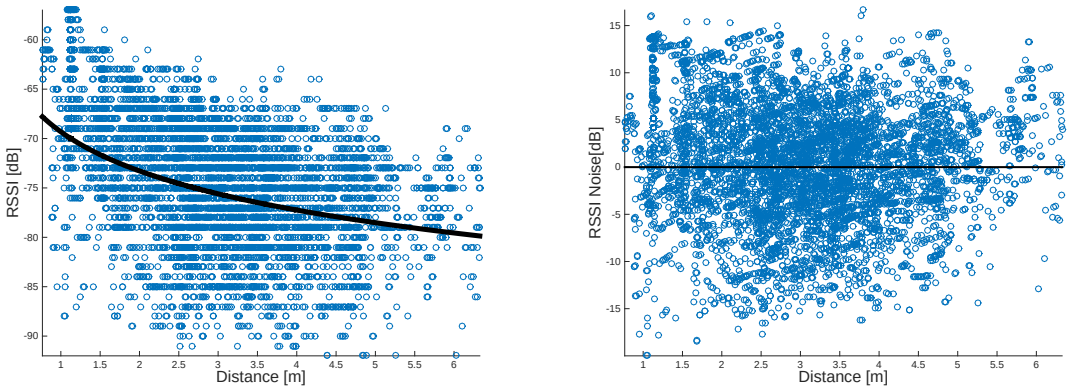
The antenna gains $G(\Psi)$ are modeled as 3rd order splines with 1st order continuity. The mean of the antenna gains across the relative angle range, $[-\pi, \pi]$, is equal to zero. The spline itself consists of six equal sized segments of ≈ 1.05 rad. The coefficients of the spline were estimated form the recorded flight data together with the model parameters A and n.

Figure 4-4a shows the measured signal strength as a function of the distance between two MAVs, where the black line represents the identified RSSI - distance relationship, not taking into account the effect of the antenna gains. The model represented by the black line represents the information available for estimating the transmitter-receiver distance. The detailed discussion of the distance estimation algorithms is given in Section 4-3-1.

Figure 4-4b shows the measurement noise due to the noise of the sensor, multi-path propagation effects and the measured interference of other transmitting devices in the environment. From these figures the last parameter of the sensor model, standard deviation of the noise, σ, can be estimated to be ≈ 3.2. It should be noted that since the Bluetooth protocol utilizes the commonly used 2.4GHz frequency band, the measurement noise varies significantly depending on the actual environment. If there are a large number of devices in the environment using this frequency band, the value of σ^2 increases significantly.

4-2-3 The Virtual Flight Arena

The virtual flight arena is the virtual space where the simulated MAVs perform their tasks and come into conflict with each other. The main task that it servers within the simulator is to propagate the MAVs in time and provide the true distances between the MAVs to their sensor models. Since the method by which the MAVs avoid the bounds of the designated



(a) The scattered dots represent the signal strength measurements at varying distances. The black line represents the modeled RSSI measurements, not taking the antenna gains into account.

(b) The noise of the measured RSSI.

Figure 4-4: RSSI as a function of distance.

flight region is not part of this thesis, a secondary task of the virtual flight arena is to ensure that the MAVs stay within their bounds of the flight arena by signaling them to turn as they approach the arena's bounds.

Another task of the flight arena is to allow the calculation of both the cost functions of the EA, and the performance metrics of the avoidance behavior. At each time step of the simulation, an evaluation function is called which, on one hand, determines if a collision between two MAVs has occurred, while on the other hand allows the calculation of the performance metrics of interest. The cost functions of the EA and specific metrics used to evaluate the avoidance behavior's performance are discussed in detail in Sections 5-3 and 7-2-1 respectively.

4-3 Controller Structure

The controller module of the MAVs consists of two parts. On one hand, it contains the distance estimation filters, discussed in Section 4-3-1, that allow RSSI-distance relationship to be inverted, while on the other hand, it also contains the BT defining the behavior of the MAVs. The exact implementation of the BT is presented in Appendix B. The inputs of the controller module are the measured RSSI values from the other swarm members, while the module's outputs are the commanded velocity, commanded heading angle and the commanded flight altitude - the last of which is set to a constant 0.5 meters.

The distance to each neighbor in the swarm is tracked individually via distance estimation filters and the smallest of these estimates is passed to the BT as its sole input. The outputs of the BT are defined to be the commanded velocity, turn rate and a commanded heading increment. From these, the turn rate command is meant for continuous heading changes, while the heading increment is meant to represent an instant, large change in heading angle. The last two of these are integrated within the controller module to produce the commanded MAV heading.

4-3-1 Distance Estimation Filters

In order to estimate the inter-MAV distances, the model given in Equation 4-2 needs to be inverted. The main reason why the simple inversion of this model is impossible lies in the fact that the antenna gain terms $G(\Psi_{1-2})$ and $G(\Psi_{2-1})$ are unknown and can not be estimated from a single RSSI measurement. Furthermore, as the inter-MAV distance d lies inside a *log* function, simply inverting this model would cause the noise term $\mathcal{N}(0, \sigma^2)$ to undergo an exponential transformation, amplifying the measurement noise.

In order to obtain distance estimated from the RSSI measurements, in this work the antenna gains $G(\Psi_{1-2})$ and $G(\Psi_{2-1})$ are assumed to be zero. The effects of such an assumption are illustrated on Figure 4-5, showing the significant deviations between the true transmitter-receiver distance and the one estimated from RSSI measurements. The results indicated in this figure are simulation results and do not consider the effects of noise/interference. From this figure it can be observed, that the magnitude and sign of the estimation error is a function of the relative heading between the transmitter and receiver antennas.

By assuming the antenna gain terms to be zero, the model is simplified to the form given in Equation 4-3, which, in fact, corresponds to the free-space-loss model given in Equation 3-1 with an additional noise term.

$$RSSI[dB] = A - 10n \cdot log_{10}(d) + \mathcal{N}(0, \sigma^2) \tag{4-3}$$

This equation can be inverted by a one dimensional EKF, with the single state corresponding to the estimated distance. The state and output prediction equations, together with the output equation's derivative are given in Equation 4-4, while the remainder of the algorithm used in Equation 4-5. The parameters of the filter, Q and R, are set to be 0.1 and 100 respectively.

$$\begin{aligned} \hat{d}_{k+1|k} &= \hat{d}_k \\ \hat{h}_{k+1|k} &= A - 10n \cdot log_{10}(\hat{d}_{k+1|k}) \\ \hat{H}_d &= \frac{-10n}{\hat{d}_k \cdot log(10)} \end{aligned} \tag{4-4}$$

$$\begin{aligned} P_{k+1|k} &= P_k + Q \\ K &= P_{k+1|k}\hat{H}_d(\hat{H}_d P_{k+1|k}\hat{H}_d + R)^{-1} \\ \hat{d}_{k+1|k+1} &= K(RSSI_{pre} - \hat{h}_{k+1|k}) \\ P_{k+1|k+1} &= (1 - K\hat{H}_d)P_{k+1|k}(1 - K\hat{H}_d) + KRK; \end{aligned} \tag{4-5}$$

The parameter $RSSI_{pre}$ in the state update equation corresponds to a prefiltered RSSI value, where the prefilter takes the highest RSSI reading in the past 0.5 seconds. The main reason behind applying such a prefilter is to reduce the effects of interference and measurement noise. Furthermore, since the MAVs move continuously, this prefilter also limits the effects of the antenna gains to a certain extent. A drawback of using such a

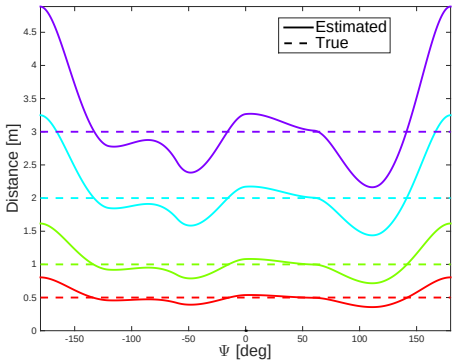


Figure 4-5: Distance estimates for given RSSI values. Dashed lines represent the true transmitter-receiver distances, while the solid lines the distance estimates as a function of relative angle, if the antenna gains are assumed to be zero.

prefilter is that it introduces a bias towards smaller distance estimates.

Such a bias is acceptable, or even preferred when compared to a bias towards over-estimating the distances, as it only results in unnecessary avoidance maneuvers instead of possible collisions. An additional, less desirable effect of such a prefilter is that as the distance between two MAVs *decreases*, the output of such a filter follows the corresponding increase in measured signal strength, however, the decreasing RSSI readings that accompany the increasing inter-MAVs distances are followed with a delay of ≈ 0.5 seconds.

4-3-2 Incorporating the Avoidance Behavior

The avoidance behavior of the MAVs designed in this thesis are encoded in the form of BTs. The flow through a BT is defined by the composite nodes and the state-returns of the leaf nodes. Inputs, current outputs and any internal parameters used are accessed by the nodes via a Blackboard structure, serving as the memory of the BT. The structure of the Blackboard consists of one input, the distance to the closest MAV, three outputs, corresponding to the velocity-, turn rate- and heading increment- commands and one additional parameter that can be used as an internal state.

In this work two types of composite nodes are used. *Sequences* (marked with "→") are composites that *tick*, i.e. trigger or evaluate, their children until either all of them succeed or one of them fails, in which case they return success or failure respectively. *Selectors* (marked with "?") are the other type of composite node used, which tick their children until one of them succeeds or all of them fail, returning success or failure respectively.

Next to the composite nodes, two types of leaf nodes are used. *Condition* nodes compare a Blackboard parameter to a constant, returning success if the comparison evaluates to true or failure otherwise. Comparisons can be performed with the *greater-* or *less-* than operators. The only action node implemented is the *Set* node, which equates one of the Blackboard parameters to a constant value. The Set node always returns success. A more comprehensive overview of the BT implementation is given in Appendix B.

Chapter 5

Evolving an Avoidance Behavior

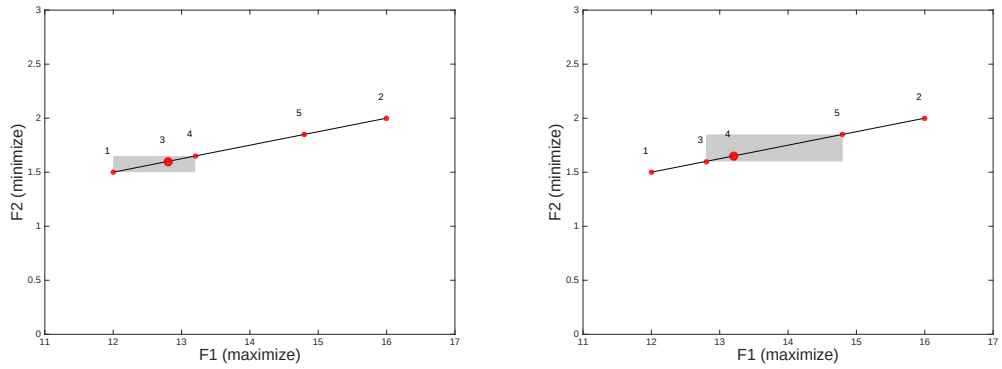
In this chapter the evolutionary setup used in this thesis is presented. Section 5-1 presents the GA used to optimize the MAV's controller with the genetic operators discussed in Section 5-2. The cost functions used during the evolution are discussed in Section 5-3, while the algorithm parameters are summarized in Section 5-4. Finally, the results of the evolution are presented in Section 5-5.

5-1 The NSGA-II Algorithm

The Non-dominated Sorting Genetic Algorithm II (NSGA-II) (Deb et al., 2002) is an elitist, MOGA that attempts to spread the obtained solutions evenly on the Pareto front, where no one solution is better than another. As discussed in Section 3-2-2, MOGAs rank the solutions based on domination, where a solution is said to dominate another one if it performs better on at least one performance metric while not performing worse on any other ones. This allows the algorithm to separate the candidate solutions to different domination fronts, based on the number of individuals that dominate a specific solution.

The main difference between NSGA-II and other MOGAs is the method by which solutions are ranked within one domination front. The *crowding distance* of a solution is defined to be proportional to the perimeter of an n-dimensional cuboid in the solution space, constructed between the solution's neighbors along the different objectives. The NSGA-II algorithm, in an attempt to spread the solutions evenly on the different domination fronts, makes use of this metric as an indication of the density of solutions in a certain region of the solution space. An example of how crowding distance is estimated is shown on Figure 5-1. Here the crowding distance of genomes three and four, part of the same domination front, correspond to the perimeter of the gray rectangles constructed around them, indicating which of the two solutions lies in a less densely populated region of the domination front.

Elitism is ensured within NSGA-II by making use of an *archive*, which contains the best performing solutions from the previous generation(s). In the implementation used within this thesis, the size of the archive is equal to the size of the population, N_{pop}. In each generation,



(a) Crowding distance of Genome 3

(b) Crowding distance of Genome 4

Figure 5-1: The crowding distances of the genomes are proportional to the perimeter of the rectangles (two dimensional cuboids) that surround them. It can be seen, that Genome 4 has a larger crowding distance, since it lies in a less dense region of the domination front, therefore it is a better candidate than Genome 3 for reproduction

a new population is created from this archive. Then, the members of this new population are combined with the archived solutions and the archive is again filled with the best performing solutions from this combined population. Such an approach ensures that none of the best N_{pop} solutions are discarded until a better alternative is found.

5-2 Genetic Operators for Behavior Trees

Encoding BTs in the form of common GA genomes, like bit-strings or real-valued strings, is difficult due to the information contained in the structure of the tree itself. An encoding form used to evolve BTs, similar to the one typically used in GAs is *grammatical evolution* (Perez & Nicolau, 2011). Grammatical evolution, however, still imposes restrictions on the shape of the evolved tree and makes it difficult to control its branching. Tree structures are typically evolved in the field of Genetic Programming (GP) and the genetic operators used in this work are taken from there.

Initialization

The initial population, consisting of N_{pop} genomes are initialized using the *grow* (Koza, 1994) method. All nodes are initialized with their maximum number of children, N_{child}, which are randomly selected from the available node types. The tree's depth is limited to N_{Depth}, where all nodes are leaves. This results in an initial population with large variations in the trees' shape, beneficial for the GA. The blackboard variables available for the Conditions nodes correspond to all BT inputs, outputs and internal states, while the Action nodes have access only to the Blackboard's output and state variables. The Blackboard variables on which a node operates are selected randomly from the ones available for them.

Evaluation

In each generation, the behaviors represented by population members are evaluated. The simulation environment used for evaluating the members was presented in Section 4-2, while the cost functions used are discussed in Section 5-3. Each individual is evaluated N_{Sim} times on the same set of N_{Sim} initial conditions to estimate the overall performance of the encoded avoidance behavior, not tied to one specific initialization. The duration of a simulation is limited to T_{Sim} seconds, however a trial is stopped as soon as two MAVs collide with each other.

Selection

Candidate members for creating the new generation are selected by a version of the Tournament Selection (Miller & Goldberg, 1995), where a group of N_T members of the current population are selected randomly, compared against each other and the "winner"of the tournament, with the best performance is returned. The tournament's size, N_T determines the selection pressure. A higher N_T corresponds to a faster convergence of the algorithm, since if N_T corresponds to N_{Pop}, the best individual from the current population will always be selected for reproduction, while a tournament of size 1 corresponds to randomly selecting members for reproduction. Care should be taken when choosing the value of N_T, as a too high selection pressure may result in a premature convergence of the population to a local optima, resulting in sub-optimal performance.

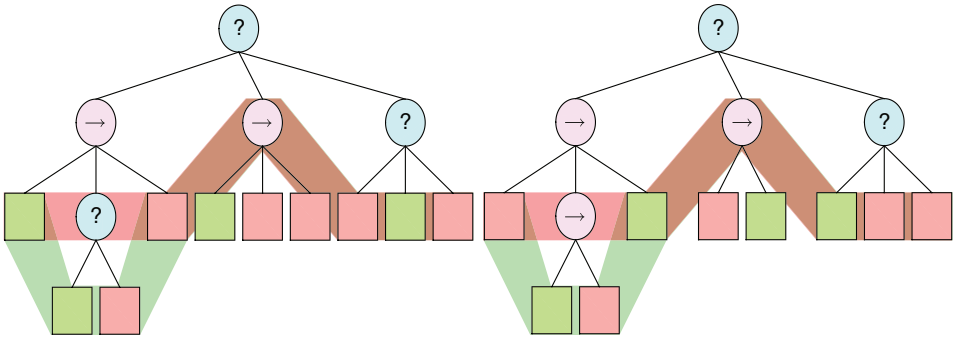
The different members selected for the tournament are compared with the *crowded comparison* operator. The crowded comparison operator compares two genomes based on whether one dominates the other. If neither of them is dominated, the genome with the higher crowding distance is selected.

Crossover

The crossover operator recombines the genetic material of the two parents to form two children. The crossover operator used in this thesis is the *uniform crossover operator* (Poli & Langdon, 1995), which exchanges genetic material between two genomes along the boundary of their common region.

The common region of the two genomes is determined based on their tree structure. In this work a loose definition of the common region is used, where the only requirements for the nodes at a specific location in the two trees to be considered common is that they have the same number of children. A more strict definition of the common region considers the node types as well. All nodes with parents in the common region belong to either the common- or boundary region, furthermore all leaf nodes with parents in the common region are part of the boundary between the two trees.

An illustration of the common region of two BTs is shown on Figure 5-2, highlighting the boundary region as defined in this work in green and the more stringent definition in red. The two BTs have a single composite, the root node's second child, in their boundary region as per the definition used in this thesis, since it has three children in the tree shown on Figure 5-2a, while having only two in the tree on Figure 5-2b. With the more restrictive definition,



(a) Boundary nodes of BT 1 **(b)** Boundary nodes of BT 2

Figure 5-2: Two BTs with the boundary of the common region highlighted between them. Green represents the definition used in this work, taking into account only the number of children the nodes have. Red represents the more strict definition, also considering the node types. The only composite node in the boundary region is the root's second child, as it has three children in tree (a) and only two in tree (b). The leaf nodes with parents in the common region are part of the boundary. With the strict definition, the Selector of the root's first child on (a) is also part of the boundary as it corresponds to a Sequence on (b).

the selector node under the root's first child shown on Figure 5-2a belongs to the boundary region as well, since it corresponds to a sequence node in the tree shown on Figure 5-2b. All the leaf nodes of which the parent belongs to the common region are part of the boundary.

Along the boundary region, all nodes are exchanged with a probability P_C. This allows the highest levels of sub-behaviors to be exchanged between the parents, resulting in the exchange of more genetic material in a single step, facilitating a faster convergence of the algorithm to the optimal solution (Poli & Langdon, 1995). A drawback of this operator, however, is that redundant nodes surrounding the ones encoding relevant sub-behaviors are also exchanged, resulting in unnecessarily complex trees that are difficult to simplify with this operator.

Another limitation of this operator can be observed when the boundary of the two tree's common region contains only the two root nodes. If this operator was applied in that case, it would reproduce the two parents, making the operation redundant. The only case when this would happen is if the roots of the two trees have different number of children. Crossover in such a case is implemented by considering the first m children of both tree roots to be the boundary region, where m corresponds to the smaller child count of the two root nodes.

Mutation

Mutation operators are intended to introduce small random changes to the genetic material of the genomes in order to exploit a local optima or introduce large variations into the genetic material, facilitating the exploration of the solution space. In this work two kinds of mutations are used. Micro-mutations alter the genetic material of the nodes already present in the tree, while Macro-mutations replace nodes/branches of the BT by randomly generated tree branches.

Micro-mutations of leaf nodes are implemented as a small, random change to the node's *constant* parameter, within $\pm 10\%$ of the valid parameter range. If the resulting node parameter

value lies outside of its valid range, the mutation is performed with the same magnitude in the other direction, corresponding to a sign change of the offset value. This facilitates the optimization of the node's performance by optimizing its parameters.

As composite nodes have no parameters and their children may represent complete sub-behaviors, any mutations that they undergo may result in significant changes in the resulting behavior. Nevertheless, following the definition used here, the micro-mutation of composite nodes is meant to represent changes in the composite node that do not completely redefine their operation. They are defined to be a random action of swapping two children, removing a child or adding a randomly generated sub-tree.

Some restrictions are applied when mutating composite nodes. No child can be removed from a composite when its number of children is less than three, as it would result in a composite with one, or no child at all. Furthermore, no child can be added to a composite if the number of its children already equals the maximum number of children allowed.

Macro-mutations are implemented by replacing a node of the tree by a randomly generated tree branch, also called Headless-Chicken Crossover (Angeline, 1997). This allows new genetic material to be introduced to the population.

All mutations are performed on a per-node basis, for each node of the tree. Macro-mutations are performed with a probability of P_{HCC}, while micro-mutations with a probability of P_m, if no macro-mutation was applied.

5-3 Cost Functions

One of the most influential factors on the results of an evolutionary process, and perhaps the least trivial one, is the proper definition of cost functions. A properly defined set of cost functions should fully describe *what* the end result is expected to do, rather than *how* it should do it. This principle is violated to a certain extent in this thesis as the task defined for the MAVs is manually appended to the evolved tree as its last node. Such a structure is enforced in order to enforce clear separation between the MAV's avoidance and task performance behavior. The remainder of this section discusses the cost functions used within this thesis.

The area covered with the last branch of the BT

As the MAV swarm's goal is to explore an area, this cost function provides a direct measure of how well they accomplish this task. On this cost function, the resulting behavior's performance is measured by dividing the virtual flight arena into 5×5 cm cells that are initially marked as not visited. The size of these cells was experimentally determined as to limit the achievable coverage when the MAVs perform redundant actions. At every time step of the simulation when an MAV ticks the last branch of its BT position of its centroid is marked on this coverage map as covered. The final value of this cost function is given by the ratio of visited cells and the total number of cells available.

Time spent using the last branch of the BT

The exploration strategy defined in this thesis is not ideal in the sense that the MAVs often end up visiting regions of the arena that have previously been already covered, resulting in no benefit for the MAVs if they use their task behavior when entering such regions. This would not necessarily be the case if the task behavior of the MAVs was implemented differently or they would be performing a different task altogether. This cost function is intended to measure how much the task behavior of the MAVs is used in a given trial. The base value of this cost function is determined by dividing the tick count of the task behavior by the tick count of the BT's root node.

Simulation Time

The primary goal of this thesis is the development of an avoidance behavior that is capable of preventing the MAVs from colliding with each other. Since a simulation trial is terminated as soon as a collision has occurred, this performance metric is a direct indication of how well the behavior performs in terms of preventing collisions. The value of this cost function is defined as the duration of a given trial divided by the maximum simulation time, T_{Sim}. While this is the only cost function directly rewarding the MAVs for avoiding collisions, the benefits of successful collision avoidance are also reflected in the first objective, since a longer trial typically can result in a higher area coverage.

The size of the BT

The difficulty for the human designer to understand the encoded behavior in a BT is proportional to its size. Because of this, a small BT that possesses the desired traits of the avoidance behavior is preferred over a larger BT that does a similar job. The value of this cost function is equal to 1 if the BT has less than 10 nodes, equal to 0 if it has 100 or more nodes and varies linearly between them. Note, that the appended branch containing the task behavior of the MAVs is not included when evaluating this cost function.

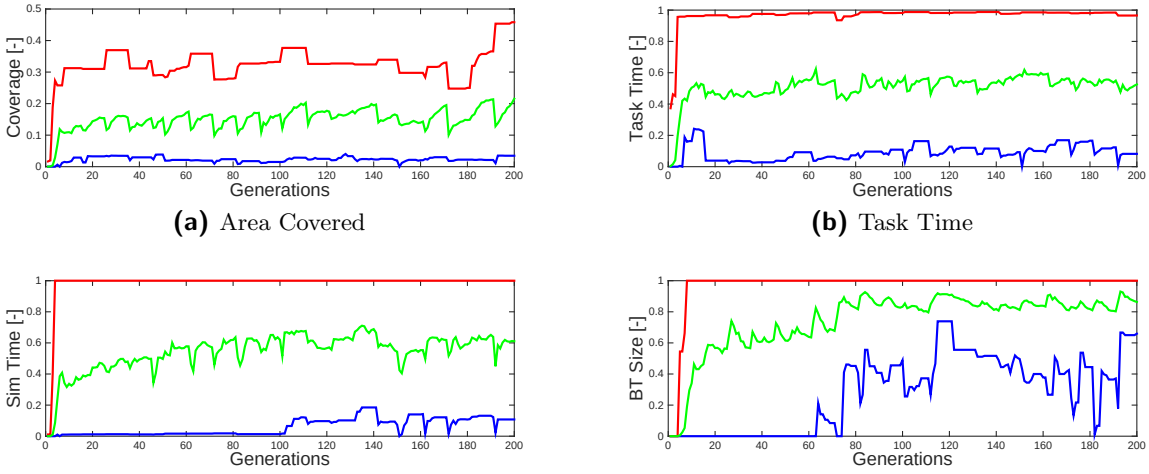
5-4 Evolution Parameters

Before the GA can be used, its parameters defined in this chapter need to be specified. These are given in Table 5-1 and were chosen by observing the GA's performance with different parameter combinations. The main driving parameters of the algorithm's convergence rate are the tournament size N_T, the crossover- and mutation probabilities, P_C, P_m and P_{HCC}. From these, perhaps the micro-mutation probability P_m is unconventionally high when the operation itself is performed on each node of the tree. Considering the implementation of this operator, where its main purpose is designed to be the tuning of the node parameters, together with the elitist approach of the NSGA-II algorithm, the value of 20% for this parameter was found to be suitable.

In the evaluation step of the GA, each individual is evaluated N_{Sim} times on different initial conditions in order to estimate their general performance, not tied to a certain initialization.

Table 5-1: The parameters of our evolutionary algorithm

	Parameter	Value	
N_{Gen}	Number of generations	200	[-]
N_{Pop}	Population size	100	[-]
N_{Child}	Maximum number of children	5	[-]
N_{Depth}	Maximum tree depth	5	[-]
N_T	Tournament size	10	[-]
N_{Sim}	Number of simulations	5	[-]
T_{Sim}	Simulation time	300	[s]
P_C	Crossover probability	70	[%]
P_m	Micro-mutation probability	20	[%]
P_{HCC}	Macro-mutation probability	0.1	[%]



(a) Area Covered **(b)** Task Time

(c) Simulation Time **(d)** Tree Size

Figure 5-3: The maximum (red), mean (green) and minimum (blue) values of the different cost functions at different generations. The valid range of all cost functions is between zero and one.

In order to facilitate the convergence of the GA, however, the set of initial conditions on which the individuals are evaluated are maintained across a number of generations. The initial conditions of the simulations are changed only if the number of domination fronts in the population drop below 10, or a certain set of initial conditions have been used 10 times.

5-5 Evolution Results

In this section the results of the evolutionary optimization are discussed. Figure 5-3 shows the progress of the cost functions discussed in Section 5-3 over the 200 generations. Perhaps the most curious thing to observe on these figures is the periodic rise and sharp drops of the population's mean and worst performing individuals. This is due to the algorithm starting to converge to a local optima, specific to certain initialization sets - an effect of maintaining the initializations across a number of generations.

On Figure 5-3 it can be seen, that in the first few generations the EA finds solutions that are

able to maximize all individual cost functions with the exception of the area coverage. These cost functions typically correspond to trivial solutions. An example behavior that can maximize both the cost functions concerning the simulation time and the BT's size is a tree with a single node that sets the flight velocity of the MAVs to zero. Alternatively, a BT maximizing the task time and BT size metrics is simply a tree containing only the task behavior of the MAVs. Both of these metrics maximize only two of the cost functions while they have close to zero performance on the other two metrics.

For all cost functions considered in this work, the mean typically lies close to half way between the best and worst performing individuals of the population, indicating that the population has a diverse set of solutions, representing a variety of trade-offs between the objectives. The only exception from this is the metric concerning the BT's size, which shows a steady convergence towards the population's maximum. This indicates that the algorithm steadily purges the nodes not useful for the behavior across the different generations. Another thing to notice on this cost function is that the population's mean converges to a value around 0.8 and remains there. This is due to the crossover operator used in this work, which is unable to remove nodes that are in the common region of two trees, making it increasingly more difficult to remove unnecessary nodes as the shape/structure of the trees become more similar.

Figure 5-4 shows the distribution of the last generation's members across the different objectives. On these figures it can be observed that the population is spread out in the solution space, representing a large variety in behaviors. The last generation of the evolution has 37 genomes out of the 100 on the Pareto front. The likely reason why not all solutions have converged to this front lies in the variety of approaches and the limited number of genome evaluations performed. All individuals of the population represent a different behavior that have, in a Pareto optimal sense, a better or worse performance than another one, depending on the initialization that it was evaluated on. Therefore, with a limited set of genome evaluations/initializations, the generalized performance of the individuals can not be perfectly measured.

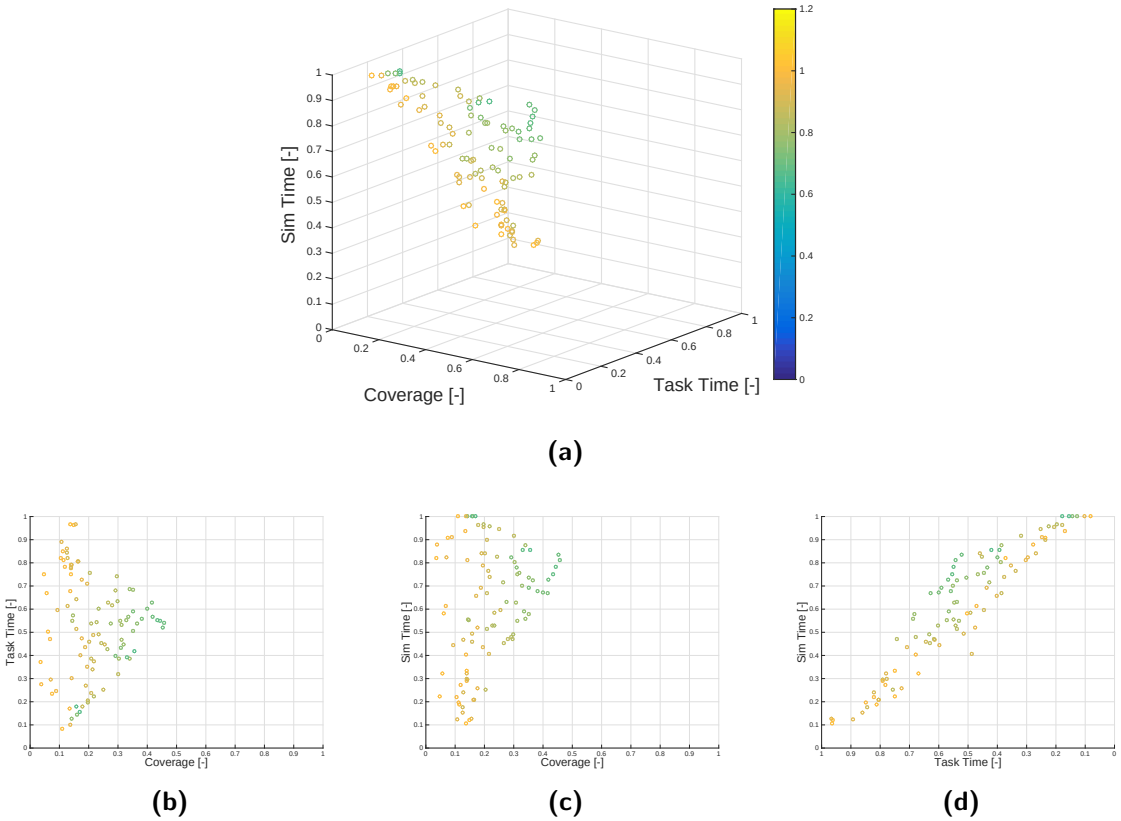


Figure 5-4: The population of the last generation in the solution space. **(a)**: the complete solution space. **(b)-(d)** projections of (a) onto its axis. The color axis corresponds to the last objective function, the BT size.

Chapter 6

Collision Avoidance Strategy

Two collision avoidance strategies relying on signal strength measurements have been designed within this thesis, which are discussed in this chapter. First, in Section 6-1 an avoidance strategy is presented, which can guarantee collision avoidance based solely only on distance estimates and their first derivatives. Such a strategy, however, can not be used based RSSI measurements, since the effects of the antenna gains make it impossible to meaningfully differentiate the resulting distance estimates. In Section 6-2 an alternative strategy is presented, capable of resolving many conflict scenarios, that was designed manually as a baseline to which the evolved controller's performance to be compared to. Finally, in Section 6-3 the strategy resulting from the EA is discussed.

6-1 A Collision Avoidance Strategy Based On Closure Rate

A difficulty with ensuring collision avoidance based purely on distance estimates lies in the fact that an approaching MAV may come from any direction in the three dimensional space. If two MAVs flying with the same speed V on a collision course, the conflict can be resolved if at least one of them reverses its flight direction, flying back along the way where it entered the conflict from, with a speed equal to V. A proof of this, considering a two dimensional environment, shown below:

Consider two MAVs, A and B, at locations $\vec{P}_{A_0}$ and $\vec{P}_{B_0}$ with constant velocity vectors $\vec{V_A}$ and $\vec{V_B}$ at time t_0, where the magnitude of the velocity vectors is equal, $|\vec{V_A}| = |\vec{V_B}|$. Let these conditions represent a collision course between them. Their positions and the relative position vector $D(t)$ between them, as a function of time are:

$$\vec{P}_A(t) = \vec{P}_{A_0} + \vec{V_A}t$$
$$\vec{P}_B(t) = \vec{P}_{B_0} + \vec{V_B}t$$
$$\vec{D}(t) = \vec{P}_A(t) - \vec{P}_B(t) = (\vec{P}_{A_0} - \vec{P}_{B_0}) + (\vec{V_A} - \vec{V_B})t$$

Let $d(t)$ be the magnitude of $\vec{D}(t)$, with P_{A_0x}, P_{A_0y}, P_{B_0x}, P_{B_0y}, V_{Ax}, V_{Ay}, V_{Bx} and V_{By} the x and y coordinates of the initial positions and velocity components of the MAVs:

$$d(t) = \sqrt{(D_x + V_x t)^2 + (D_y + V_y t)^2}$$

$$where:$$

$$D_x = P_{A_0x} - P_{B_0x}$$
$$D_y = P_{A_0y} - P_{B_0y}$$
$$V_x = V_{Ax} - V_{Bx}$$
$$V_y = V_{Ay} - V_{By}$$

Let t_c denote the time of the closest point of approach, where $d(t)$ has its minimum. This time instance corresponds to when $\dot{d}(t)$ is equal to zero:

$$\dot{d}(t_c) = \frac{V_x(D_x + V_x t_c) + V_y(D_y + V_y t_c)}{\sqrt{(D_x + V_x t_c)^2 + (D_y + V_y t_c)^2}} = 0$$
$$t_c = -\frac{D_x V_x + D_y V_y}{V_x^2 + V_y^2}$$

The distance between the MAVs at time t_c can be determined to be:

$$d(t_c) = \sqrt{\left(D_x - \frac{V_x(D_x V_x + D_y V_y)}{V_x^2 + V_y^2}\right)^2 + \left(D_y - \frac{V_y(D_x V_x + D_y V_y)}{V_x^2 + V_y^2}\right)^2}$$

In order for a collision between MAVs A and B to occur, the distance between them at time t_c must be less than the diameter of an MAV, indicated here with D. From this, it follows:

$$\sqrt{\left(D_x - \frac{V_x(D_x V_x + D_y V_y)}{V_x^2 + V_y^2}\right)^2 + \left(D_y - \frac{V_y(D_x V_x + D_y V_y)}{V_x^2 + V_y^2}\right)^2} \leq D$$

$$therefore:$$

$$D_x^2 + D_y^2 - \frac{(D_x V_x + D_y V_y)^2}{V_x^2 + V_y^2} \leq D^2$$

From this it follows, that:

$$\frac{V_x}{V_y} \leq \frac{D_x D_y \pm D\sqrt{D_x^2 + D_y^2 - D^2}}{D_y^2 - D^2}$$

Since the terms on the left side of the inequality are constant, and one of the solutions corresponds to a negative value for t_c, the following relationship between V_x and V_y can be determined:

$$V_x \leq C V_y$$

Substituting back the individual velocity vectors $\vec{V}_A$ and $\vec{V}_B$:

$$V_{Ax} - V_{Bx} \leq C(V_{Ay} - V_{By})$$

Now let MAV B reverse its velocity vector as $\vec{V}_{B1} = -\vec{V}_B$. In order for the MAVs to remain on a collision course, the following equations must be simultaneously true:

$$\begin{aligned} V_{Ax} - V_{Bx} &\leq C(V_{Ay} - V_{By}) \\ V_{Ax} - V_{B1x} &\leq C(V_{Ay} - V_{B1y}) \\ therefore: & \\ V_{Ax} + V_{Bx} &\leq C(V_{Ay} + V_{By}) \end{aligned}$$

This can only be true if:

$$\frac{V_{Ax}}{V_{Ay}} = \frac{V_{Bx}}{V_{By}}$$

Given that $|\vec{V}_A| = |\vec{V}_B|$, it follows that $\vec{V}_A = \vec{V}_B$. Therefore, if $\vec{P}_{A_0} \neq \vec{P}_{B_0}$, and $\vec{V}_A = \vec{V}_B$, the two MAVs could not have been initially on a collision course. Since this is a violation of the initial assumption, it follows that reversing the velocity of one MAV is sufficient to avoid the collision.

q.e.d.

From here, it can be seen that if at least one of the MAVs reverses its flight direction when a collision is about to occur, this collision will be prevented. This, however, does not resolve the conflict itself, since if both of MAVs reverse their flight direction, the distance between them will increase and when it has grown sufficiently large for them to start flying forward again, they would enter the same conflict as before. Therefore, if the MAVs perform a turning maneuver when they are at a sufficiently safe distance from each other, while still employing strategy of flying back whenever entering a conflict, the consecutive conflict scenarios will always different from each other. Repeatedly going through such a cycle will eventually resolving the conflict, at which point the MAVs are on a flight path that ensures collision free flight.

The avoidance behavior depicted on Figure 6-1 follows this principle. Here, $D3$ marks the distance at which the avoidance behavior should start. $D2$ marks the smallest distance until which the MAVs are required to fly straight backwards after a conflict, while $D1$ indicates the critical distance at which the MAVs should reverse their flight direction. The parameter T marks the turn rate at which the avoidance turn is to be performed and it has a direct influence on how fast a conflict is resolved.

The algorithm itself can be explained as follows. If two MAVs are within $D2$ and $D3$, they start turning to avoid a possible conflict. Depending on the relative angles at which the MAVs enter the conflict, such a turn might maintain them on a collision course - though there are many scenarios that can already be resolved by this turn. If the distance between the two MAVs falls below $D1$ and it is decreasing the MAVs reverse their flight direction and fly straight until they are at least at a distance of $D2$ from each other. This cycle is repeated until the conflict is resolved.

Since in order to prevent a collision it is enough if one of the MAVs reverses its flight direction, the proposed behavior can resolve multi-MAV conflicts as well, given that d and $\dot{d}$ always accurately track the *closest* MAV and the MAV dynamics is fast enough. Disregarding the turning behavior of the avoidance algorithm presented in Figure 6-1, a worst

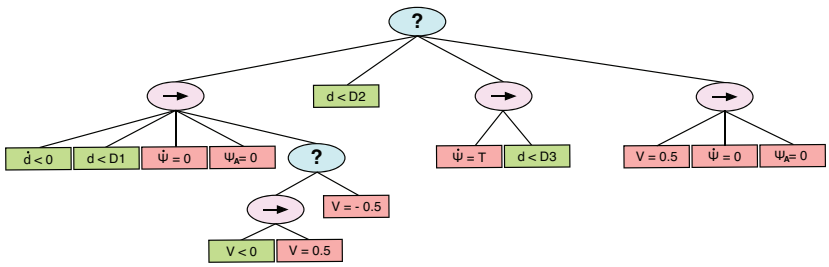


Figure 6-1: A manually designed avoidance behavior with a theoretical guarantee of collision avoidance in a conflict between two MAVs, given perfect distance measurements. This behavior can be extended to a conflict between multiple MAVs. Its inputs are: d - the distance to the closest MAV and $\dot{d}$ - the corresponding closure rate (first derivative of d). The outputs of the behavior are: V - the commanded velocity, $\dot{\Psi}$ - the commanded turn rate and Ψ_A - heading increment (unused). $D1$, $D2$, $D3$ and T are the algorithm's parameters, where $D1$ - threshold corresponding to a critical distance, $D2$ - the smallest distance considered to be safe, while $D3$ - desired minimum inter-MAV distance. T is the conflict resolution turn rate, which should be non-zero.

case scenario for a three MAV conflict is shown on Figure 6-2. This situation corresponds to all three MAVs flying along a one line - two of them in one direction, while the third one in the opposite direction.

In such a scenario, first the blue and green MAVs will get in conflict and start flying backwards. This, in turn, creates a conflict between the red and green MAVs, which results in the green MAV reversing its flight direction again, re-creating the conflict between the blue and green MAVs. Without the turning action this cycle will continue forever, but a collision will never occur. The turning action represented by the middle branch on Figure 6-1 can ensure that the consecutive conflicts of the green MAV will always differ, eventually resolving the conflict. This principle holds for any number of MAVs in conflict.

6-2 Manually Designed Avoidance Behavior

In order to solve the collision avoidance problem, the BT given in Figure 6-3 was developed. The parameters used are: d - represents the input of the BT, the inter-MAV distance, V - the commanded forward velocity in meters per second, $\dot{\Psi}$ - the commanded turn rate in degrees per second, Ψ_A - a heading increment in degrees, while AE is an internal parameter enabling/disabling the use of the heading increment.

The BT shown in Figure 6-3 contains 19 nodes and can be divided into two parts: the *avoid-*

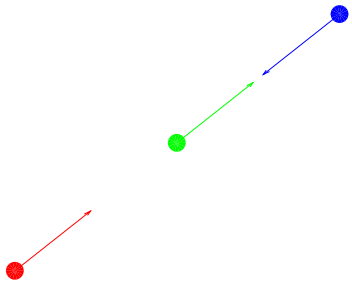


Figure 6-2: A worst case scenario for multiple MAVs, where the avoidance algorithm does not include a turning behavior. By reversing the flight direction of the green and blue MAVs when their closure rate is negative and the distance between them falls below a threshold $D1$, a collision between them is prevented. This results in a conflict between the green and red MAVs, which can be resolved in a similar manner as between the blue and green MAVs. Since the blue MAV will resume its forward flight when the conflict with the green MAV is resolved, the conflict resolution between the green and red MAVs will result in a new conflict between the green and blue one. The green MAV will repeatedly avoid the blue and red MAVs, while both of these avoid the green one. This cycle will continue and can only be resolved by introducing a turning behavior - though a collision will not occur even without such a turning behavior.

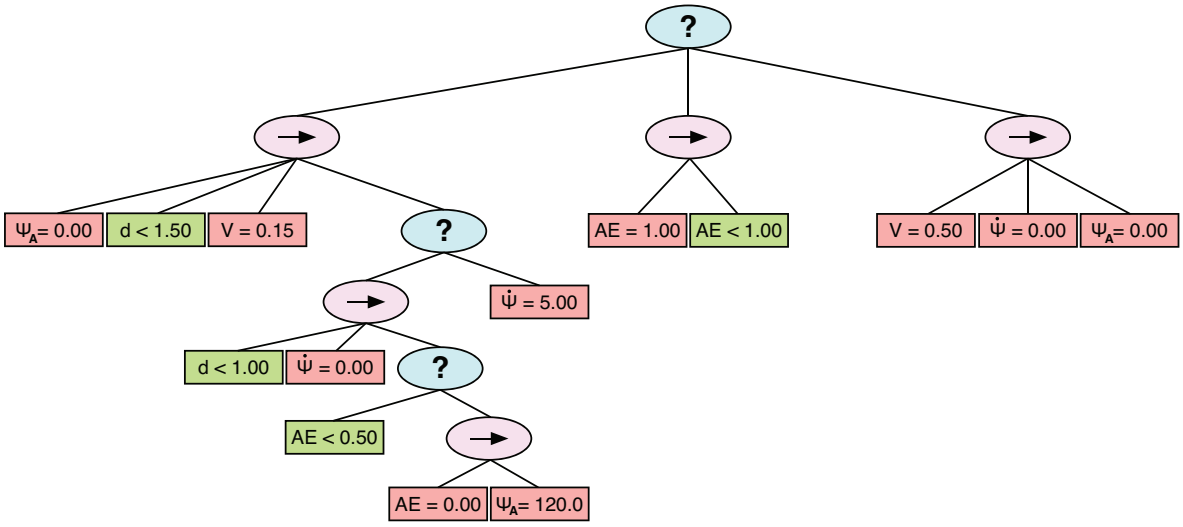
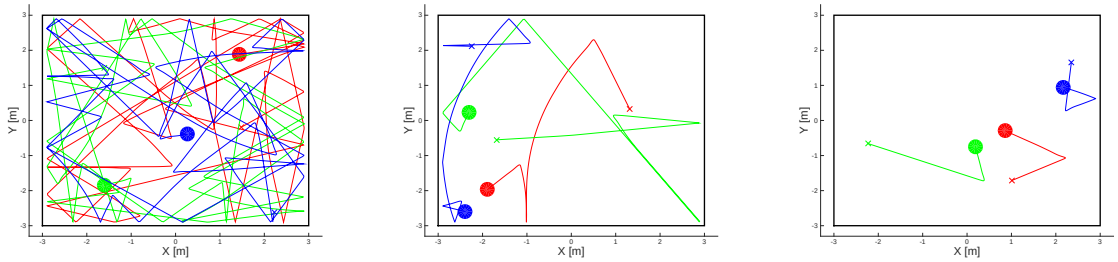


Figure 6-3: The manually developed avoidance behavior

ance behavior - represented by the left side of the tree, starting from the first Selector node under the root; and the *task behavior* - represented by the right side of the tree, under the root's second Selector node. The central Sequence is used to reset the internal parameter, re-enabling the avoidance turn, and it always returns false. The avoidance behavior itself can be explained, as follows. If two MAVs get close than $d = 1.5$ meters, they both slow down and perform a slow turn with $\dot{\Psi}$ of $5°/s$. If the distance between the MAVs drop below $d = 1$ meters, they both start flying straight at low speed and, if enabled, they perform their avoidance turn. If the distance between them is more than $d = 1.5$ meters, the MAVs enable their avoidance turn and enter their task behavior.

At first the internal state, AE, might seem as a factor that reduces the avoidance algorithm's



(a) Example of a successful trial

(b) Failure case due to arena bounds

(c) Failure case due to multiple consecutive conflicts

Figure 6-4: Examples of simulation trials with the manually designed avoidance behavior

performance. In fact, such an observation is correct as the use of this internal state is the root cause of the two most common failure modes of the algorithm. The use of AE, however, is made necessary by the noise present on the distance estimates. Because of this noise, when two MAVs get close to each other they would repeatedly perform their avoidance maneuvers as the distance estimate oscillates around 1 meters, if the number of these is not limited by the AE flag. This effect is further aggravated by the estimation biases resulting from the directional gains of the antennas, since as the MAVs perform their avoidance turn, the resulting relative angles are different, corresponding to different antenna gains.

The behavior presented here, though capable of resolving many conflicts, is not ideal. Figure 6-4 shows examples of three simulation trials, showing a successful trial where all conflicts are resolved in Figure 6-4a and the two trials representing the most common failure cases - Figures 6-4b and 6-4c.

Figure 6-4b shows the failure case of the algorithm when two MAVs avoid each other close to a wall, but due to one of them being forced to perform a turn, they end up colliding. Here, the red and blue MAVs successfully resolve a conflict, but the blue MAV is close to the wall of the arena and when it turns away from it - collides with the red MAV.

Figure 6-4c presents the failure case of the proposed behavior when there are multiple, consecutive conflicts. On this figure, initially the red and blue MAVs successfully avoid each other. Shortly after resolving this conflict, however, the green and red MAVs get into conflict with each other, but the AE flag of the red MAV is disabled, so it is not allowed to perform its avoidance maneuver, resulting in a collision.

6-3 Evolved Avoidance Behavior

An alternative avoidance behavior was provided by the evolutionary algorithm, as shown on Figure 6-5. Initially it contained 29 nodes, from which 19 were redundant and removed manually.

The behavior illustrated in Figure 6-5 can be explained as follows. If two MAVs are closer than $d = 1.37$ meters, they start flying slowly backwards while turning with $\dot{\Psi} = 9.55°/s$. The second child of the root node ensures that this behavior will not be changed as long as the

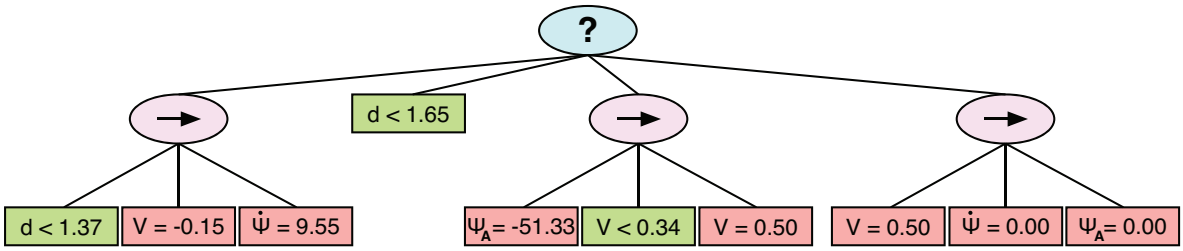
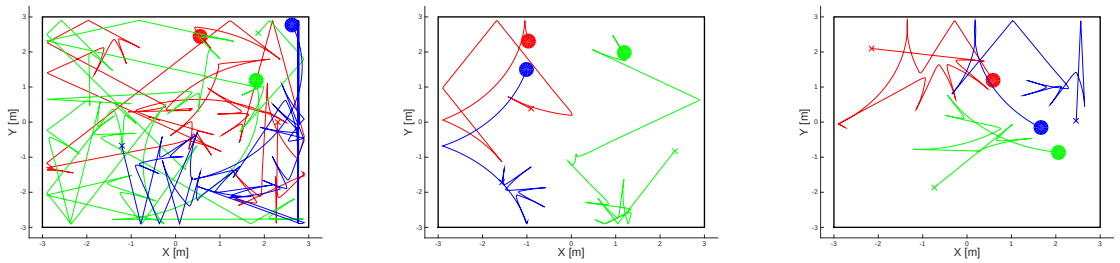


Figure 6-5: The evolved avoidance behavior



(a) Example of a successful trial

(b) Failure case due to arena bounds

(c) Failure case due to multiple consecutive conflicts, combined with wall avoidance

Figure 6-6: Examples of simulation trials with the evolved avoidance behavior

distance between the MAVs is less than $d = 1.65$ meters. If the distance between the MAVs is more than $d = 1.65$ meters and they were flying backwards, they make a one time turn of $\Psi_A = -51.33°$ and start flying straight, after which they enter their default task behavior.

This behavior is similar to the manually developed avoidance behavior in the sense that it relies on performing a discrete avoidance turn. The main difference between them is that the evolved controller includes a phase where the MAVs fly backwards, ensuring that when they perform their avoidance turn, they are sufficiently apart. This allows them to repeat their avoidance maneuver if the first one did not succeed, putting the MAVs on incrementally different flight paths as they repeatedly avoid each other. This can be clearly seen on Figure 6-6b between the blue and green MAVs and on Figure 6-6c between the red and blue MAVs.

Figure 6-6 shows examples of simulation experiments done with the evolved avoidance behavior. An example of a successful trial is shown on Figure 6-6a, while the two most common failure cases are shown on Figures 6-6b and 6-6c.

Figure 6-6b presents the failure mode of the avoidance behavior, where the wall avoidance behavior of the red MAVs turns it away from the northern wall of the arena, resulting in a crash with the blue one. Even though this crash was caused by the red MAV avoiding the northern wall, the avoidance behavior failed at the moment when the red and blue MAVs started flying alongside each other, close to the southern wall. Such a scenario was primarily caused by the blue MAV avoiding the green one, when the red MAV approached it from behind. Since their headings were nearly identical at the moment when they started their avoidance behavior, the resulting scenario will likely have caused an unresolvable conflict even

without a wall avoidance maneuver.

Figure 6-6c indicates a failure scenario of the behavior where the red and green MAVs are trying to avoid each other. At a certain point,the blue MAVs starts avoiding the red MAV, shortly after which the green one starts its avoidance maneuver with the red MAV. As the blue MAV reaches the northern wall of the arena, it is forced to turn around. At a certain point as both blue and green MAVs are avoiding the red one, they start avoiding each other while still flying in the same direction. This leads to an eventual collision between the blue and green MAVs.

Chapter 7

Results

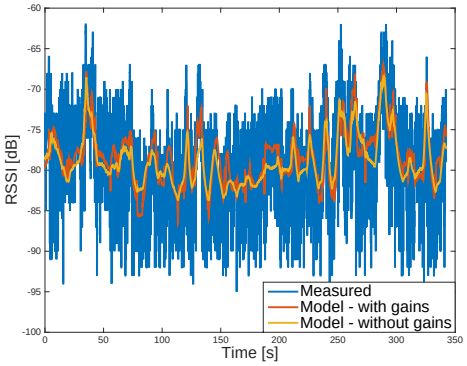
In this chapter the results obtained within this thesis are presented. First, In Section 7-1 the real-world results of the RSSI based distance estimation method are discussed. The collision avoidance and task performance results obtained both in simulation and real-world experiments are then discussed in Section 7-2.

7-1 RSSI Based Distance Estimation

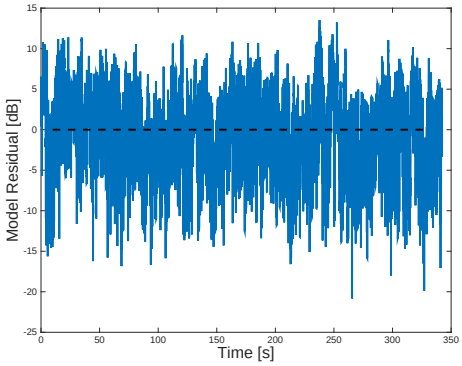
In this section, the predictions of the signal strength - distance model presented in Section 4-2-2 are compared to the behavior of the signal strength measured on-board. Furthermore, the real-world performance of the distance estimation method discussed in Section 4-3-1 is evaluated here.

Figure 7-1a shows the measured RSSI values from one of the datasets collected during the evaluation of the avoidance behaviors with two ARDrones. In order to investigate the Gaussianity of the model residuals, the model fits indicated on Figure 7-1a were calculated off-line and the parameters A and n of the sensor model given in Equation 4-2 were estimated from the respective dataset. Because of this, the mean of the model residual indicated on Figure 7-1b is zero. In reality, these model parameters were always estimated off-line from previously recorded datasets. As the parameters of the model are varying in time due to external factors, such as the intensity of which the 2.4GHz frequency band is used, the model residual of the on-line distance estimates was never exactly zero.

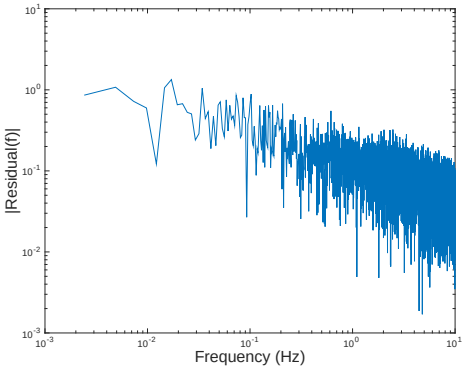
Looking at Figure 7-1b, showing the model error as a function of time, a certain periodicity can be observed. This is confirmed by the Fourier transform of the model residual, shown on Figure 7-1c, indicating that the frequency components below 1 Hz of the model residual have significantly more power than the ones above. The residual of the model developed in Section 4-2-2 was assumed to be zero-mean Gaussian noise, incorporating the effects of sensor noise, signal interference and the effects of multi-path propagation. The most likely source of the observed periodicity is due to the effects of multi-path propagation. As the ARDrones move with respect to each other, the transmitted signal is reflected by the objects



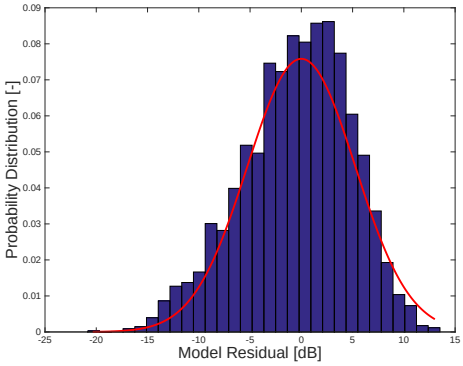
(a) Recorded and predicted RSSI values from one of the experimental datasets



(b) Residual of the model considering the antenna gain effects



(c) The Fourier transform of the model residual



(d) The estimated probability distribution of the model residual together with a zero-mean gaussian distribution with $\sigma = 5.26$

Figure 7-1: The recorded RSSI values and the off-line model predictions from an experiment performed with two ARDrones. The model parameters A and n of Equation 4-2 were estimated from the same dataset as indicated here, while the antenna gain model is the same as shown on Figure 4-3a. **(a)**: the measured RSSI values and off-line predictions with and without the antenna gains considered. **(b)**: model residual as a function of time with the model considering the antenna gains. The standard deviation is estimated to be 5.26. **(c)**: the Fourier transform of the model residual, showing its periodic nature. **(d)**: the probability distribution of the model residual.

in the environment, increasing- or decreasing the apparent signal strength, depending on the phase offset of the LOS and the reflected signal. The phase offset itself is a function of the distance between the transmitter and receiver antennas and their distance from the reflective surfaces. As the ARDrones move around with respect to these reflecting surfaces and each other, the phase offset between the LOS and reflected signals vary periodically, resulting in periodic variations in the apparent signal strength.

Comparing the model residual's Probability Density Function (PDF) to its Gaussian approximation on Figure 7-1d, two things can be observed. On one hand, the range of the residual is

$\approx [-21, +13]$ dB, while on the other hand, a probability mass concentration can be observed in the range of $0 - 4$ dB range. Both of these are likely caused by multi-path effects, which are not accurately captured by the Gaussian approximation. Depending on this phase offset between the LOS and reflected signals, the apparent signal strength varies between zero and twice the transmitted signal's strength. In logarithmic form, this variation is represented as a gain in the range $(-\infty, +3]$ dB. Assuming that the phase offset between the two signals is uniform in the range of $[-\pi, \pi]$, the PDF of the multi-path propagation has the same probability mass in the range of $[0, +3]$ dB as in the range of $(-\infty, 0)$. This explains the probability mass concentration above 0 dB observed on Figure 7-1d. Nevertheless, such an approximation seems to capture most of the relevant effects of interference and sensor noise. Furthermore, without the construction of an accurate (static) obstacle map and the knowledge of the exact motion of all transmitter/receiver pairs in the environment, the effects of multi-path propagation can not be exactly modeled.

The standard deviation of the Gaussian model fit shown on Figure 7-1d was estimated from the dataset to be $\sigma = 5.26$. This is a significantly higher value than the 3.2 implemented in the simulation model discussed in Section 4-2-2. The most likely reason for this increase in noise/interference power can be attributed to the increased used of the 2.4 GHz frequency band in the close environment of the flight arena at the time of the performed experiments than during the period when the senor model was identified.

Another likely source for the discrepancies between the constructed RSSI-distance model and the on-board measurements is that the antenna gain effects are only considered in two dimensions. Even though during the experiments all drones were commanded to fly at an altitude of 0.5 meters, due to aerodynamic effects, and non-ideal altitude control behavior, there was a constant variation in the altitude of the drones. This results in additional variations in the antenna gain effects due to the difference in altitude between the drones, especially at close distances, which are not properly modeled in the simulation environment.

Figure 7-2 shows the resulting distance estimates from the dataset considered in this analysis. The black line on the figure indicates the true distance between the ARDrones as given by the Optitrack system, the blue line indicates the distance estimate as obtained on-board, while the red line indicates an off-board distance estimate. In order to consider the effects of the Gaussian noise assumption made, the off-board estimate was obtained by first simulating the RSSI values based on the true distance between the ARDrones. The simulated RSSI values take into account the antenna gain effects and Gaussian noise with $\sigma = 5.26$ is added to it. The distances shown on Figure 7-2 are obtained by applying the same maximum-value prefilter and EKF to these simulated readings in an identical way as in the case of the on-board estimates. When considering the results indicated on this figure, it should be kept in mind, that the model parameters used for the on-line model estimates were obtained by a previous identification flight, therefore they are close, but not identical to the ones used for the off-line estimation.

The blue and red lines on Figure 7-2, corresponding to the on-board and off-board estimates respectively, show good correspondence for most cases. There are, however, multiple instances where the on-board estimate indicates significantly higher distances than both the true distance and the estimate based on the signal strength model. The most likely reason for these is the signal fading caused by multi-path propagation.

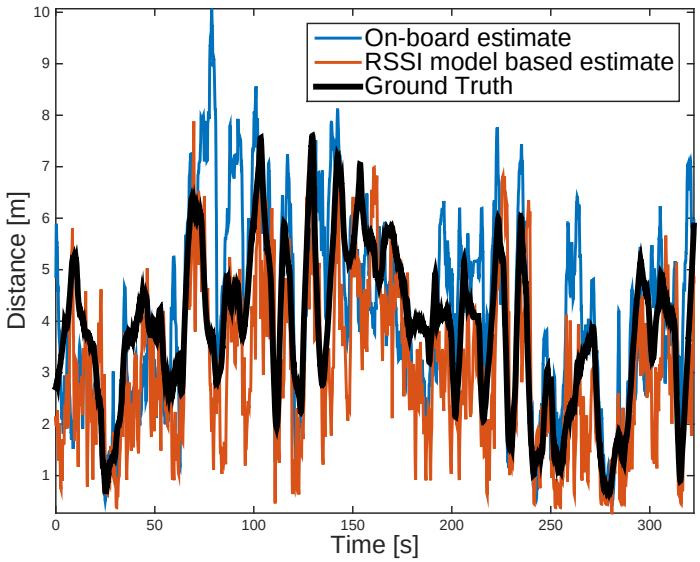


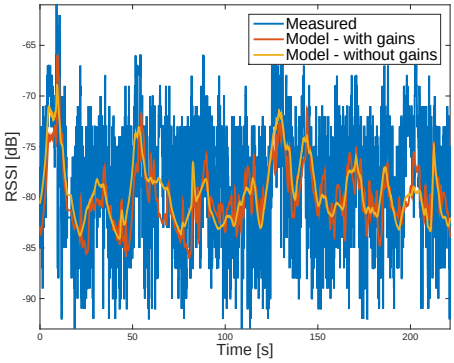
Figure 7-2: True distance between the ARDrones, on-board estimate and off-board estimate based on the sensor model's output

Due to physical space limitations of the flight arena, the parameters of the signal strength model were always identified from data collected with two ARDrones. Figure 7-3a shows the signal strength measured, and corresponding output of the signal strength model from an experiment with three ARDrones. The model parameters, A and n are the ones identified from the above discussed dataset, with two ARDrones. The dataset presented here was collected on the same day as the one shown on Figure 7-1, about three hours later and it indicates that the model parameters are both time varying and a function of the number of drones in the environment. On Figures 7-3b and 7-3c the model residual and corresponding PDF estimates are shown, indicating a strong model mismatch causing the distances to be severely underestimated. Because of this, before conducting the experiments, the model parameter A was adapted to reduce the severity of this underestimation based on flights with three ARDrones. This was also made necessary by the fact that throughout the day, both the magnitude of the interference and the model parameters, especially A, was observed to change. A likely explanation for these changes is the variation of the Wi-Fi traffic in the environment. The introduction of a third ARDrone in the environment, for example, results in an increase in Wi-Fi traffic due to an extra data link between the ground station and the ARDrone.

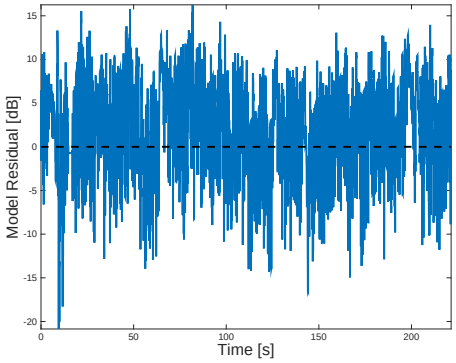
Looking at Figure 7-3c, the estimated PDF of the model residual is shown. The orange line indicates a zero-mean Gaussian model with $\sigma = 5.27$, as estimated from the model residual. The red line corresponds to the orange line shifted to the mean of the distribution. The important thing to note here is that the due to the introduction of another (moving) transmitter in the environment, the multi-path effects discussed above become more dominant in the model residual. A consequence of this is, that as more transmitters are introduced in the environment, the Gaussian noise assumption starts to become invalid.

Figure 7-3d shows the true distance between two of the three ARDrones, together with the on-board and off-board distance estimates. The off-board distance estimates shown here were

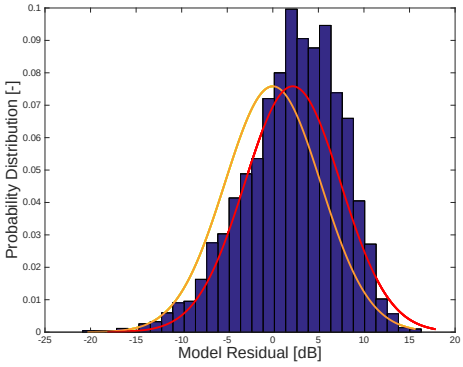
obtained from the recorded RSSI measurements, but the model parameters, A and n are maintained the same as estimated from the dataset obtained with two ARDrones. As can be seen on this figure, this results in a severe underestimation of the distances between the two ARDrones, requiring the adaptation of the model parameters as a function of the number of transmitters in the environment.



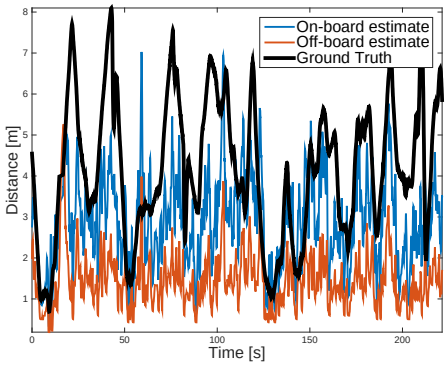
(a) Recorded and predicted RSSI values from one of the experimental datasets with three ARDrones. The model parameters are the same as used on Figure 7-1a



(b) Residual of the model, considering the antenna gain effects



(c) The estimated probability distribution of the model residual. The orange line indicates a zero-mean Gaussian distribution with $\sigma = 5.27$. The red line corresponds to the orange line being shifted to the PDF's mean.



(d) True and estimated inter-MAV distances. The off-board estimate is performed with the model parameters obtained from a dataset with two drones

Figure 7-3: The recorded RSSI values and the off-line model predictions from an experiment performed with three ARDrones. The model parameters were estimated from the dataset shown on Figure 7-1.

7-2 Performance of the Avoidance Behavior

Both of the avoidance behaviors proposed in Sections 6-2 and 6-3 were evaluated in a number of simulation and real-world experiments. In Section 7-2-1, the performance metrics used for evaluating the avoidance behaviors is discussed, while Sections 7-2-2 and 7-2-3 present the results of the simulation and real-world experiments respectively.

7-2-1 Performance Metrics

In the research question posed in Chapter 1, two important aspects of the avoidance behavior can be identified. On one hand, the avoidance behavior needs to ensure that the MAVs do not collide with each other, while on the other hand, it should minimize its interference on the task performance of the MAVs. Based on this, the following performance metrics are considered during the evaluation of the avoidance behaviors:

Mean Trial Time

An experimental trial is said to last until a collision between two MAVs occurs, or 300 seconds have passed. Therefore, the mean trial time is an indication of how long it takes for two MAVs to get into a conflict that they can not resolve.

Survival Probability of the MAVs

The survival probability of a trial for the MAVs is defined as the likelihood of a trial ending without a collision between two MAVs. This is a direct indication of the probability of the MAVs resolving all arising conflicts in a given trial.

Conflict Resolution Success Rate

The success rate of the conflict resolution is defined as the ratio of successfully resolved conflicts and the total number of encountered conflicts. This is an indication of how likely it is that given a conflict, it will be successfully resolved. It should, however, be noted that this is a somewhat counter-intuitive performance metric, as the failure of resolving one conflict out of 100 and 1000 results in a success rate of 99% versus 99.9%, not properly reflecting the fact that in the second case more than ten times as many conflicts were resolved.

Area Coverage

The task of the MAVs is defined to explore an unknown area, while avoiding collisions with the other members swarm. This metric gives an indication of how well the MAVs perform their task in terms of exploration. In a similar way as with the EA's cost function, the flight space available for the MAVs is divided into 5×5 cm grids, where each cell is marked as covered when the MAVs enter them, resulting in a high resolution coverage map.

MAV Task Time

Using this performance metric allows the impact of the avoidance behavior on the MAVs' task performance to be evaluated. It corresponds to average time the MAVs spend performing their task (ticking the last node of their BT) during a trial.

7-2-2 Simulation Results

Both avoidance behaviors designed within this thesis were developed with MAVs of 0.8 meters diameter, as this corresponds in size to the real-world setup used within this work. The approach of relying solely on RSSI measurements for avoiding collisions, however, is more beneficial for smaller MAVs with more restricted payload capabilities than the ones considered here. In order to evaluate the benefits provided by the proposed avoidance algorithms, the MAV's size was varied between 0.1 and 0.8 meters in the simulation experiments presented here. This allows the determination, on one hand, if employing an avoidance behavior is beneficial for the MAVs at all, while on the other hand, the performance of the proposed algorithms for smaller sized MAVs can be evaluated as well.

Another real-world restriction that was considered during the development of the avoidance algorithms was the physical size of the space where the resulting behaviors can be experimentally evaluated. As the maximum size of the usable flight space for evaluating the real-world performance of the MAVs is restricted to 6×6 meters, this was used as a baseline during the avoidance algorithms' development.

A distance of two meters or less between two MAVs is considered to be a conflict in this work. With this in mind, one can see that three MAVs in a 6×6 meter environment result in 12 m^2 of "conflict free" space for an MAV, whereas the area of a circle with a diameter of two meters is ≈ 12.6 m^2. Such a comparison, of course, is biased as it does not consider situations where part of the conflict region of an MAV lies outside the flight region - nevertheless it indicates that there is very little conflict free space for three MAVs in a 6×6 meter environment. Therefore, the avoidance algorithm's performance was evaluated on a range of arena sizes between 6 and 12 meters in side length in the simulation experiments.

All results presented in this section are based on 1000 simulation trials in the environment presented in Section 4-2. In these trials all MAVs were initialized at random initial locations and headings. The maximum simulation time was set to be 300 seconds, however a trial was stopped as soon as a collision has occurred. In order to compare the performance of the proposed avoidance behaviors, the performance metrics discussed in Section 7-2-1 were evaluated for MAVs without active collision avoidance as well. The performance metrics shown on Figures 7-4(a) - (c) are based on a simulation arena of 6×6 meters with varying MAVs sizes, while the ones shown on Figure 7-4(d) - (f) are based on an MAV diameter of 0.8 meters.

Considering the mean trial time shown on Figure 7-4a, the most important thing to notice is that the simulation time more than doubles when using either of the collision avoidance behaviors when compared to the MAVs not actively avoiding each other. The expected simulation time with the evolved avoidance algorithm remains above 130 seconds, more than 43% of the maximum simulation time for all MAV sizes. When compared to the performance of the manually designed avoidance algorithm, for smaller MAVs the performance is similar, however, the evolved behavior's superiority becomes more dominant as the MAV's size increases.

A similar trend can be observer on Figure 7-4b, where the probability of resolving all conflicts in a given trial are shown. The likelihood of a trial ending without a collision is about 57% with the evolved avoidance behavior for an MAV of 0.1m diameter, which drops to slightly below 10% for MAVs of 0.8m in diameter. With the manually developed avoidance behavior the probability of survival varies between about 55% for small MAVs, while it drops to below

3% for larger ones. Without any active collision avoidance, the probability of a small MAV surviving a trial is around 5%, which drops to ≈ 0 for larger MAVs.

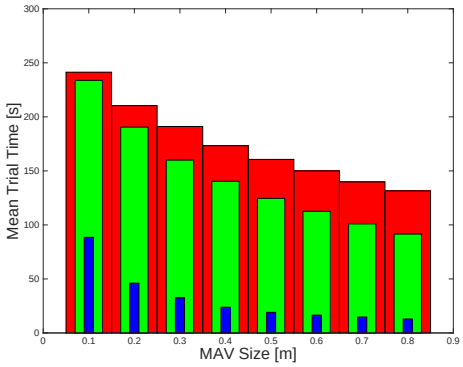
Figure 7-4c shows the probability of successfully resolving a conflict. The evolved behavior's success rate ranges between 88% to 96% for the different MAV sizes, resolving on average between 13 to 22 conflicts in each trial. For the manually designed behavior, this range is between 78.0% and 94.8%, resolving on average between 7 and 16 conflicts per trial. Without active collision avoidance, the MAVs resolve less than two conflicts per trial if their diameter is above 0.5m, which rises to slightly above 11 conflicts for an MAV of 0.1m diameter - values correspond to the MAVs not colliding purely by chance in a given trial.

The rate of mean area coverage is shown on Figure 7-4d. It should be noted, that the values indicated on this graph are meant only as a basis of comparison, since only the centroids of the MAVs are marked on the coverage maps - representing the MAVs as if their size was 5×5 cm. By looking at flight trajectories of the MAVs shown on Figures 6-4a and 6-6a, it can be observed that the MAVs have covered more than 25% of the available area, indicating that the coverage rates shown on Figure 7-4d are significant underestimates of the true area coverage.

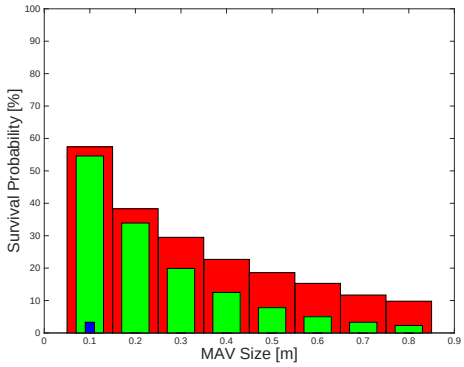
The most curious thing to observe on this figure is that the area coverage rate increases for a while, than it starts to drop. This is due to the quadratic increase in the number of cells to visit in a fixed maximum trial time. This figure indicates that for area sizes up to $\approx 9 - 10$ meters in side length, the increase in trial time due to the reduced number of conflicts increases the coverage rate sufficiently to compensate for the increase in area size. For areas larger than 10×10 meters, the maximum trial time of 300 seconds starts to dominate the results, reducing the coverage rate.

From Figure 7-4d it can also be observed that the evolved behavior covers more area compared to the manually designed one when more conflicts occur, but as the size of the area increases, the manually designed behavior starts to outperform the evolved one. The benefits of the evolved behavior, in terms of area coverage for smaller area sized, can be explained by the combination of increased trial time and the likely fact that an avoidance turn of 51° is more optimal than 120° for achieving higher area coverage.

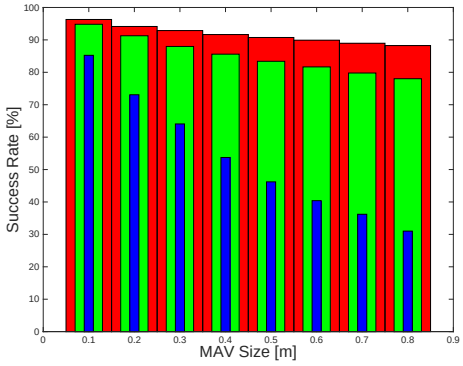
The average time spent by the MAVs performing their task is shown on Figure 7-4e. Comparing the task time of the MAVs using the manual and evolved behaviors, it can be seen that for smaller areas the MAVs using the evolved avoidance behavior spend more time performing their task, while the manually designed behavior has better performance for larger areas. This can be explained by the task time rate, as shown on Figure 7-4f, showing the ratio of time spent by the MAVs performing their task during a trial. From this figure it is clear, that the manually developed avoidance behavior has a higher task time rate, which is compensated for by the increased trial time on smaller area sizes. The difference in task time rate between the evolved and manually designed behaviors can be explained by the fact that the evolved behavior requires more time to resolve a conflict. While the manually designed behavior performs an avoidance turn as soon as the MAVs get too close to each other, with the evolved behavior the MAVs first have to fly back, and only then do they perform their avoidance turn and resume their task behavior. The task time of MAVs without active collision avoidance corresponds to their mean trial time, as their task rate is equal to 100%.



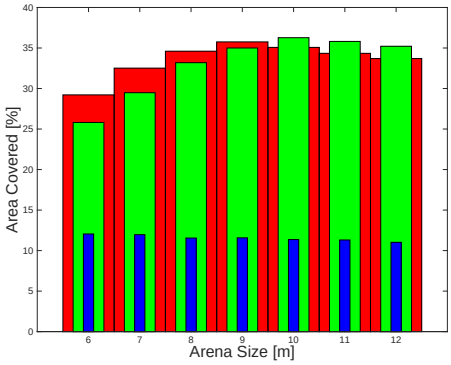
(a) The average time of a trial



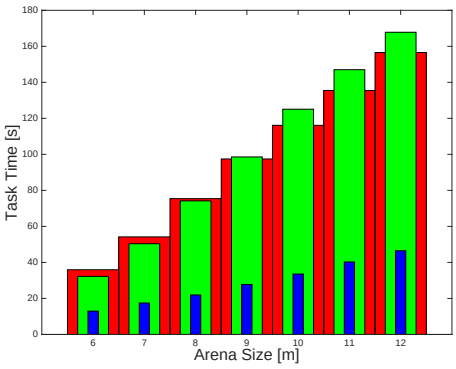
(b) The probability of resolving all conflicts in a trial



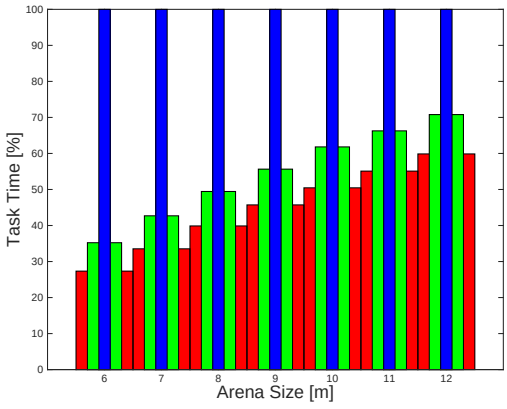
(c) The probability of successfully resolving a given conflict



(d) Area covered by the MAVs



(e) The average time the MAVs spend performing their task



(f) Time ratio in which the MAVs perform their task

Figure 7-4: Results of the simulation experiments. **Red** indicates the performance of the evolved behavior, **Green** the performance of the manually designed aboidance behavior, while **Blue** indicates performance without any active collision avoidance behavior

7-2-3 Real World Results

In order to evaluate the real-world performance of the proposed avoidance behaviors, a number of experimental trials were performed with two and three ARDrones respectively. Six datasets were collected per avoidance behavior with two ARDrones, while nine datasets per avoidance behavior with three. The experimental setup used in these experiments was discussed in Section 4-1. Only one of the experimental conditions considered in the simulation experiments was evaluated in real-life, corresponding to an arena size of 6×6 meters and an MAV size of 0.8 meters, which is approximately the diameter of an ARDrone2.

In order to protect the propellers when a collision occurs, the ARDrones were equipped with their indoor hoods. A consequence of using these hoods was that a typical collision between two drones resulted only in a gentle bump after which they typically resolved their conflict and flew in different directions. Because of this, the drones were allowed to continue flying until they either physically crashed or they ran out of battery.

A collision was defined in this thesis as the horizontal distance between two MAVs being less than or equal to their diameter. In order to maintain this definition, the collected datasets were split into different trials based on the time when two of the ARDrones collided. After a collision has occurred, the time period until all inter-MAVs distances were at least 2.5 meters was discarded, from which point the dataset was considered as a new experimental trial, corresponding to a different set of initial conditions.

During the experiments it has happened that an ARDrone flew under another one. In such a case, the Optitrack system could not always properly determine the exact position of the lower drone, providing erroneous position estimates, often at large distances from the drone's true position. As the situation was resolved, typically within a few seconds, the accuracy of the position tracking was restored. In order to eliminate these inaccurate position estimates, any trial that lasted less than 2.5 seconds was discarded. A trial time of 2.5 seconds corresponds to two ARDrones flying with their maximum allowed flight velocity of $0.5m/s$ towards each other from a distance of 2.5 meters. Therefore a trial lasting less than this is impossible and must be due to tracking errors.

In the analysis of the experimental results, all the metrics discussed in Section 7-2-1 were considered, with the exception of the survival probability as none of the experimental trials lasted more than 300 seconds. The results of the experiments are summarized in Table 7-1. These results are comparable to results of the simulation experiments in terms of the success rate of the avoidance behavior, while they also show significant deviation in terms of the average length of a trial, task time and area coverage.

The simulation results indicate a success rate of 88% and 78%, resolving 13.6 and 6.93 conflicts for the evolved and manually designed avoidance behaviors. The experimental results indicate a similar success rate for the evolved avoidance behaviors resolving, 13.8 conflicts. The success rate of the manually designed behavior is $\approx 8\%$ higher, resolving on average ≈ 6 more conflicts in these experiments than in the simulated ones.

This increase can be explained by fact that the drones often performed their avoidance turns more than once due to the high noise on the distance estimates. As the avoidance turn of the manually developed behavior is 120°, three consecutive avoidance turns in a short period

Table 7-1: Summary of results for the real-world experiments

	Trial Time	Success Rate	Task Time	Area Coverage (%)
2 ARDrones - Evolved	92.84 s	83.12 % - 9.85 conflicts	81.89 s (88.20 %)	10.73 %
2 ARDrones - Manual	86.17 s	87.67 % - 12.80 conflicts	71.74 s (83.25 %)	10.98 %
3 ARDrones - Evolved	38.00 s	87.62 % - 13.80 conflicts	24.70 s (65.01 %)	6.04 %
3 ARDrones - Manual	37.46 s	86.16 % - 13.05 conflicts	25.66 s (68.50 %)	6.13 %

result in the same, or similar, conflict scenario as before, with a high likelihood of successfully resolving the conflict. Furthermore, as sharp turns typically induce significant sway motion for the drones, which are then corrected for by the drone's controller. If two drones perform such a sharp turn while the distance between them is close to the defined conflict distance, this sway motion may result in exiting and re-entering the conflict zone multiple times, resulting in the conflict resolution being counted multiple times.

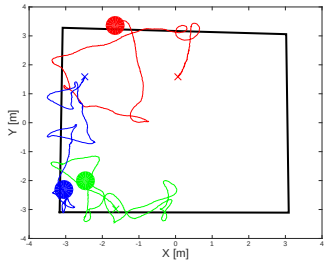
The mean trial time, therefore the task time and area coverage metrics as well, indicate a significantly worse performance than observed in simulations. Considering that the observed success rate of the avoidance behavior is similar, or higher, than observed in simulations, an increase in the number of conflicts can explain this drop in performance. This can be attributed to a significant reality gap between the experimental and simulation trials performed.

Examples of flight trajectories with the manually designed and evolved avoidance behaviors are shown on Figure 7-5. The on-board distance estimates corresponding to these flights are shown on Figure 7-6. Looking at the flight trajectories of the trials, a number of deviations from the trajectories shown on Figures 6-4 and 6-6 can be observed.

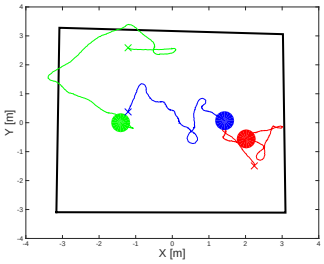
Clearly visible characteristic oscillations perpendicular to the general direction of motion of the drones can be observed on most flight trajectories shown on Figure 7-5, especially when the straight flight segments are short. These are caused by the sub-optimal tuning of the velocity controller used on-board of the drones. As a consequence of these oscillations, both the inter-MAV distances and relative angles vary. When such oscillations occur at small inter-MAVs distances, the antenna gain effects further emphasize the oscillations of the estimated distances.

Another important thing to notice on these flight trajectories is the observable sway motion during aggressive turn maneuvers of the drones. Examples of such occurrences are visible for the blue and red drones close to the center of the arena on Figure 7-5b, for the red drone turning away from the north wall on Figure 7-5c and for both the red and blue drones on Figure 7-5e, close to the arena's center. This behavior of the drones contributes to the observed reality gap in two ways. On one hand, due to the motion itself, the flight trajectories of the drones are different than observed in the simulations, while on the other hand, the also have an effect on the estimated distances - in a similar way as the observed lateral oscillations of the drones.

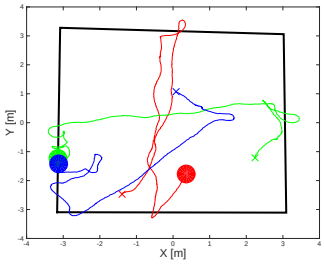
Another aspect considered significant based on the observations made during the experiments, is the effect of the propeller downwash on the neighboring drones. When two drones got close to each other, it was observed that they had an initial tendency to push each other away, while if the distance between them reduced below a certain point, they seemed to "attract" each other. A likely source of this is not necessarily the downwash of the propellers, but the



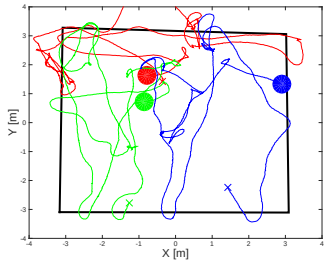
(a) The blue and green drones repeatedly avoid each other in a corner, until they collide



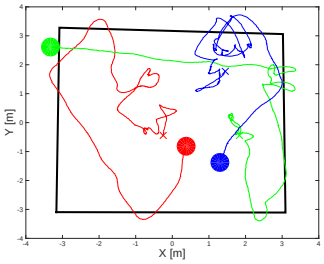
(b) The distances between the blue and red drones is severely underestimated, blocking their avoidance maneuvers



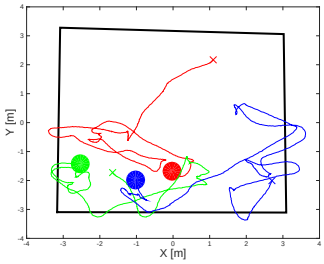
(c) Collision between the blue and green drones due to the green drone avoiding the arena bounds



(d) The red and green drones are stuck together in a similar way as shown on Figure 6-6b, resulting in a collision

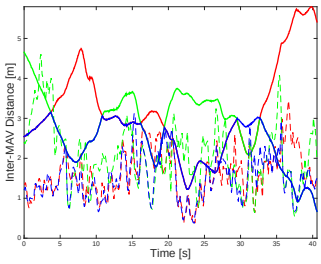


(e) The red and blue drones collide because their distances are overestimated - cf. Figure 7-6e

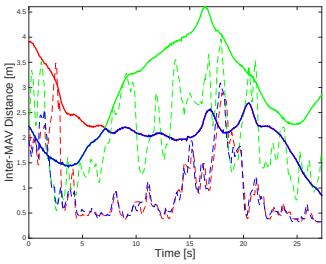


(f) All three drones are in a corner repeatedly avoiding each other until the blue drone turns away from a wall and collides with the red one.

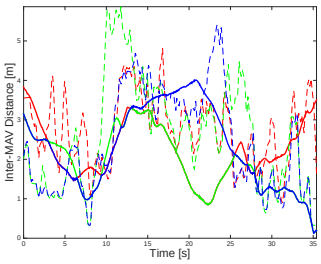
Figure 7-5: Example trajectories from the real-world experiments with three ARDrones. **(a)** - **(c)**: flight trajectories with the manually developed avoidance behavior. **(d)** - **(f)**: flight trajectories with the evolved avoidance behavior.



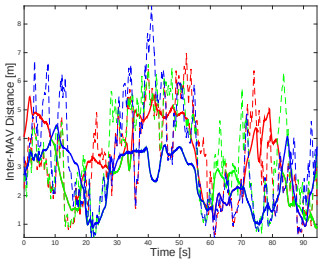
(a) Distance estimates corresponding to Figure 7-6b



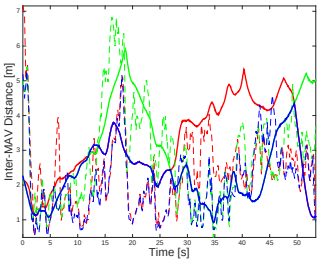
(b) Distance estimates corresponding to Figure 7-6b



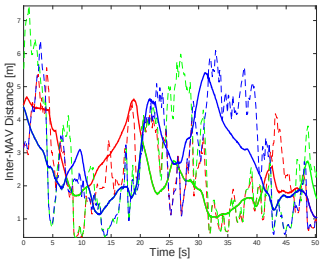
(c) Distance estimates corresponding to Figure 7-6c



(d) Distance estimates corresponding to Figure 7-6e



(e) Distance estimates corresponding to Figure 7-6e



(f) Distance estimates corresponding to Figure 7-6f

Figure 7-6: Distance estimates corresponding to the trajectories indicated on Figure 7-5.

upwash caused by it at a certain distance from the drones. Initially, as two drones approach each other, the upwash of the other drone increases the lift of the propellers closer together, tilting the drones away from each other as if there was a repulsive force between them. As the drones get closer to each other, the upwash increases the lift of the propellers further away, tilting the drones towards each other. These effects were quickly canceled out by the inner-loop controllers of the drones, their presence could, nevertheless be observed and they appeared at potentially critical times when the drones were close together.

Considering the distance estimates shown on Figure 7-6, significant oscillations can be observed, together with severe under- or over- estimation biases, causing a significant part of the observed collisions. As discussed in Section 7-1, the main cause behind the high variance of the estimates is the high and often non-Gaussian sensor noise. The non-Gaussianity of the noise term results in a decrease in performance for the EKF used to estimate the inter-MAVs distances. The high variance of the distance estimates often causes the drones to perform their avoidance turns consecutively a number of times, resulting in significantly different flight trajectories than observed in simulations.

Looking at the failures of the avoidance algorithms because of the distance estimation errors, it can be observed that the typically the failures are not caused by the variance fo the estimates, but the estimation biases. These biases were caused by the variations in the model parameters, especially the reference measurement A of the model given in Equation 4-2. During the preparations for of the experiment, it was observed that whenever the Wi-Fi traffic in the environment increased significantly, the value of A showed a corresponding increase in its value. This, in turn, results in the severe under estimation of the inter-MAVs distances if

the model parameters are not re-estimated and are the likely cause of the estimation errors shown on Figure 7-6b.

This observation is substantiated by the fact that the value of the model parameter A was observed to change depending on the time of the day when the experiments were performed. The experiments with the manual avoidance behavior were done in the late afternoon, when there were still people working in the neighborhood of the flight arena, while the experiments with the evolved behavior were does in the evening. The estimated value of the parameter A was -69.2 and -67.9 dB for the two experiments respectively, where the likely reason behind the different parameter values is the change in utilization of the 2.4 GHz frequency band.

Based on the discussion presented in this section, the some of most significant causes behind the observed reality gap can be identified as follows:

- The sub-optimal flight trajectory and velocity tracking of the drones
- The axis coupling observed during aggressive maneuvers that is not included in the simulation model
- The aerodynamics effects acting on the drones at close distances and low velocities
- Most importantly: the larger variance of the signal strength measurements and the variations in the model parameters

Chapter 8

Conclusions

The main goal of the work performed within this thesis was aimed at developing a successful collision avoidance behavior for a swarm of MAVs that rely solely on RSSI measurements, while at the same time, minimizing the impact of such behavior on the task performance of the MAVs. Of the two behaviors proposed within this thesis, one was the result of a genetic optimization, for which the simulation experiments indicate a better performance in terms of avoiding collisions, while at the same time having a more significant impact on the MAV's task performance.

The simulation results presented indicate that by using either of the proposed avoidance behaviors, it is possible to extend the flight time of the MAVs within a trial by more than 100% when compared to not having collision avoidance. Furthermore, the genetically evolved avoidance behavior is capable of resolving $\approx 88 - 96\%$ of the arising conflicts resulting in $\approx 10\%$ of the simulation trials ending without a collision. The manually designed avoidance behavior is capable of resolving $\approx 80 - 97\%$ of the arising conflicts, however the probability of a trial ending without a collision is only $\approx 5\%$ for MAVs of 0.8 meters in diameter.

In terms of task performance, the simulation experiments indicate that the evolved behavior allows the task behavior to be executed about $27 - 59\%$ of the time, depending on the size of the area. The manually designed avoidance behavior has a higher task rate, between $35 - 71\%$. This, in turn, results in the fact that for small areas, corresponding to higher MAVs densities, the evolved behavior results in better performance due to the increased trial time, while for larger areas the manually designed behavior has a higher task time.

The real experiments, performed with two and three ARDrones, show good correspondence with the simulation results in terms of the behavior's success rate. In terms of trial time, task time and area coverage, a significant drop in performance, corresponding to a factor of ≈ 3.5, could be observed for both avoidance behaviors proposed. This is an indication of a large reality gap between the simulation environment and the real flight environment.

The most important source of this reality gap was identified to be the significantly lower RSSI noise variance and the model parameters time invariance in the simulation environment. Both of these factors contribute to low quality of the distance estimates observed during the ex-

periments.In addition to this, a number of implementation related issues were identified, all of them contributing to this reality gap by increasing the number of conflicts when compared to the simulated environment.

The experiments performed highlight two main reasons why both of these avoidance behaviors have typical failure modes. By far the most common reason for the avoidance behavior's failure is the reliability of the distance estimates. In close, urban environments the 2.4GHz frequency band is heavily used by communication devices relying on Bluetooth and Wi-Fi transmissions. These transmissions, in turn, cause high levels of interference on the measured signal strength, resulting in poor estimate qualities.

Additional failure modes of the algorithms can be identified due to the lack of information available about the environment. As the MAVs try to avoid each other close to the arena's bounds, they often crash because they are forced to turn around without the avoidance behavior having any knowledge of it. In many of these scenarios, the conflict could have been resolved, did the MAVs not have to avoid the arena's bounds.

Based on the results obtained in this work, it can be concluded that in order to obtained a reliable and robust collision avoidance behavior, higher quality, consistent distance estimates are required. Better results than presented here can also be achieved by including mode information about the environment, such as the relative position of the static obstacles that the MAVs have to avoid. Furthermore, in this thesis it was shown that if the inter-MAVs distances can be meaningfully and reliably differentiated, a collision avoidance behavior, though not necessarily ideal in the sense of task performance, can be employed that guarantees collision avoidance between two MAVs.

Considering the research question of this work, it can be said that the probability of a collision with another MAV in a swarm can be reduced based solely on RSSI measurements, however the complete mitigation of such a risk is not possible without employing a trivial strategy, such as not moving at all, or only moving as long as the closest neighbor is indicated to be sufficiently far apart.

Finally, from the work presented in this thesis it can be concluded that the RSSI based avoidance behaviors presented in this thesis can *not* guarantee that the members of the swarm will not collide. It can, however, successfully resolve a number of conflicts in different scenarios, thereby increasing the flight time of the MAVs. Furthermore, it can achieve this with low computational costs and without *any* additional sensors required to be carried on-board. Because of this, such an avoidance behavior can be beneficial for small groups of MAVs with strict payload limitations, operating in a large environment. In such scenarios, employing either of the proposed avoidance behaviors can further reduce the probability of a collision, extending the flight time of the MAVs.

Chapter 9

Recommendations

Distance Estimation

Perhaps the most important recommendation for future research on the topic of intra-swarm collision avoidance is to improve the quality of the available distance estimates. While RSSI based distance estimation is an attractive choice for MAVs with strict payload limitation, it has severe limitations in terms of the quality of estimates that can be derived from it. Using different means to estimate the distances between MAVs, though likely requiring an additional sensor to be carried on-board, could greatly improve the performance of the collision avoidance behavior. In addition to that, it could provide more and/or higher level information, such as bearing or closure-rate estimates, which in turn would allow more efficient avoidance algorithms to be developed.

The quality of the RSSI based distance estimates could also be improved by utilizing a different frequency band from the 2.4 GHz. This would significantly reduce the interference on the measurements and would likely eliminate the time varying behavior of the model parameters. Furthermore, the exact noise model should be identified as a function of the number of drones in the environment. With an appropriate noise model available, a particle filter could provide significantly better distance estimates than the EKF used in this work.

Another benefit of using a particle filter instead of the EKF is that given additional information, such as a secondary RSSI measurement from a transmitter having a different relative angle, the effects of the antenna gains on the measurements could be reduced. Initial investigations into the topic indicate that by utilizing two transmitter-receivers pairs on the MAVs, if they fly with a fixed absolute heading, the relative bearing between them can also be obtained by tracking the antenna gain effects. An alternative method for reducing the effects of antenna gains, at least as long as the MAVs fly at the same altitude, is to use a dipole antenna as a transmitter/receiver, which has uniform gain perpendicular to its axis.

Simulation Reality Gap

During the real-world experiments a large reality gap was observed, of which a significant part can be attributed to the method in which the RSSI measurements were modeled. A number of additional sources were also identified to contribute to this reality gap, including the sub-optimal inner-loop controls of the ARDrones and the axis coupling during certain maneuvers. The magnitude of observed the reality gap can be further reduced by better tuning of the inner-loop controller's parameters and modeling the axis coupling in the simulation environment.

While the aerodynamic effects of the drone's propellers can not be considered large in terms of magnitude, the effect of these occurred at potentially critical times. Because of this, the occurrence of these effects should be either limited, by having the drones fly at higher altitudes, or should be included in the simulation model.

The Information Available for the Avoidance Algorithm

The simulation experiments indicated a number of failure modes of the avoidance algorithms that can be attributed to the lack of information about the environment. The most common failure mode observed in the simulation experiments was due to one of the MAVs turning away form the boundaries of the arena while avoiding the other MAV. The frequency of such collisions could be significantly reduced by providing the collision avoidance behavior information about the environment. The overall efficiency of the avoidance behavior could probably be increased even further by co-designing or co-evolving the obstacle- and collision-avoidance behaviors.

MAV Task Definition

The goal of this thesis was formulated in the form of *preventing* MAVs from colliding with each other while performing an arbitrary task. In an attempt to obtain a flexible, and possibly portable, avoidance behavior that can be used with an arbitrary task, the MAV's task was defined manually as exploring an area by flying straight and turning away from the arena's boundaries. Such an exploration behavior is sub-optimal in the sense that the MAVs often end up visiting the same region of the environment, therefore difficult to measure objectively during evolution. Therefore, the task of the MAVs should be defined in a way that can be clearly and objectively measured, without requiring multiple cost functions. This would result in a more controllable balance between task performance and collision avoidance for the EA.

Evolutionary Setup

Within this thesis there was limited emphasis put on investigating the effects of different genetic operators and optimization parameters on the performance of the EA. Using EAs for optimizing BTs is a fairly new field of research with limited scientific literature available. Because of this, more a in-depth investigation is required in order to determine what the optimal genetic operators for BTs are. For example, the crossover operator used in this thesis has a limited ability to remove unnecessary nodes from the tree as the size of the common region grows. On the other hand, it has beneficial traits in the sense that it exchanges complete sub-behaviors between the trees.

Appendix A

Literature Review on Relative Localization

In order to consider some alternative methods for relative position sensing, in this appendix a more in-depth review of the available relative localization techniques is presented. As the topic of this thesis the swarming of Micro Aerial Vehicles (MAVs), the sensors considered in this review are restricted to those with a sufficiently small weight and power requirements and with the potential for being installed on-board of MAVs.

The main localization principles considered in this review are as follows.

- **Time Of Flight (TOF)** or **Time Of Arrival (TOA)**: where the location of the signal's source is determined from the *absolute time* it takes the signal to reach different sensors
- **Time Difference Of Arrival (TDOA)**: where the direction and/or distance to the source is estimated from the *difference in time* of receiving the same signal at different sensors
- **Received Signal Strength (RSS)**: which relies on the comparison of the received *signal's strength* to a reference given by an attenuation model
- **Direct Bearing Measurements**: which rely on the measurement of the angle between the signal's source and the pointing orientation of the sensor
- **Contact Based Localization**: There are studies considering physical connections between ground robots as a means for formation control(Dorigo, 2005), where the angle between the connection point and the heading of the robot is used as the relative bearing, while the length of the connection joint is considered as the distance. In (Briod et al., 2013) a frame is constructed around a MAV that allows the determination of the bearing of an obstacle by means of measuring the contact force between the bounding frame and the object. This form of localization is only mentioned for the sake of completeness

and it is not considered further, as it is deemed impractical as a means for relative localization in a swarm of MAVs. The use of such techniques would either require the MAVs to be connected at all times, potentially destabilizing the MAVs, or it would provide information only upon contact.

A-1 Time of Arrival

The distance to the source of a signal can be estimated from the time it takes the signal to arrive, if the velocity of propagation is known. By using multiple sensors at different spacial locations, the source of a signal can be determined as the point of intersection of the imaginary spheres with origins at the different sensor's locations and radii proportional to the signal's time of flight. The difficulty of this localization method is that the receivers need to be aware of the time when the transmission was sent in order to calculate the time of flight of the signal. This issue is easily addressed when using RAdio Detection And Ranging (RADAR) or LIght Detection And Ranging (LIDAR) as distance sensors as the time of transmission is already available. For sensors where the signal's source is not the same as the signal's receiver, this issue is typically addressed by using Radio Frequency (RF) and acoustic transmissions simultaneously (Savvides, Han, & Strivastava, 2001; Rivard & Bisson, 2008; Shatara & Tan, 2010), and using the time when the RF transmission was received as the absolute start time of the acoustic transmission.

Savvides et al. (Savvides et al., 2001) proposed a system for determining the unknown location of a micro sensor in a static environment based on an ultrasonic pulse's the time of flight, received from multiple transmitters with known locations. In their system a radio network is available amongst the agents for communication and to indicate the ultrasonic pulse's transmission time. The location-aware nodes transmit their locations to the nodes in need of localization, allowing them to determine their own location based on TOA measurements and the know locations of the transmitting nodes.

Rivard and Bisson (Rivard & Bisson, 2008) used omni-directional ultrasonic pulses as a means for relative localization in a group of ground based robots. They used a single, omni-directional transmitter on each robot together with three omni-directional sensors. The sensors used are common ultrasonic transmitters and receivers, with a sound-reflecting cone suspended above them, allowing the transmission and reception of the pulses in an omni-directional fashion. Their system estimates the sound source's bearing from the closed-form solution of the intersection of three circles and estimate the distance as the average of the three independent estimate's average. They calculate the distance estimate as the average of the three measurements as they found it to be too sensitive to measurement noise when calculated from the closed-form solution. Furthermore, in order to eliminate erroneous measurements caused by reverberations from the environment, they suggest to use the fist appearance of the ultrasonic pulse as the time of flight and ignore the rest.

Shatara and Tan (Shatara & Tan, 2010) proposed an underwater ranging system based on TOA estimates for robotic "fish". The main difference between this work and the works in (Savvides et al., 2001; Rivard & Bisson, 2008) is that the ultrasonic pulse's behavior in an underwater environment required the predetermination of the signal's transient behavior and the amplitude that was to be expected. They propose an empirical, first order, model for the

signal's transient, and the use of two acoustic pulses: the first one to determine the signal's amplitude that is to be expected, while the second one for determining time of flight. Furthermore, in this work they use only a pair of sensors, allowing the determination of distances only. The bearing estimates are calculated from the readings of a compass.

In all of these works, the time synchronization required for estimating the time of flight of the acoustic signal was performed by sending a radio transmission simultaneously with the acoustic signal. Since the propagation velocity of the radio signal is several orders of magnitude larger than that of the acoustic waves, the transmission time of the radio signal can be assumed to be zero, indicating the time of transmission. A significant issue remaining is the estimation of the relative position from the distance estimates between the transmitter and the different sensors. While a closed-form solution for such problems exists, due to the distance measurement's noise, this provides inaccurate position estimates at best. Shen, Zetik, and Thoma (Shen et al., 2008) compare several common methods, like the Least Squares estimation, the Taylor Series method and the Maximum Likelihood method, for estimating relative position from TOA and TDOA measurements through simulations and gives recommendation for the appropriate algorithms to use in a number of different scenarios.

A-2 Time Delay of Arrival

If the time when a signal was transmitted is unknown, the Time Difference Of Arrival (TDOA) of the respective signal to the different receivers can be used. In order for the time difference to be measurable, the signal's propagation velocity should be moderately low, like in the case of the TOA based localization. From the inter-sensor time difference of receiving the same signal, a set of hyperboloids can be constructed and the intersection of these provides a location estimate for the transmitter. Two significant challenges are present when using TDOA based localization: the accurate estimation of the time delay between different sensor pairs, and the accurate localization based on these measurements.

The Generalized Cross-Correlation (GCC) (Knapp & Carter, 1976) is one of the most commonly used method for determining the time delay between a signal reaching two different sensors. In essence, the GCC estimates time delay from the different sensor's cross-power spectrum, multiplied by a filtering function $\Psi(f)$, via inverse Fourier transform. There are several variants of GCC, the difference between them being the filter applied. Sakhnov, Verteletskaya, and Simak (Sakhnov et al., 2011) presents several common choices for $\Psi(f)$ and their performance evaluations in Table 2 of Section 3.1 and Section 4 of their work respectively.

The determination of a signal source's location from TDOA estimates is difficult when the source is far away from the sensor array, as the signals reaching the different sensors are nearly parallel to each other. While assuming them to be parallel simplifies the estimation of the relative bearing of the source with respect to the sensor array, the estimation of the distance becomes impossible. Valin and Michaud (Valin & Michaud, 2003) use a ground based robot and a large microphone array to determine the bearing of a sound source in a noisy and reverberant room, however, since they assume the source to be in the far field, distance estimation between the robot and the sound source is not possible.

An alternative TDOA based localization is the Steered Response Power (SRP) (Badali, Valin,

Michaud, & Aarabi, 2009), which gives the likelihood of a signal coming from a certain location, given the sensor readings recorded. With such an algorithm, the signal source's location can be determined by searching through the location space for the most likely position. Basiri et al. (Basiri et al., 2012) use SRP with a particle filter to locate the source of an emergency whistle from a MAV, while in (Basiri & Schill, 2013) they use the same algorithm for determining the relative positions in a group of three MAVs, however the estimation of the relative positions was performed off-line based on the recorded data. Several alternative methods to the particle filter proposed in (Basiri et al., 2012) are compared by Badali et al. (Badali et al., 2009).

A-3 Received Signal Strength

As an alternative to the time it takes a signal to reach a receiver, the signal's attenuation can be used to estimate the distance that it has traveled. Received Signal Strength (RSS) based localization relies on deriving distance estimates by comparing the strength of the received signal to a baseline given by an attenuation model. Algorithms based on this principle were proposed (Chen & Kobayashi, 2002; Hightower & Vakili, 2001; Bahl & Padmanabhan, 2000) for locating users of wireless devices in an indoor environment with transponders at known locations, in a similar fashion as Savvides et al. (Savvides et al., 2001) did with TOA measurements. For inter-robot relative localization in an unknown environment, however, such transponders with known locations are not available. Instead, the directionality of sensors can be exploited to determine the location of a signal source.

Pugh, Raemy, Favre, Falconi, and Martinoli (Pugh et al., 2009) propose the use of Infrared (IR) LEDs and sensors for determining the relative positions in a group of ground-based robots by exploiting the IR sensor's directional sensitivity. They model the sensor's directionality as the product of signal's strength at the receiver and the incidence angle's cosine: $r_{measured} = r_{received} cos(\Theta)$. Combining such measurements from different sensors, the transmitter's location can be determined as the intersection point of the otherwise ambiguous bearing measurements. In order to eliminate the interference of naturally occurring IR sources, they modulated the IR LED's output and encoded local messages in them, considering the measurement invalid if the message was not received.

Roberts et al. (Roberts et al., 2012) propose a relative position sensor for MAVs based on the same principle by placing a ring of IR LEDs and sensors around the MAV. In addition to the relative localization provided by such a sensor, they used the reflections of the MAV's own transmission as a way to detect obstacles in the path of the MAV. The main difference between the method proposed by Pugh et al. and the one proposed by Roberts et al. is that Roberts et al. used a set of cascaded filters and amplifiers on the IR sensor readings to reduce the effects of measurement noise, while Pugh et al. used the RSS measurements of the sensors directly.

Raharijaona, Mignon, Juston, and Viollet (Raharijaona et al., 2013) present sensor for estimating the relative bearing of a flickering IR source with tiny sensor of 1 cm^3 and 0.33 g by placing IR receivers on the side of a small cube and exploiting their directional sensitivity in a similar fashion as in (Pugh et al., 2009; Roberts et al., 2012). They can estimate the relative azimuth and elevation of the IR source with minimal weight and power consumption,

however the sensor proposed by them is not capable of estimating distances.

A-4 Direct Bearing Measurements

There are different technologies available for directly measuring the bearing towards a signal's source, such as directional antennas or Very high frequency Omni-directional Range (VOR) beacons or camera images. As most of these technologies, like an array of directional antennas or a VOR beacon are large both in terms of size and weight, only computer vision is considered as a feasible direct bearing estimator for MAVs.

Technological advancements both on the software and hardware sides have enabled robots and MAVs to access visual information and autonomously perform things like Simultaneous Localization and Mapping (SLAM) (Heng, Meier, Tanskanen, Fraundorfer, & Pollefeys, 2011) based on vision, however the mass and power requirements of such hardware are still a challenge for small scale MAVs.

Cutler et al. (Cutler et al., 2013) proposed a method for self-localization with respect to a set of static IR markers. They propose using the off-the-shelf, small sized, light weight camera of a Wii remote for locating the IR markers in the environment. The image processing is done by the camera itself, which returns the relative locations of the markers in two-dimensional space at a high rate. The algorithm proposed uses this information together with the geometry of the target to be tracked to estimate the MAV's relative position and attitude with respect to the markers. By placing the markers on another MAV, the proposed method could allow the estimation of the relative location between MAVs.

Faigl et al. (Faigl et al., 2013) developed a relative localization algorithm for estimating distances between a small camera and a specific, elliptical marker. They have shown that a refresh rate of at least 7 Hz is possible with accurate position estimation for up to five meters, however, in its current state, the algorithm proposed supports the detection and tracking of a single target only. Furthermore, the algorithm requires a rather large, elliptical marker to be placed on the object to be tracked. The localization performance of the algorithm depends on the size of the marker and its distance from the camera.

Das, Fierro, and Kumar (Das et al., 2002) used a VGA camera with a reflector cone to achieve omni-directional vision in two dimension on a ground based robot. The knowledge of the reflector's size, shape and relative position with respect to the sensor allow the mapping of each pixel of the sensor to a unique location in the observable environment. The works presented in (Das et al., 2002) and (Fierro et al., 2002) mainly focus the control architecture that they developed for multi-robot coordination, however the proposed sensor tackles the problem of directionality (in 2D space), one of the drawbacks of visual localization.

A-5 Comparison of Methods

In order to assess the suitability of the different sensor types for localization on a MAV, in this section the localization methods presented in Sections A-1 - A-4 are compared based on the following criteria: Dimensionality & Directionality, Accuracy, Power, Mass and Update rate.

All sensing methods considered are suitable for three dimensional localization, however TDOA based methods require computationally intensive search through the possible location-space to obtain a distance estimate(Basiri & Schill, 2013). Furthermore, there is a trade-off to be made if visual localization is applied between dimensionality and directionality of localization. Three dimensional but directive localization(Heng et al., 2011; Pestana, Sanchez-Lopez, Saripalli, & Campoy, 2014; Faigl et al., 2013; Cutler et al., 2013) *or* omni-directional(Das et al., 2002) in two dimensions is achievable with vision.

Due to the fact that most of the sensors presented in this review were not specifically designed for MAVs, there is little information about their mass. The reviewed sensors that were proposed for MAVs cover a large range, from 0.33 g(Raharijaona et al., 2013) to 245 g(Roberts et al., 2012). The power required by the sensors lies between 1.5 - 2.6 W with the exception of the sensor proposed by Roberts et al., which requires 10 W.

The considered sensor's distance estimate accuracy are within 0.3 - 5% of their calibrated range, while the bearing accuracy is typically in the range of 5°. TOA and Vision based sensors have better distance estimates, while the ones based on RSS have better bearing accuracy. The (calibrated) sensing range of the reviewed sensors is typically under 6 meters, except for the TDOA based sensor proposed by Basiri and Schill and the underwater TOA sensor of Shatara and Tan, having ranges of 150 and 10 meters respectively. The latter of these, however, is measured underwater, where the propagation of sound has different characteristics than in air.

All proposed sensors have an update rate higher than 10 Hz, with the exception of the sensors proposed by Shatara and Tan, Faigl et al. and Rivard and Bisson, having update rates of 0.5 Hz, 7 Hz and 13.33/# Hz respectively. An update rate of 10 Hz is considered sufficient for the control of a MAV swarm, thereby most of the sensors considered in this review are suitable. It should be noted, that the update rate of 7 Hz for the algorithm proposed by Faigl et al. is a worst-case scenario.

Appendix B

The Behavior Tree Implementation

A BT is a rooted tree structure consisting of *composite*, *condition* and *action* nodes. When a node is *ticked*, i.e. triggered or evaluated, it performs its action and returns either *success*, *failure* or a *running* state. The flow through the BT is defined by the composite nodes, which evaluate their children based on their return states. The I/O parameters of the BT are passed to the different nodes via a Blackboard, called *BTWorkspace*, which can also be extended to contain internal parameters. The values contained within the BTWorkspace, with the exception of the input variables, are maintained across different ticks unless changed by an action node. This way the BTWorkspace serves as the memory of the BT.

The structure of the BT can saved and loaded as an XML document. Furthermore, in order to enable the easy implementation of the BT on the Paparazzi flight platform, a C code generation functionality has been implemented, which allows the automatic generation of a C header and source file that defines the structure of the BT. The definition of the node behavior itself is defined manually in separate C header and source files. This approach allows different BTs to be easily evaluated on the real flight platform.

Within this thesis two types of composite nodes, *Sequence* and *Selector*, one condition and one action node has been implemented. The BTWorkspace and the different node's behavior are described in more detail below and summarized in Table B-1.

BTWorkspace

The BTWorkspace serves both as the interface between the BT and the controller module and as the BT's memory. In fact, the BTWorkspace can be represented as a vector of parameters, where the first n values represent the BT's n inputs, the next m elements contain the m output parameters and the remaining parameters represent any internal state that the BT uses. The BTWorkspace also provides an access interface to the contained data via its a number of operators used by the different node types. This allows the workspace to restrict which parameters can be read or overwritten by the different nodes.

BTSelect

The Selector, or BTSelect, node evaluates its children until one of them succeeds or returns a running state, while the tick of a selector node fails when none of its children succeed. Typically the root node of a BT is a Selector node. In terms of standard operator types, it is comparable to an *or* operator, attempting to successfully tick one of its children. The node itself has no parameters and the behavior it represents is fully defined by its children.

BTSequence

The Sequence, or BTSequence, node evaluates its children until one of them fails or returns a running state. A Sequence node fails when one of its children fail. In terms of operator types, a selector is comparable to an *and* operator, succeeding only if all of its children succeed. The selector node itself has no parameters either, with its behavior entirely defined by the behavior of its children.

BTConditionC

The BTConditionC is the only condition node used within this work, which is used to compare the value of a BTWorkspace variable to a constant. The node's behavior is defined by its three parameters: *index* - the index of the BTWorkspace variable to be read, *value* - the constant value that the parameters is compared to and *type* - which defines if the parameter is to be larger or smaller than the constant. The BTConditionC node is allowed to perform comparisons on any of the workspace variables, including not only the inputs of the BT but also the currently set outputs and internal parameters. If the comparison is true, the node returns success, while if it's false, the node fails. This node never returns a running state.

BTSetC

The BTSetC is the only action node used within this thesis. It is used to set the value of any parameter in the workspace, with the exception of the input variables, equal to a constant. Its behavior is defined by two parameters: *index* - the index of the workspace parameter to be set, and *value* - the constant to set the parameter equal to. When ticked, this node always returns success.

Other Node Types Implemented

A number of other node types have been implemented, which are not used within this thesis. They are only mentioned here for the sake of completeness:

BTConditionP: Compares two workspace parameters.

BTSetP: Copies a workspace value to another location.

BTProdP: Sets a workspace variable equal to the product of two workspace variables

BTSetProportional : Linear transformation of a workspace parameter.

BTDelay : Returns a running state for a predetermined number of BT ticks..

BTWhile: Composite node which acts as a BTSequence, but returns a running state instead of failure.

Table B-1: The implemented Behavior Tree Nodes. The parameters type, index and constant refer to: the comparison type (more or less than), index of the variable in the workspace and constant to compare/set to respectively

Node Name	Type	Return State	Returned If	Parameters
BTSelect	Composite	Success Running Failure	One child succeeds A child is running All children failed	N/A
BTSequence	Composite	Success Running Failure	All children succeed A child is running One children failed	N/A
BTConditionC	Condition	Success Running Failure	The condition is true Never The condition is false	type, index, constant
BTSetC	Action	Success Running Failure	Always Never Never	index, constant

Bibliography

Angeline, P. (1997). Subtree crossover: Building block engine or macromutation. *Genetic Programming*. Retrieved from http://ncra.ucd.ie/COMP41190/SubtreeXoverBuildingBlockorMacromutation.ps

Auerbach, J. E., & Bongard, J. C. (2012). On the relationship between environmental and morphological complexity in evolved robots. *Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference - GECCO '12*, 521. Retrieved from http://dl.acm.org/citation.cfm?doid=2330163.2330238 doi: 10.1145/2330163.2330238

Badali, A., Valin, J.-M., Michaud, F., & Aarabi, P. (2009, October). Evaluating real-time audio localization algorithms for artificial audition in robotics. *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2033–2038. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5354308 doi: 10.1109/IROS.2009.5354308

Bahl, P., & Padmanabhan, V. (2000). RADAR: An in-building RF-based user location and tracking system. *INFOCOM 2000. Nineteenth ...*, *00*(c).

Basiri, M., & Schill, F. (2013). Audio-based Relative Positioning System for Multiple Micro Air Vehicle Systems. *...: Science and Systems*(266470). Retrieved from http://www.roboticsproceedings.org/rss09/p02.pdf

Basiri, M., Schill, F., Lima, P. U., & Floreano, D. (2012, October). Robust acoustic source localization of emergency signals from Micro Air Vehicles. *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 4737–4742. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6385608 doi: 10.1109/IROS.2012.6385608

Bluegiga Technologies. (2014). *Range Analysis - Appilcation Note* (Tech. Rep.).

Bongard, J. C. (2013, August). Evolutionary Robotics. *Communications of the ACM*, *56*(8), 74–83. Retrieved from http://dl.acm.org/citation.cfm?doid=2492007.2493883 http://www.lania.mx/ ccoello/EMOO/sbalzarini00.pdf.gz doi: 10.1145/2492007.2493883

Brambilla, M., Ferrante, E., Birattari, M., & Dorigo, M. (2013, January). Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, *7*(1), 1–41. Retrieved from http://link.springer.com/10.1007/s11721-012-0075-2 doi: 10.1007/s11721-012-0075-2

Briod, A., Kornatowski, P., Klaptocz, A., Garnier, A., Pagnamenta, M., Zufferey, J.-C., & Floreano, D. (2013, November). Contact-based navigation for an autonomous flying robot. *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 3987–3992. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6696926 doi: 10.1109/IROS.2013.6696926

Brisset, P., Drouin, A., Gorraz, M., Huard, P.-s., & Tyler, J. (2006). The Paparazzi Solution. In *Mav2006* (pp. 1–15). Retrieved from http://download.paparazziuav.org/papers/papers_2006/mav06_paparazzi.pdf

Brooks, R. a. (1990). Elephants Don ' t Play Chess. *Robotics and Autonomous Systems*, *6*, 3–15. doi: 10.1016/S0921-8890(05)80025-9

Champandard, A. (2007). Behavior trees for next-gen game AI. Game Developers Conference Europe 07.

Chen, Y., & Kobayashi, H. (2002). Signal strength based indoor geolocation. *2002 IEEE International Conference on Communications. Conference Proceedings. ICC 2002 (Cat. No.02CH37333)*, *1*(1), 436–439. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=996891 http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=996891 doi: 10.1109/ICC.2002.996891

Colledanchise, M., Marzinotto, A., & Ogren, P. (2014, May). Performance analysis of stochastic behavior trees. *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 3265–3272. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6907328 doi: 10.1109/ICRA.2014.6907328

Cutler, M., Michini, B., & How, J. P. (2013, May). Lightweight infrared sensing for relative navigation of quadrotors. *2013 International Conference on Unmanned Aircraft Systems (ICUAS)*, 1156–1164. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6564807 doi: 10.1109/ICUAS.2013.6564807

Dantu, K., Kate, B., Waterman, J., Bailis, P., & Welsh, M. (2011). Programming micro-aerial vehicle swarms with karma. *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems - SenSys '11*, 121. Retrieved from http://dl.acm.org/citation.cfm?doid=2070942.2070956 doi: 10.1145/2070942.2070956

Das, A., Fierro, R., & Kumar, V. (2002). A vision-based formation control framework. *Robotics and . . .*, *18*(5), 813–825.

Deb, K., Pratap, a., Agarwal, S., & Meyarivan, T. (2002, April). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182–197. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=996017 doi: 10.1109/4235.996017

Dorigo, M. (2005). Swarm-bot: An experiment in swarm robotics. *Swarm Intelligence Symposium, 2005. SIS 2005. . . .*, *2005*, 199–207.

Dorigo, M., Trianni, V., Sahin, E., Gross, R., Labella, T. H., Baldassarre, G., ... Gambardella, L. M. (2004). Evolving Self-Organizing behaviors for a Swarm-bot. *Autonomous Robot*, *17*(2-3), 223–245.

Dorigo, M., Tuci, E., Trianni, V., Gross, R., Nouyan, S., Ampatzis, C., ... Mondada, F. (2006). SWARM-BOT: Design and implementation of colonies of self-assembling robots. In *In computational intelligence: Principles and practice* (pp. 103–136). Retrieved from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.145.9758

Dromey, R. (2003). From requirements to design: formalizing the key steps. *First International Conference onSoftware Engineering and Formal Methods, 2003.Proceedings.*. doi: 10.1109/SEFM.2003.1236202

Dunkley, O., Engel, J., & Cremers, D. (2014). Visual-Inertial Navigation for a Camera-Equipped 25 g Nano-Quadrotor. *IROS 2014 Aerial Open Source Robotics Work- shop*, 4–5.

Faigl, J., Krajnik, T., & Chudoba, J. (2013). Low-cost embedded system for relative localization in robotic swarms. *Robotics and ...*. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6630694

Fierro, R., Das, a., Spletzer, J., Esposito, J., Kumar, V., Ostrowski, J. P., ... Southall, B. (2002, October). A Framework and Architecture for Multi-Robot Coordination. *The International Journal of Robotics Research*, *21*(10-11), 977–995. Retrieved from http://ijr.sagepub.com/cgi/doi/10.1177/027836490202101098 1 doi: 10.1177/027836490202101098 1

Floreano, D., Dürr, P., & Mattiussi, C. (2008, January). Neuroevolution: from architectures to learning. *Evolutionary Intelligence*, *1*(1), 47–62. Retrieved from http://link.springer.com/10.1007/s12065-007-0002-4 doi: 10.1007/s12065-007-0002-4

Forster, C., Pizzoli, M., & Scaramuzza, D. (2014). SVO : Fast Semi-Direct Monocular Visual Odometry. *IEEE International Conference on Robotics and Automation (ICRA)*. doi: 10.1109/ICRA.2014.6906584

Francesca, G., Brambilla, M., Brutschy, A., Trianni, V., & Birattari, M. (2014). AutoMoDe: A novel approach to the automatic design of control software for robot swarms. *Swarm Intelligence*, *8*(2), 89–112. doi: 10.1007/s11721-014-0092-4

Hauert, S., Winkler, L., Zufferey, J.-C., & Floreano, D. (2008, August). Ant-based swarming with positionless micro air vehicles for communication relay. *Swarm Intelligence*, *2*(2-4), 167–188. Retrieved from http://link.springer.com/10.1007/s11721-008-0013-5 doi: 10.1007/s11721-008-0013-5

Hauert, S., Zufferey, J.-C., & Floreano, D. (2008, October). Evolved swarming without positioning information: an application in aerial communication relay. *Autonomous Robots*, *26*(1), 21–32. Retrieved from http://link.springer.com/10.1007/s10514-008-9104-9 doi: 10.1007/s10514-008-9104-9

Heng, L., Meier, L., Tanskanen, P., Fraundorfer, F., & Pollefeys, M. (2011, May). Autonomous obstacle avoidance and maneuvering on a vision-guided MAV using on-board processing. *2011 IEEE International Conference on Robotics and Automation*, 2472–2477. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5980095 doi: 10.1109/ICRA.2011.5980095

Hightower, J., & Vakili, C. (2001). Design and calibration of the spoton ad-hoc location sensing system. *unpublished, August*.

Julier, S. (2002). The scaled unscented transformation. *Proceedings of the 2002 American Control Conference (IEEE Cat. No.CH37301)*, *6*(2), 4555–4559. doi: 10.1109/ACC.2002.1025369

Julier, S., & Uhlmann, J. (2004). Unscented Filtering and Non Linear Estimation. *Proceedings of the IEEE*, *92*(3), 401–422.

K. Ma, P. Chirarattananon, S. Fuller, R. (2013). Controlled flight of a biologically inspired, insect-scale robot. *Science*.

Kalman, R. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME, Journal of Basic Engineering*, *82*(1), 35–45.

Knapp, C., & Carter, G. (1976). The generalized correlation method for estimation of time delay. *Acoustics, Speech and Signal ...*, *2*, 320–327. Retrieved from `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1162830`

Koza, J. (1994). Genetic programming as a means for programming computers by natural selection. *Statistics and Computing*. Retrieved from `http://link.springer.com/article/10.1007/BF00175355`

Krajník, T., Vonásek, V., Fišer, D., & Faigl, J. (2011). AR-drone as a platform for robotic research and education. *Communications in Computer and Information Science*, *161 CCIS*, 172–186. doi: 10.1007/978-3-642-21975-7_16

Kushleyev, A., Mellinger, D., Powers, C., & Kumar, V. (2013). Towards a swarm of agile micro quadrotors. *Autonomous Robots*, *35*, 287–300. doi: 10.1007/s10514-013-9349-9

Ludwig, L., & Gini, M. (2006). Robotic swarm dispersion using wireless intensity signals. *Distributed Autonomous Robotic Systems 7*, 135–144. doi: 10.1007/4-431-35881-1-14

Maniezzo, V. (1994, January). Genetic evolution of the topology and weight distribution of neural networks. *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council*, *5*(1), 39–53. Retrieved from `http://www.ncbi.nlm.nih.gov/pubmed/18267778` doi: 10.1109/72.265959

Marzinotto, A., Colledanchise, M., Smith, C., & Ogren, P. (2014, April). Towards a Unified Behavior Trees Framework for Robot Control. *Robotics and Automation ...*, *57*(4), 345–370. Retrieved from `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6907656` doi: 10.1109/ICRA.2014.6907656

Miller, B., & Goldberg, D. (1995). Genetic algorithms, tournament selection, and the effects of noise. *Complex Systems*, *9*, 193–212. Retrieved from `http://www.complex-systems.com/pdf/09-3-2.pdf`

Nolfi, S., & Floreano, D. (2000). *Evolutionary robotics*. Mit Press Cambridge.

Ogren, P. (2012). Increasing Modularity of UAV Control Systems using Computer Game Behavior Trees. *AIAA Guidance, Navigation, and Control Conference*, 1–8. Retrieved from `http://arc.aiaa.org/doi/abs/10.2514/6.2012-4458` doi: 10.2514/6.2012-4458

Parrot, S. (2015). *AR.Drone 2.0 Parrot new wi-fi quadricopter.* Retrieved 2015-05-19, from `http://ardrone2.parrot.com`

Perez, D., & Nicolau, M. (2011). Reactiveness and navigation in computer games: Different needs, different approaches. *... Games (CIG), 2011 IEEE ...*. Retrieved from `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6032017`

Pestana, J., Sanchez-Lopez, J. L., Saripalli, S., & Campoy, P. (2014, June). Computer vision based general object following for GPS-denied multirotor unmanned vehicles. *2014 American Control Conference*, 1886–1891. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6858831 doi: 10.1109/ACC.2014.6858831

Petrovi, P. (2008). Evolving Behavior Coordination for Mobile Robots using Distributed Finite-State Automata. *Frontiers in Evolutionary Robotics*(April), 413–438.

Pintér-Bartha, A., Sobe, A., & Elmenreich, W. (2012). Towards the Light - Comparing Evolved Neural Network Controllers and Finite State Machine Controllers. *10th International Workshop on Intellegent Solutions in Embedded Systems*, *0*, 83–87. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6273587

Poli, R., & Langdon, W. (1995). On the Search Properties of Different Crossover Operators in Genetic Programming. Retrieved from http://discovery.ucl.ac.uk/1327770/

Pugh, J., Raemy, X., Favre, C., Falconi, R., & Martinoli, a. (2009, April). A Fast Onboard Relative Positioning Module for Multirobot Systems. *IEEE/ASME Transactions on Mechatronics*, *14*(2), 151–162. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4787109 doi: 10.1109/TMECH.2008.2011810

Raharijaona, T., Mignon, P., Juston, R., & Viollet, S. (2013). A tiny lensless position sensing device for the tracking of active markers. *REDUAS-2nd IFAC ...*, *2013*. Retrieved from http://hal.inria.fr/UNIV-AMU/hal-00910804

Remes, B., Hensen, D., Tienen, F. V., Wagter, C. D., Horst, E. V. D., & Croon, G. D. (2013). Paparazzi : how to make a swarm of Parrot AR Drones fly autonomously based on GPS . (September), 17–20.

Rivard, F., & Bisson, J. (2008, May). Ultrasonic relative positioning for multirobot systems. *... and Automation, 2008. ...*, 323–328. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4543228 http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4543228 doi: 10.1109/ROBOT.2008.4543228

Roberts, J. F., Stirling, T., Zufferey, J.-C., & Floreano, D. (2012, February). 3-D relative positioning sensor for indoor flying robots. *Autonomous Robots*, *33*(1-2), 5–20. Retrieved from http://link.springer.com/10.1007/s10514-012-9277-0 doi: 10.1007/s10514-012-9277-0

Sahin, E. (2005). Swarm robotics: From sources of inspiration to domains of application. *Swarm robotics*, 10–20. Retrieved from http://link.springer.com/chapter/10.1007/978-3-540-30552-1_2

Sakhnov, K., Verteletskaya, E., & Simak, B. (2011). Echo Delay Estimation Using Algorithms Based on Cross-correlation. *Journal of Convergence Information Technology*, *6*(4), 1–11. doi: 10.4156/jcit.vol6.

Savvides, A., Han, C.-C., & Strivastava, M. B. (2001). Dynamic fine-grained localization in Ad-Hoc networks of sensors. *Proceedings of the 7th annual international conference on Mobile computing and networking - MobiCom '01*, 166–179. Retrieved from http://portal.acm.org/citation.cfm?doid=381677.381693 doi: 10.1145/381677.381693

Sbalzarini, I., Müller, S., & Koumoutsakos, P. (2000). Multiobjective optimization using evolutionary algorithms. *Proceedings of the summer ....* Retrieved from http://www.lania.mx/ ccoello/EMOO/sbalzarini00.pdf.gz

Scheper, K. (2014). *Behaviour Trees for Evolutionary Robotics.* Unpublished master's thesis, Delft University of Technology, the Netherlands. Retrieved from http://repository.tudelft.nl/assets/uuid:dde8d42e-590a-465d-abaf-760ec304760f/kschepe

Seybold, J. S. (2005). *Introduction to RF Propagation.* doi: 10.1002/0471743690

Shatara, S., & Tan, X. (2010). An efficient, time-of-flight-based underwater acoustic ranging system for small robotic fish. *Oceanic Engineering, IEEE Journal of*, *35*(4), 837–846. Retrieved from http://faculty.nps.edu/pcchu/web_paper/ieee/ieee_mancini.pdf http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5604645

Shen, G., Zetik, R., & Thoma, R. (2008). Performance comparison of TOA and TDOA based location estimation algorithms in LOS environment. *Positioning, Navigation and ...*, *2008*, 71–78. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4510359

Stirling, T., Roberts, J., Zufferey, J.-C., & Floreano, D. (2012, May). Indoor navigation with a swarm of flying robots. *2012 IEEE International Conference on Robotics and Automation*, 4641–4647. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6224987 doi: 10.1109/ICRA.2012.6224987

Thrun, S. (2002). Particle Filters in Robotics. *Proceedings of Uncertainty in AI*, *1*, 511–518. Retrieved from http://uai.sis.pitt.edu/papers/02/p511-thrun.pdf

Ugur, E., Turgut, A. E., & Sahin, E. (2007). Dispersion of a swarm of robots based on realistic wireless intensity signals. *22nd International Symposium on Computer and Information Sciences, ISCIS 2007 - Proceedings*, 419–424. doi: 10.1109/ISCIS.2007.4456899

Uzol, O., Yavrucuk, I., & Sezer-Uzol, N. (2008). Collaborative target tracking for swarming MAVs using potential fields and panel methods. *Proceedings of the AIAA Guidance, ...*(August), 1–8. Retrieved from http://arc.aiaa.org/doi/pdf/10.2514/6.2008-7167

Valin, J., & Michaud, F. (2003). Robust sound source localization using a microphone array on a mobile robot. *Intelligent Robots and ...*, *0*(October), 1228–1233. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1248813

Valmari, A. (1998). The State Explosion Problem. *Lectures on petri nets i: Basic models*, 429–528.

Wagter, C. D., Tijmons, S., Remes, B. D. W., & Croon, G. C. H. E. D. (2014). Autonomous Flight of a 20-gram Flapping Wing MAV with a 4-gram Onboard Stereo Vision System. *IEEE International Conference on Robotics & Automation (ICRA)*, 4982–4987. doi: 10.1109/ICRA.2014.6907589

Yu, X., & Mitsuo, G. (2010). *Introduction to evolutionary algorithms.* Springer Science & Business Media. doi: 10.1109/ICCIE.2010.5668407