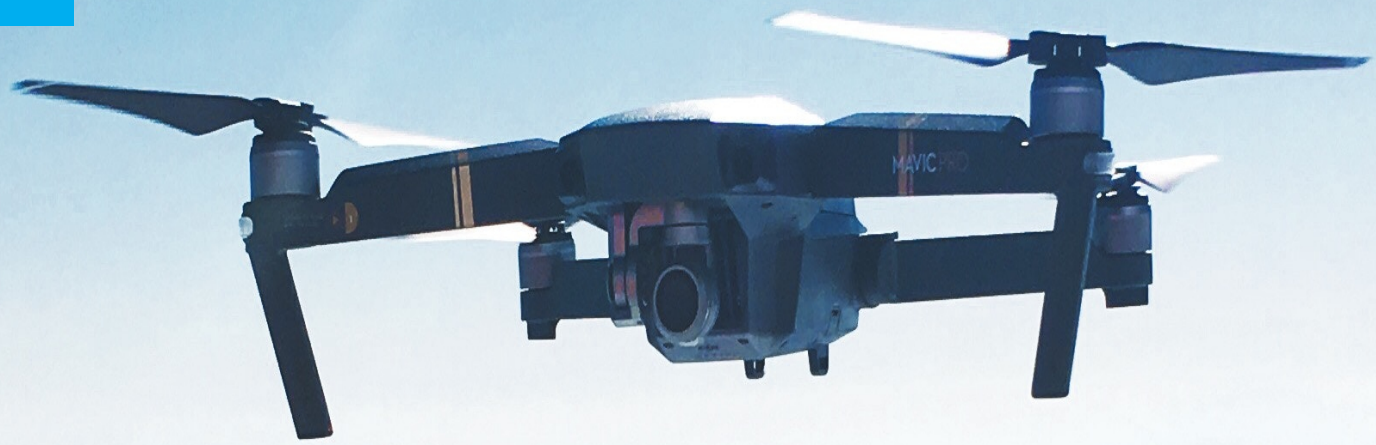
High-Level Command Mapping for Multi-Robot Aerial Cinematography

Robert Durrant

Master of Science Thesis



TU Delft Delft University of Technology

Cognitive Robotics

High-Level Command Mapping for Multi-Robot Aerial Cinematography

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

Robert Durrant

November 23, 2018

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

Delft University of Technology
Department of
Cognitive Robotics (CoR)

The undersigned hereby certify that they have read and recommend to the Faculty of Mechanical, Maritime and Materials Engineering (3mE) for acceptance a thesis entitled

High-Level Command Mapping for Multi-Robot Aerial Cinematography

by

Robert Durrant

in partial fulfillment of the requirements for the degree of

Master of Science Systems and Control

Dated: November 23, 2018

Supervisor(s):

Dr. J. Alonso-Mora

Reader(s):

Prof.dr.ir D. Abbink

Dr. W. Pan

Abstract

Aerial cinematography has seen an increased use of Unmanned Aerial Vehicle (UAV) due to technological advancements and commercialisation in recent years. The operation of such a robot can be complex and requires a dedicated person to control it. Automation of the cinematography allows for the use of multiple robots, which further increases the complexity of performing cinematography. High-level command interpretation is required to allow for an intuitive interface suited for an inexperienced user to control such a system.

Natural Language (NL) is an intuitive interface method which allows a user to specify a extensive range of commands. A Cinematographic Description Clause (CDC) is defined to extract information from a processed NL command. A minimum input approach is considered such that a user has to merely specify the number of robots and the people to record, whereby the specification of a behaviour is optional. An environment is considered in which up to three robots have to frame two people. Taking into account their orientation, relative global location and the user command, a set of behaviours can be determined based on cinematographic practices. Camera views and image parameters are determined through behaviour specific non-linear optimisations and assigned to the robots using a Linear-Bottleneck Algorithm (LBA). A collision-free global path is computed for each robot with an A* search algorithm. Finally, a Model Predictive Control (MPC) determines low-level inputs such that the user command can be achieved.

Three situations are considered to validate the performance of the system given the minimal user input. First, tracking of the dynamic orientations of the people is evaluated for up to three robots, whereby camera positions are determined autonomously. Next, dynamic motions of the two people through an environment highlight the limitations of the system due to collision mitigation, mutual visibility and robot dynamics. An extension to multiple simultaneous commands increases the quantity of robots and people that can be tracked. This allows for an assessment of the flexibility and scalability of

the proposed high-level command interpretation methodology.

Table of Contents

Acknowledgements vii

1 Introduction 1
- 1-1 Related Works . . . 2
- 1-2 Objective . . . 4
- 1-3 Contribution . . . 5
- 1-4 Outline . . . 5

2 Preliminaries 7
- 2-1 Definitions . . . 7
- 2-2 Problem Formulation . . . 10
- 2-3 System Overview . . . 11

3 Natural Language Processing 13
- 3-1 Syntax vs. Semantics . . . 13
  - 3-1-1 Syntax . . . 14
  - 3-1-2 Semantics . . . 14
- 3-2 Inference . . . 15
- 3-3 Spatial Description Clause . . . 16
- 3-4 Cinematographic Description Clause . . . 17

4 Cinematography 19
- 4-1 Taxonomy . . . 19
- 4-2 Configuration: Single Person . . . 23
- 4-3 Configuration: Two People . . . 24

4-3-1 Master Shot . . . 26
4-3-2 Parallel Configuration . . . 26
4-3-3 Face to Face Configuration . . . 27
4-3-4 Right Angle Configuration . . . 28

5 Set-points **31**
5-1 Action Line & Configuration . . . 31
5-2 Target Orientation . . . 32
5-3 Placement Single Camera . . . 33
5-4 Placement Two Cameras . . . 33
5-5 Placement Three Cameras . . . 39
5-6 Feasible Setpoints . . . 39

6 Assignment & Global Planner **43**
6-1 Hungarian Algorithm . . . 43
6-2 Linear-Bottleneck Algorithm . . . 44
6-3 Assignment Comparison . . . 45
6-4 Global Planner . . . 48

7 Model Predictive Control **51**
7-1 Cost Terms . . . 52
7-2 Implementation . . . 55

8 Results **57**
8-1 General Set-up . . . 57
8-2 Simulation: Dynamic Target Orientation . . . 59
8-3 Simulation: Dynamic Motion . . . 71
8-4 Simulation: Multiple Commands . . . 79

9 Conclusion **85**

A Cinematographic Elaboration **89**

B Simulation Supplement **91**
B-1 MPC Weights . . . 91
B-2 Visualisation tool . . . 91
B-3 Simulation: Dynamic Target Orientation . . . 93
B-4 Simulation: Dynamic Motion . . . 93
B-5 Simulation: Multiple Commands . . . 95

C Mambo **97**
C-1 Linux Setup 97
C-2 Establish Connection 98

Bibliography **99**

Glossary **105**
List of Acronyms 105
Nomenclature 106

Acknowledgements

I would like to express my gratitude to my supervisor Dr. J. Alonso-Mora for his guidance and assistance during the course of this thesis project.

I would also like to show gratitude to Hai Zhu and the AMR-group of the Cognitive Robotics department for their aid when requiring advise for solving a problem.

Finally, I am grateful for the moral support of my family and friend over the course of my studies.

Delft, University of Technology
November 23, 2018

Robert Durrant

Chapter 1

Introduction

Consider the situation in which a director on a movie or television show set is giving the cameramen the verbal assignment to frame two actors on camera, i.e. "Record Jeff and Susan from three angles within the scene". From experience, cinematographic practices and knowledge of the scene, the cameramen will determine their position in the environment and the camera angles. However, when substituting the cameramen for aerial vehicles equipped with cameras, the same verbal command does not contain enough information for the vehicles to function. They lack knowledge of the environment, the capacity to interpret the verbal command, which actors need to be framed and cinematographic practices to create aesthetically pleasing footage.

Recent developments of Unmanned Aerial Vehicle (UAV) has made their use within cinematography popular. Unlike the use of cameramen or dollies, UAV are less constrained by gravitational influence. This provides a new repertoire of camera positioning and movement within an environment. Although equivalent results could be achieved with a helicopter, the cost and size of UAVs is significantly lower and multiple vehicles can be used concurrently. Current use might require a certified UAV pilot who can pilot and orient the camera. Some UAVs make use of a pilot and cinematographer, having a multiplicity of two people for each vehicle used.

The vision for this thesis is twofold. The first is to automate the use of multiple UAVs, referred to as agents, for cinematographic use to reduce the required resources. Secondly, simplify the use of such a system such that a single dedicated person or the director could operate it. The current state-of-the-art approaches [1, 2] are taken to further automate the use of the system and incorporate cinematographic taxonomy and practices. The focus is on framing two people or actors, hence referred to as targets, on the image plane of the agents camera. The operator would only have to state the desired number of agents and the targets to frame. Depending on the number of agents within the system,

providing individual command for each agent becomes increasingly cumbersome as the amount increases. Therefore, additional to a minimal input approach, an abstraction from a tele-operation to a behavioural approach is required. This implies that it would be possible to specify a desired behaviour as addition to the minimal input.

Allowing for the specification of a behaviour, cinematographic terms are to be introduced such that professionals within the field could easily use the system. For example, a director could issue desired camera formations and framing on the image plane which the system would be able to achieve autonomously. To get insight into cinematography, [3, 4] are used. They present a taxonomy and describe a large array of situations, camera positioning and image plane positioning of people.

1-1 Related Works

Initially, a range of multi-agent aerial application was considered for the focus of this thesis in the fields of surveillance, construction, and entertainment. Considering surveillance, a decentralized control scheme for a multi-agent system to provide optimal coverage of an environment by minimizing the information per pixel is presented in [5, 6], making use of a location-based multi-hop algorithm. Exploration of an unknown environment by multiple agent is realized in [7], making use of distance sensors for local navigation and reverse hop-counts gradient for long-range navigation.

Within the field of construction, [8] presents the construction of a tower using four quadrotors. Trajectories are planned through the environment using space reservation, waypoint-based navigation and a trajectory planning algorithm. Transportation of a single flexible payload by six quadrotors, minimizing the deformation during flight is presented in [9]. A hybrid Kalman filter and continuous-time, infinite-horizon LQR controller is utilized.

Looking at the field of entertainment, quadrotor choreography synchronized to music is presented in [10], generating offline trajectories accounting for motion constraints and motor limitations. Creating a three-dimensional display with aerial vehicles is presented in [11], making use of a k-means clustering algorithm to generate a representative volumetric point cloud of an object and Optimal Reciprocal Collision Avoidance (ORCA) for collision-free trajectories.

However, the most intriguing application of aerial vehicles within the field of entertainment is aerial cinematography. Current challenges within autonomous UAV cinematography are outlined in [12]. Flight restrictions on outdoor usage, privacy concerns of people features in the recordings and UAV localization are a few such challenges. Focusing on the cinematographic aspect, there are three main challenges. The first relates to the high-level command specification of desired footage by a director. Second is the interpretation and derivation of a low-level cinematographic plan. The third is the autonomous and real-time adaptation to the present environment.

A real-time Model Predictive Control (MPC) controller is presented in [1], where collision avoidance and cinematographic terms are considered. Experiments are performed with a single agent that has to focus on up to three objects. A limitations of this approach is that the global environment and command specification are not taken into account, along with the utilization of a single agent. An extension of this method to account for global trajectories is introduced in [2]. A Model Predictive Contouring Control (MPCC) approach is used, allowing for the specification of cinematographic plan, called virtual rails, for agents to follow. Key-frames for a dynamic scene can be specified through a story-board interface. Experiments are performed with two agents, demonstrating object placement on the image plane and mitigating agent visibility on each others camera frame.

An alternative interface for specifying a global path is presented in [13], making use of a drawing interface to specify trajectories for a single UAV. The user-defined trajectories are optimised with respect to the dynamics of UAV, creating smooth trajectories. Use is made of IQP to solve a non-linear optimisation and an LQR controller is used to track the trajectory for a single agent.

An autonomous cinematographic UAV is presented in [14], where trajectory generation based on camera placements in an indoor environment is considered. A command-line based interface allows for the specification of low-level cinematographic plans to position the UAV to frame up to two people in a static and dynamic environment without obstacles present.

An outdoor environment is considered in [15], where static viewpoints for a single UAV are determined to frame two people on the image plane that are being tracked using RTK GPS. The people can be dynamic when the static viewpoint for the UAV is achieved, not explicitly taking into account dynamic objects. Also, collision avoidance is only implicitly taken into account, meaning a minimum distance is adhered to when the UAV is in motion under the assumption the people remain static.

Current state-of-the-art focusses predominantly on the automation of cinematography for a single or multiple UAVs and the low-level specification of a cinematographic plan. Desired would be an additional abstraction such that high-level command specification can be performed with an intuitive interface methodology. Haptic interaction such as [16], which controls up to three heterogeneous UAVs for a range of applications, would be equivalent to the tele-operation of agents and allows for limited scalability. Therefore, a closer look will be taken at GUI, gesture-based and verbal interfaces.

Visual identification of the operator stance allows for the control of 20 ground-based robots by making use of an external visual sensor [17]. A behavioural approach allows for the exertion of both static and dynamic configurations of the robots. Combining stance identification with the approach presented in [11] to represent three-dimensional objects with a swarm of agents, would allow for a gesture-based interface for high-level cinematographic commands. Object representation is achieve with two robots during experiments and up to 80 UAVs to represent an array of objects.

Making use of multiple vehicles equipped with a visual sensor, gesture commands can be observed in [18, 19]. Through a consensus approach the observed input of vehicles at varying positions and orientations with respect to the operator, is mapped to a control input through support vector machine. The use case is the selection of a subgroup from a swarm of robots.

An external speech recognition system is used in consensus with the facial recognition through which an operator can specify tele-operation commands [20]. Based on the percentage of facial view, three quadcopters achieve a consensus whether the command applied to them.

Navigation through an environment making use of verbal directional statements for a single UAV is performed in [21]. A high-level command mapping into a structure understandable by the robot and grounding elements within the environment allows for the derivation of a low-level trajectory autonomously.

Both gesture-bases and verbal commands are an intuitive interface method for command specification. However, gestures require a constant view of the operator such that inputs can be provided. This could result in one agent having to focus on the operator when an external sensor is not utilised. Verbal commands provide more flexibility, where the operator is less constrained by his location. It also allows for an overview of the environment which is beneficial for a director, contrary to having to be located within to provide gesture in view of the agents.

1-2 Objective

An initial evaluation was performed on methods of interfacing for Human-Robot System (HRS), be it single or multi-agent, and the application for which UAVs are used. The choice was made to focus on the application of multi-agent aerial cinematography with a verbal-command control interface. This resulted in the following research question:

> *How can verbal commands be incorporated within a multi-constraint optimisation problem for real-time, multi-agent cinematography in a dynamic environment?*

The research question constructed is ambiguous and has multiple areas of research grouped together within the formulation. The first research area is Natural Language Processing (NLP), where the focus is on methodologies for language processing. The second research area is the mapping of such a processed language command to multi-agent system commands. The third is the real-time, multi-agent control in a dynamic environment for cinematography. A solution for the third research area is presented in [1, 2]. Considering constraints within the flight envelope for collision avoidance, the dynamic constraints for agents and image plane constraints, agent motion can be computed. A more in-depth description will be presented in Chapter 7.

The implementation of verbal-command control for a system can be segregated into two NLP fields, namely speech recognition and natural language understanding. The former is the processing of the sound wave recorded by a microphone to a string or equivalent understandable structure for a computer. The latter is the labelling of the words or word sections within the string for a computer to understand the natural language input. The understanding of natural language can be quite complex, as desired command/behaviour for the system can be expressed in a large number of word combinations.

Mapping of the processed language command to usable commands for a multi-agent system is a significant problem in itself. The mapping should be able to distil a large variety of possible inputs to usable commands for multiple agents. The basis for the interpretation would be rules or conventions used in the field of cinematography. These would allow the construction of a meta-language between the Natural Language (NL) string and usable input commands for the MPC or agents directly.

Therefore, the focus is on high-level command input interpretation to manoeuvre multiple agent through a dynamic environment to frame two objects on the image plane of each agent based on cinematographic practices. The research question is reformulated as follows:

> *How can control inputs be determined for variable number of agents for tracking two objects on the image planes in a dynamic environment based on a processed NL input command?*

1-3 Contribution

The contribution of this thesis is a methodology for high-level command interpretation for multiple agent towards the application of aerial cinematography. A command structure is defined through which a varying number of agent can be specified to record up to two targets, along with desired framing objectives in accordance with cinematographic practices. The high-level interpretation provides viewing positions and image variables for a low-level controller to achieve. A state-of-the-art low-level controller with minor modification is used to compute agent inputs.

1-4 Outline

The environment, agents and obstacles are defined in Chapter 2, along with a mathematical problem formulation. Next, a brief overview of NLP is presented in Chapter 3. Also, a command structure will be defined for the interpretation of NL cinematographic commands. A cinematographic taxonomy, camera positioning for a single target and target configurations for two targets are presented in Chapter 4. The set-point computation for camera instances given the processed operator input is shown in Chapter 5. These set-points are then assigned to the available agents and a global trajectory is

determined in Chapter 6. The final step is the computation of control commands for individual agents through an MPC scheme in Chapter 7. Simulations are performed for which the results are presented in Chapter 8. Conclusions pertaining to the performance of the system, along with possible future work, are outlined in Chapter 9.

Chapter 2

Preliminaries

This chapter contains a description of the environment and elements contained within in Section 2-1. This is followed by a mathematical problem formulation in Section 2-2. Finally, a outline of the computational steps for the system is presented in Section 2-3.

2-1 Definitions

The general terminology used throughout is as follows. To maintain generality, the term agent is used to refer to the aerial vehicles. The person who has control over the system and provides the control inputs is the operator. A description of the environment, agents and obstacles is provided.

Environment

A dynamic environment is considered, containing agents, static obstacles and dynamic obstacles. A global coordinate system $\mathbb{W} \in \mathbb{R}^3$ is stated in which agents and obstacles are completely defined. A flight envelope is defined for the agents to adhere to, see Equation (2-1). A controlled environment is assumed, meaning the operator has control of the placement of static obstacles and a degree of control on the manoeuvring of the dynamic obstacles.

Assumption 1. *The environment is controlled, meaning the operator specifies the locations of the static obstacles and has influence on the manoeuvring of the dynamic obstacles.*

Assumption 2. *The position and orientation of obstacles and agents within the environment is known at any given time.*

$$\mathbb{W} = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} \in \begin{bmatrix} (x_{min}, x_{max}) \\ (y_{min}, y_{max}) \\ (z_{min}, z_{max}) \end{bmatrix} \tag{2-1}$$

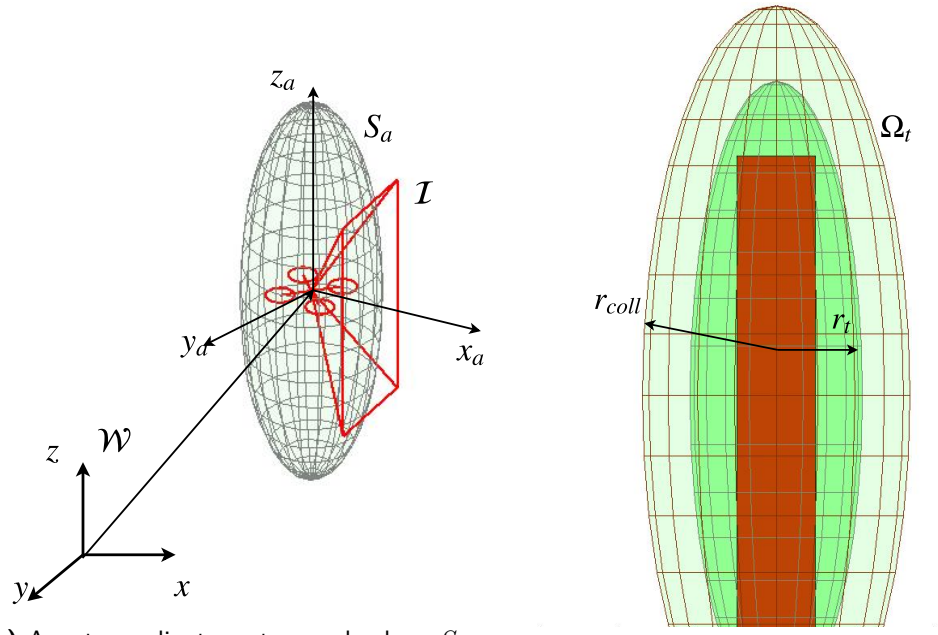
Agents

Agents n are defined as the set of spheres $\mathcal{A} = \{S_a | a = \{1, ..., n\}\}$ with a centre position $\mathbf{p}_a \in \mathbb{W}$, radius r_a and a quaternion orientation $\mathbf{q}_a$ of the agent body frame $\mathbb{A} \in \mathbb{R}^3$ with respect to $\mathbb{W}$. A second radius r_{coll} is defined to create a second sphere S_{coll} around each agent as a virtual potential field to be taken into account for collision avoidance. A representation of the agent can be found in Figure 2-1a. The control inputs $\mathbf{u}_a$ for an agent to manoeuvre in the flight envelope are the pitch angle θ_c, roll angle φ_c, vertical velocity v_{z_c} and the yaw rate $\dot{\psi}_c$. The agents are non-holonomic, having six degrees-of-freedom and only 4 actuators. Use is made of the on-board low-level controllers that provide stable flight, hover when $\mathbf{u}_a$ is zero and allows for vertical take-off and landing. The controllers provide a decoupling of some of the input variables. Commanding the angles (θ_c, φ_c), the agents assumes these rotations. This results in a translation along the global horizontal (x_w, y_w) plane, while maintaining its current height z_w. Vertical movement along the z-axis of the body frame $\mathbb{Q}$ is only affected by v_{z_c}. Equivalent, the yaw ψ_a of the agents is only influenced by the commanded $\dot{\psi}_c$.

The states for an agent are $\mathbf{x} = [\mathbf{p}_a, \mathbf{v}_a, \theta_a, \varphi_a, \psi_a]^T \in \mathbb{R}^9$, where $\mathbf{v}_a \in \mathbb{R}^3$ is the agent's velocity along the global axes. Use can be made of a first-order continuous low-pass Euler approximation [22, 23]. The dynamics are presented in Equation (2-2). The control inputs are not assumed to be the actual values of the states as it could be possible that a jump in control input is present, which is not possible for the agent to achieve. The exception is $\dot{\psi}_c$ which is assumed to be the actual yaw rate, as the yaw adapt based on the dynamics.

$$\begin{aligned} \begin{bmatrix} \dot{v}_x \\ \dot{v}_y \end{bmatrix} &= \mathbf{R}_z(\psi_a) \begin{bmatrix} \tan(\theta_a) \\ -\tan(\varphi_a) \end{bmatrix} - c_d \begin{bmatrix} v_x \\ v_y \end{bmatrix} \\ \dot{v}_z = \frac{1}{\tau_{v_z}} \left(k_{v_z} v_{z_c} - v_z\right) \qquad & \dot{\theta} = \frac{1}{\tau_\theta} \left(k_\theta \theta_c - \theta_a\right) \\ \dot{\varphi} = \frac{1}{\tau_\varphi} \left(k_\varphi \varphi_c - \varphi_a\right) \qquad & \dot{\psi} = \dot{\psi}_c \end{aligned} \tag{2-2}$$

Each agent is equipped with a pan-tilt gimbal camera. Initially, the assumption is made that the camera has a fixed, vertical orientation and the gimbal controls $\mathbf{u}_{gimb}$ are set to zero. This is done to get initial functionality and might be revised if time permits. A camera reference frame $\mathbb{C} \in \mathbb{R}^3$ is defined, along with the image plane $\mathbb{I} = [\mu_x \ \mu_y]^T$ and



(a) Agent coordinate system and sphere S_a representation

(b) Representation of Obstacle Ω_o with safety radius r_t and collision radius r_{coll}

Figure 2-1: Model representations of the an agent and obstacle

position μ along the image axis. The camera plane is assumed to be vertical in $\mathbb{W}$ and in landscape mode. The latter specifies that the horizontal length is larger than the vertical length $\mu_{x_{max}} > \mu_{y_{max}}$. The intrinsic camera parameters $\mathbf{P}$ and camera rotation matrix $\mathbf{R}_c$ can be found in Equation (2-3), where (f_x, f_y) are the focal lengths and (C_x, C_y) is the the centre of the image plane, respectively.

Assumption 3. *The gimbal control inputs* $\mathbf{u}_{gimb}$ *are assumed to be zero.*

Assumption 4. *The camera plane is vertical and in landscape mode* $\mu_{x_{max}} > \mu_{y_{max}}$.

$$\mathbf{P} = \begin{bmatrix} f_x & 0 & C_x \\ 0 & f_y & C_y \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{R}_c = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \end{bmatrix} \tag{2-3}$$

Obstacles

Obstacles within the environment consist of static objects, dynamic objects and targets. A set of obstacles $\mathcal{O} = \{\Omega_o | o = \{1, ..., m\}\}$ is defined for m obstacles in the environment. An ellipsoid Ω_o is defined by the position $\mathbf{p}_o \in \mathbb{W}$ and an orientation ψ_o of the obstacle

body frame $\mathbb{O} \in \mathbb{R}^3$ with respect to $\mathbb{W}$. The length l, width w and height h of the obstacle are supplemented with a safety radius r_t to create the semi-principal axes $(\frac{l}{2} + r_t, \frac{w}{2} + r_t, \frac{h}{2} + r_t)$ for the ellipsoid. Similar to the agents, a second ellipsoid Ω_{coll} with an additive radius r_{coll} to the dimensions is defined as a virtual potential field to be taken into account for collision avoidance. A representation of an obstacle can be found in Figure 2-1b. The assumption is made that the obstacles are connected to the ground plane. This excludes movement along the vertical axis and rotation around the horizontal axes. Therefore, a top-down view is sufficient to define the orientation ψ_o of the obstacles. The targets t that can be selected as a focal point for the agent are a subset of the obstacle set $\mathcal{T} \subset \mathcal{O}$, whereby the position is denoted as $\mathbf{p}_t$ and orientation as ψ_t.

The motion of an obstacle Ω_o within the environment is determined with a constant velocity model $\ddot{\mathbf{p}}_o = \omega_o$, where $\omega_o \sim \mathcal{N}(0, \sigma)$. Given the current position $\mathbf{p}_o$, the future positions are predicted through a standard linear Kalman filter [22, 23].

Assumption 5. *Obstacle dimensions (l, w, h) are assumed constant during operation.*

Assumption 6. *All obstacles are assumed to be connected to the ground plane.*

Control Input

The main focus is on the research area concerned with high-level command interpretation. Therefore, the assumption is made that the input for the system is a processed verbal-command provides by the operator $\mathbf{u}_{op}$.

Assumption 7. *The input that is considered as the system input is a processed, labelled NL verbal-command $\mathbf{u}_{proc}$, given the operator verbal-command $\mathbf{u}_{op}$.*

2-2 Problem Formulation

Given the processed verbal-command input $\mathbf{u}_{proc}$ specified by the operator, agent variables $(\mathbf{p}_a, \mathbf{q}_a)$ and target variables $(\mathbf{p}_t, \psi_t)$, compute the control input $\mathbf{u}_a$ for $n \in \{1, 2, 3\}$ agents to frame $m \in \{1, 2\}$ targets on the respective image planes of the camera, while assuring collision-free motion through the environment. This can be split up into two sub-problems.

The first problem is the high-level command interpretation, i.e. the determination of the desires state $\mathbf{s}_{goal}$ of the system by evaluating the input $\mathbf{u}_{proc}$, which sets the values for n and m. The state represents the set-points for the agents and is defined as $\mathbf{s}_{goal} = \{\mathbf{p}_{a_d} \in \mathbb{R}^{3\times n},\ \sigma_{I_d} \in \mathbb{R}^{m\times n},\ \mathbf{m}_d \in \mathbb{R}^{2\times m\times n}\}$, whereby $\mathbf{p}_{a_d} \in \mathbb{W}$ is the desired position of the agent, σ_{I_d} is the desired size on the image plane and $\mathbf{m}_d \in \mathbb{I}$ is the desired image position. The state of the system is subject to the current $(\mathbf{p}_t, \psi_t)$ of the m targets and $(\mathbf{p}_a, \mathbf{q}_a)$ of the n agents. The formulation is presented in Equation (2-4).

$$\mathbf{s}_{goal} = f(\mathbf{p}_a,\ \mathbf{q}_a,\ \mathbf{p}_t,\ \psi_t,\ \mathbf{u}_{proc}) \tag{2-4}$$

The second problem concerns the computation of low-level control inputs $\mathbf{u}_a$ for each agent $i \in n$ given the goal state $\mathbf{s}_{goal_i}$. A receding horizon optimisation is formulated in Equation (2-5). For a given time horizon N and time step k, the states $\mathbf{x}_k^i \in \mathcal{X}$ and inputs $\mathbf{u}_{a|k}^i \in \mathcal{U}$ that minimizes the cost function g at each time step k. It is subject to the initial state $\mathbf{x}_0^i = \hat{\mathbf{x}}_0$ and the discretisation of the agent dynamics in Equation (2-2). The position of the agent $\mathbf{p}_{a|k}^i$ has to be in $\mathbb{W}$ subject ot avoiding collision with the obstacle set $\mathcal{O}(\mathbf{p}_{o|k})$ at their position at time step k. Also, inter-agent collision avoidance between the set of agents $\mathcal{A}^j(\mathbf{p}_{a|k}^j) = \mathcal{A}(\mathbf{p}_{a|k}) \setminus \{S_a^i\}$ given the position $\mathbf{p}_{a|k}^j$ at time step k needs to be taken into account.

$$\begin{aligned}
\underset{\mathbf{x}_k^i,\ \mathbf{u}_{a|k}^i}{\arg\min}\ & \sum_{k=1}^{N} g(\mathbf{p}_{a|k}^j,\ \mathbf{p}_{t|k},\ \mathbf{s}_{goal}^i) \\
\text{s.t. } & \mathbf{x}_0^i = \hat{\mathbf{x}}_0 \\
& \mathbf{x}_k^i \in \mathcal{X} \\
& \mathbf{u}_{a|k}^i \in \mathcal{U} \\
& \dot{\mathbf{x}}_{k+1}^i = h(\mathbf{x}_k^i, \mathbf{u}_{a|k}^i) \\
& \mathbf{p}_{a|k}^i \in \mathbb{W} \setminus \{\mathcal{O}(\mathbf{p}_{t|k}) \cup \mathcal{A}^j(\mathbf{p}_{a|k}^j)\}
\end{aligned} \tag{2-5}$$

2-3 System Overview

The system consists of a number of computation steps that convert the operator input $\mathbf{u}_{proc}$ to individual agent commands $\mathbf{u}_a$, see Algorithm 1. The first step is to compute the set-points $\mathbf{s}_{goal}$ for an ideal situation. The feasibility of the set-points needs to be evaluated. As there is a disconnect between the set-points and the agents, an assignment needs to be performed. Once assigned, a global trajectory is determined for each agent. Finally, the input $\mathbf{u}_a$ for each agent can be determined and applied.

Algorithm 1 System Description

$\mathbf{u}_{proc} \leftarrow$ Operator input	▷ Chapter 3
$\mathbf{s}_{goal} \leftarrow$ Compute set-points	▷ Chapter 5
Check feasibility $\mathbf{s}_{goal}$	▷ Section 5-6
Assign $\mathbf{s}_{goal}$ to n agents	▷ Chapter 6
for all n **do**	
Compute global trajectory	▷ Section 6-4
$\mathbf{u}_a \leftarrow$ Compute agent input	▷ Chapter 7
Apply $\mathbf{u}_a$	
end for	

Chapter 3

Natural Language Processing

The field of Natural Language Processing (NLP), although not the main focus of the thesis, is considered to gain an understanding of the type of control input that could be given. A mapping needs to be formulated to process the vocal, frequency input into a usable, labeled structure. An increasing number of services and machines incorporate NLP, such as Siri from Apple[1], Google Assistant[2] or Amazon Alexa[3]. A large range of applications can be accessed and utilized by voice prompts. However, the methods used to process recorded prompts is generally a black box. Therefore, a range of processing methods are presented in Section 3-1. When implementing these methods for generalized use, inference has to be taken into account, see Section 3-2. Implementation of a NLP approach to direct an aerial vehicle through an environment is presented in Section 3-3. Finally, this approach is customized for use in a cinematographic paradigm in Section 3-4.

3-1 Syntax vs. Semantics

A thorough survey of NLP is presented in [24], with the following definition: "NLP is a theory-motivated range of computational techniques for the automatic analysis and representation of human language". The focus is on the evolution of NLP, presenting three paradigms with a range of approaches. The three paradigms are Syntax, Semantics and Pragmatics. Current research pertains to the Semantic paradigm, having Pragmatics as future achievement.

[1] Apple Siri: https://www.apple.com/siri (accessed 17 September 2018)
[2] Google Assistant: https://assistant.google.com (accessed 17 September 2018)
[3] Amazon Alexa: https://developer.amazon.com/alexa (accessed 17 September 2018)

3-1-1 Syntax

The foundation of a linguistic structure is the word. In the Syntax paradigm, each word is individually processed and assigned meaning. Today, syntax-centered NLP is still the most popular way to manage tasks such as information retrieval and extraction, auto-categorization and topic modeling. There are three popular ways of approaching syntactical NLP, namely: keyword spotting, lexical affinity and statistical NLP.

Keyword Spotting

As the name suggests, keyword spotting evaluates the provided textual input to classify into categories based on a reference corpus. The linguistic corpus consists of unambiguous words, which are surface features. Reliance on the presence of surface features is the major weakness of this method. Despite this weakness and naive approach, the relative simplistic implementation makes it popular for usage.

Lexical Affinity

Given a linguistic corpus, a probabilistic affinity is assigned to arbitrary words in a text. The text is then organized based on these assigned affinities. This method outperforms the spotting of keywords. However, due to operation on word-level, it can be tricked by words with differing contextual meanings. The source of the corpus introduces a bias and limits re-usability.

Statistical NLP

Statistical NLP makes use of machine-learning algorithms trained on a large annotated, linguistic corpus. It is able to learn the valence of desired and arbitrary keywords, punctuation and word co-occurrence frequency. Acceptable accuracy can be achieved when the text input is sufficiently large, but is semantically weak.

3-1-2 Semantics

Semantic-based NLP focuses on the intrinsic meaning associated with natural language text, relying on implicit denotative features associated with natural language text. Concept-based approaches are able to detect semantics that are expressed in a subtle manner, through the analysis of concepts that do not explicitly convey relevant information, but which are implicitly linked to other concepts that do so.

Endogenous NLP

Machine-learning techniques are used to build structures that approximate concepts found in a large set of documents. Endogenous knowledge is built up and used to analyze the corpus, exempt fro prior semantic understanding. Two learning approaches can be taken, namely lexical semantics and compositional semantics. Lexical semantics makes use of the meaning of individual words, while compositional semantics considers the meaning of sentences.

Taxonomic NLP

Taxonomic NLP makes use of subsumptive or hierarchical semantics expressed in NL. Subsumptive knowledge representations build upon the IsA relationship extracted through syntactic patterns for automatic hypernym discovery. Relying on subsumptive knowledge, which is strictly defined, combined handling of differing nuances concepts is not possible. Also, the linguistic corpus sets a typicality in the taxonomy.

Noetic NLP

Making use of machine-learning techniques such as neural networks, idiosyncratic knowledge about objects, actions events and people is determined. The objective is to perform reasoning in an adaptive and dynamic way, generating context-dependent results and discovering new semantic patterns. Meaningful semantic elements are determined through construction-based semantic parsing and performing part-of-speech tagging. Constructions are composed of fixed lexical elements. They provide the parser a sense of what lexical elements are used together. Constructions are typically nested within one another, allowing for the capability of finding only those construction overlaps that are semantically sensible. The main advantage is that small sections of text are required to extract meaning.

3-2 Inference

Describing an environment or task can be done in a multitude of word combinations. This makes the extraction of information from an input a complex problem. Also, the amount of information can differ depending on the operator. The concept of level of abstraction is addressed in [25]. Stated is that a NL command can be divided into abstract, mid-level and fine-grained commands. An example of this would be 'Record John', 'Frame John from the front' and 'Set the camera in front of John and frame him from the front in the centre of the image', respectively. Depending on the contents of the command, additional information is required to execute functionality. Therefore, inference is required. It is defined as: "The act of passing from statistical sample data

to generalizations"[4]. Based on frequent occurring views for cinematography, a type of default behaviour can be specified to compensate for the lack of information provided in the input. A system that can leverage inference will provide robustness for variation in operator input commands, while task grounding is more accurate.

3-3 Spatial Description Clause

A approach to traverse a mapped environment with an aerial vehicle by providing NL directions is presented in [21]. The objective is to create a robust system that accepts NL commands from novice operators, while accommodating for as diverse a range of inputs as possible. The input has to be mapped to motion primitives for the agent to perform. Making use of noetic NLP, a construction named Spatial Description Clause (SDC) is defined in [26]. As a noetic NLP is applied, it is possible to evaluated input text as small as a single sentence. The SDC is a semantic structure that robustly captures the meaning of spatial directions and consists of four basic elements, namely the figure F, a verb V, a spatial relation SR and a landmark L as seen in Figure 3-1. Each element can be omitted and determined implicitly based on the current position and environment.

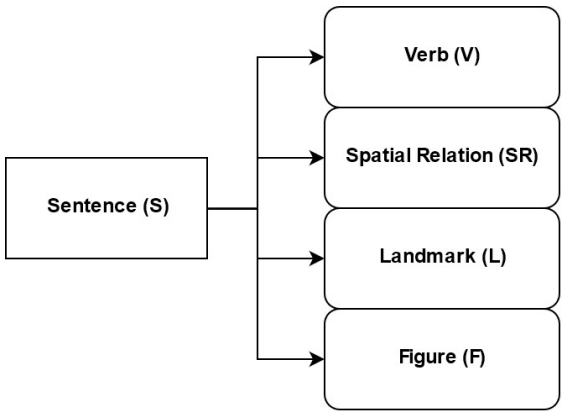


Figure 3-1: Spatial Description Clause as defined by [26]

A Conditional Random Field (CRF) is used to label semantic elements within the command, which is trained on an annotated corpus. A greedy algorithm allows for the formation of SDC. Based on the sequential nature of NL directions, multiple commands can be provided in one input. The SDC is grounded within the environment by assigning a probability to each element given respective to predefined sets. An optimised path is then determined by computing the maximum joint distribution of the path and the SDC by making use of the Viterbi algorithm [27].

[4]Merriam-Webster definition: https://www.merriam-webster.com/dictionary/inference (accessed 17 September 2018)

3-4 Cinematographic Description Clause

The SDC is designed for the application of single agent navigation through an environment. This is not sufficient for the envisioned task as cinematographic terms need to be identifiable and usable for multiple agents. Therefore, the SDC is redefined as the Cinematographic Description Clause (CDC) to utilize cinematographic taxonomy. Figure 3-2 depicts the structure, allowing for the specification of the number of agents, the focal objects, framing requirements and camera positioning. The cinematographic taxonomy is elaborated further in Chapter 4.

The SDC is designed for the application of single agent navigation through an environment. Use of multiple agents within a cinematographic application will not be expressible with this construction. Therefore, the SDC is refined as the CDC to utilize cinematographic taxonomy. Figure 3-2 depicts the construction, allowing the specification of the number of agents, the targets and cinematographic elements.

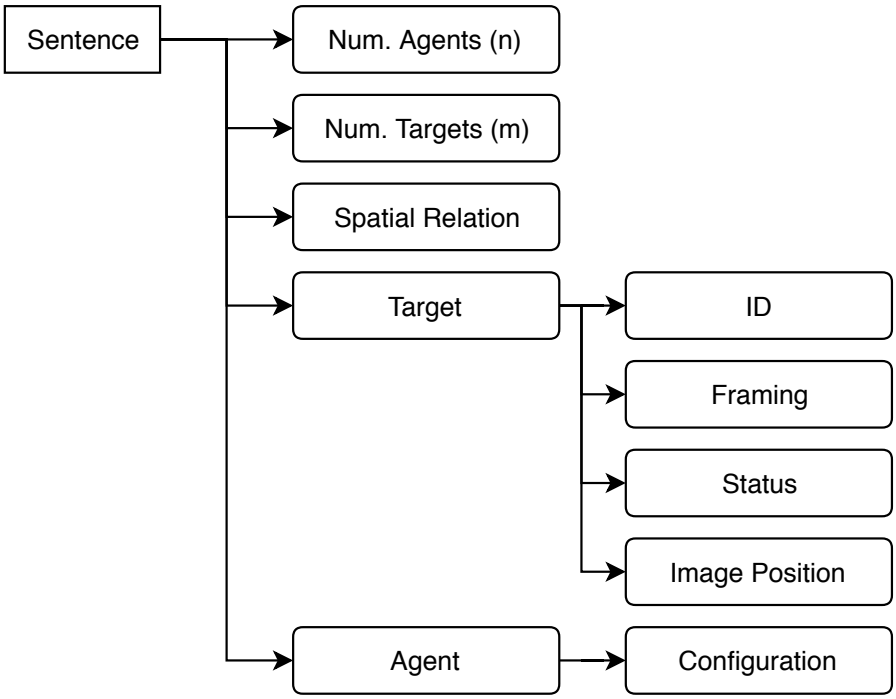


Figure 3-2: Cinematographic Description Clause

Spatial relation is the configuration the agents are desired to achieve around the defined targets. Each target can be defined by a nested construction, where a label referring to the target in the environment, the camera framing, status and image position can be defined. For each agent, the relative configuration with respect to the targets can be specified. Each element can be omitted when specifying a NL command, whereby parameters revert to default settings or inferred.

Visualization

As Assumption 7 is made, a GUI panel is designed to simulate the CDC that can be specified, see Figure 3-3. The minimum input $\mathbf{u}_{cdc}$ the operator has to specify is the number of agents to be used and the number of targets to focus on. Based on this information, all other parameters are inferred. Note that the inferred parameter can be specified or altered to achieve the desired behaviour.

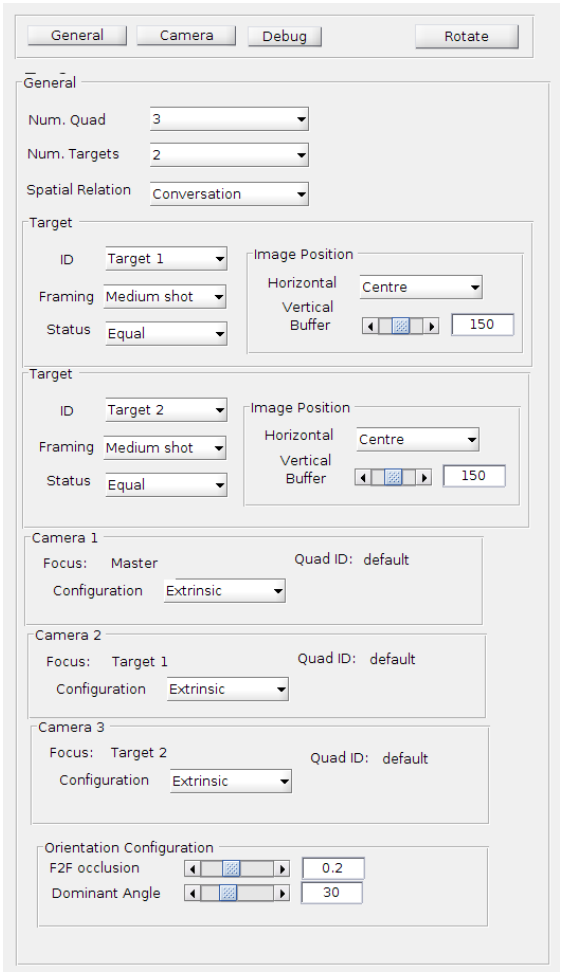


Figure 3-3: GUI panel to specify the CDC command $\mathbf{u}_{cdc}$

Chapter 4

Cinematography

To relate the labels in the CDC to quantifiable variables that can be used for computation, a cinematographic taxonomy is presented in Section 4-1. Making use of a variable number of agents, alternative configurations can be implemented. First, configurations are presented for a single target on the image plane in Section 4-2. Then, the two target configurations are presented in Section 4-3. The term camera is used, not agent. This is done as an ideal camera instance is presumed to be positioned in the environment.

4-1 Taxonomy

The main sources of information for setting up a taxonomy for the cinematographic rules and conventions that are applied are [3, 4]. Within cinematography, the only product a consumer sees is the final edit of the recorded camera images by the agents. Therefore, the first terms introduced are related to the image plane $\mathbb{I}$ of an agent's camera. These are followed by terms related to positioning cameras in the environment $\mathbb{W}$.

Size

The first term is the size of a target σ that needs to be considered for framing. The size relates to the target ellipsoid Ω_t dimensions or fraction thereof to be projected onto the image plane $\mathbb{I}$ of the camera. As the focus is on framing people as targets, the following assumptions are made.

Assumption 8. *The height h of a target is the limiting factor when projecting a target onto the image plane $\mathbb{I}$.*

Assumption 9. *The most prominent focal point of a person is the head or face.*

When projecting on the image plane $\mathbb{I}$, the height of a target h translates to the vertical image dimension μ_y, resulting the prominent limiting factor. Also, when considering a person, the height is the prominent factor $h > (w, l)$. The head of a target is assumed to be the dominant feature. Therefore, multiple relative sizes σ can be defined with respect to the top of a target's head $\mathbf{p}_t$, along with an additional variable for the height of the head h_h. Table 4-1 presents the relative sizes with respect to the target that can be considered for projection, where a visual representation can be found in Figure A-1. A point $\mathbf{p}_{ip}$ for projection onto the image plane $\mathbb{I}$ is defined at half the size σ below the height h of a target. The operator can set the label for the size in the framing of the target element of $\mathbf{u}_{cdc}$. When this information is not provided, a Medium Shot is inferred.

Table 4-1: Relative sizes σ for framing on image plane $\mathbb{I}$

Label	Description	Size σ [m]	Centre $\mathbf{p}_{ip}$ [m]
Full shot	A person is framed from head to toe	h	$(x_t, y_t, \frac{h}{2})$
Medium shot	A person is framed from head to hip	$\frac{h}{2}$	$(x_t, y_t, \frac{3h}{4})$
Close shot	A person is framed from head to shoulders	$\frac{h}{4}$	$(x_t, y_t, \frac{7h}{8})$
Close up	A person is framed with head only	h_h	$(x_t, y_t, h - \frac{h_h}{2})$

The size on the image plane σ_I can be computed with Equation (4-1), making use of the focal length f and the distance d between the camera position $\mathbf{p}_c$ and $\mathbf{p}_{ip}$, see Equation (4-1).

$$\sigma_I = \frac{\sigma \|f\|}{\|d\|} \qquad d = \mathbf{p}_{ip} - \mathbf{p}_c \tag{4-1}$$

To be able to specify the desired image size σ_{I_d}, a buffer variable $\mu_{y_{buff}}$ is defined such that the operator can set the total number of vertical pixels the size has to be smaller than vertical image size, see Equation (4-2). When this value is not specified by the operator in $\mathbf{u}_{cdc}$, a default value is inferred.

$$\sigma_{I_d} = 2C_y - \mu_{y_{buff}} \tag{4-2}$$

Image Position

The position $\mathbf{p}_{ip}$ is projected onto the image plane and defined with respect to the centre of the image. As the origin of the image plane is defined at the top left-hand corner, the projected position $\mathbf{m}$ is determined with Equation (4-3).

$$\mathbf{m} = \begin{bmatrix} \mu_x - C_x \\ \mu_y - C_y \end{bmatrix} \tag{4-3}$$

To get initial functionality, the vertical image position is fixed $\mu_{y_d} = C_y$. As the vertical framing in Assumption 4 is taken into account with the projected size, this assumption can be made. For the horizontal position μ_x, the following labels are defined from left to right, see Table 4-2. Centring the target is a logical initial positioning. The image can be split through the centre, creating two new centre positions in each half. The rule of thirds is another way to frame a target, setting a position at one third and two third of the image plane. The image position $\mathbf{m}_d$ can be set with the image positioning element for each target in $\mathbf{u}_{cdc}$. When this information is not provided, a Centre position is inferred for a single target and a Left-Right positioning for two targets, respectively.

Table 4-2: Categorisation of image positions

Label	Description	Image Position $\mathbf{m}$ [px]
Left	Target on left half of image plane	$(\frac{C_x}{2}, C_y)$
One Third	Target at a third of the image plane	$(\frac{2C_x}{3}, C_y)$
Centre	Target in the centre of the image plane	(C_x, C_y)
Two Third	Target at two-thirds of the image plane	$(\frac{4C_x}{3}, C_y)$
Right	Target on right half of the image plane	$(\frac{3C_x}{2}, C_y)$

Continuity

Continuity is a term related to the image position of the targets on the image plane of multiple cameras. To create visually consistent and aesthetically pleasing footage, the target position $\mathbf{m}$ should be equivalent. This ensures that when switching cameras the targets inhabit the same image space. It ensures that during editing of the camera images, the targets do not jump around the image space and are where the consumer would expect them to be when a camera switch occurs.

Action line

The action line $\mathbf{a}$ is a vector parallel to the ground plane in the global coordinate system W that defined the main direction of action within a scene. For a single person situation, the vector is oriented along the yaw orientation of the target ψ_t. For a two person situation, the vector is defined as the relative distance, see Equation (4-4). The origin of the action vector is the target position $\mathbf{p}_t^i$. The action line divides the workspace, reducing the solution search space of possible positions agents can have.

$$\mathbf{a}_1 = \begin{bmatrix} \cos(\psi_t) \\ \sin(\psi_t) \\ 0 \end{bmatrix} \qquad \mathbf{a}_2 = \mathbf{p}_t^j - \mathbf{p}_t^i \tag{4-4}$$

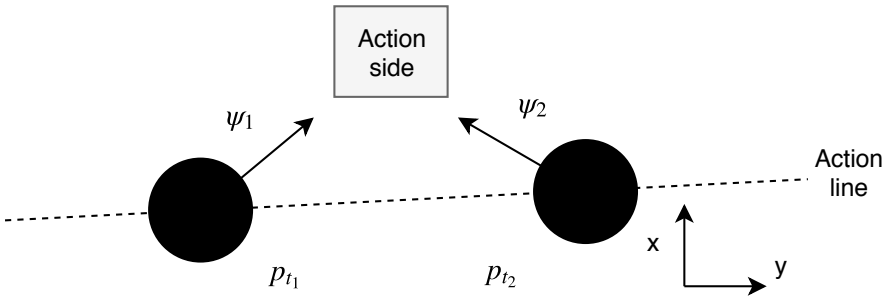


Figure 4-1: Action line for two targets

The camera positions $\mathbf{p}_c$ are all set on one side of the action line $\mathbf{a}$. This ensures that the targets occupy equivalent image spaces. The angle range on the action side is defined as ϵ. Positioning of cameras can be achieved on the alternative side of the action line. However, an active transition of a camera over the action line needs to occur. This flips the image space the targets occupy. From a continuity standpoint, having cameras placed on both sides of the action line creates footage whereby the position of the targets shift on the image plane. This makes the footage hard to follow for consumers, as the focus point is altered between shots, making the consumer have to shift attention.

Master Shot

A master shot is defined as a framing that provides an overview of the scenario occurring in the workspace [3]. This implies that both targets $\mathbf{p}_{ip}$ are visible on the image plane $\mathbb{I}$. The definition in [4] is slightly different, stating that the shot is wide enough to capture all considered targets in the scenario and that the shot captures the entire length of an action. Given a variable number of agents n, at least one must function as a master shot.

Explicit vs. Implicit

This term relates to the placing of the cameras $\mathbf{p}_c$ with respect to a perpendicular line to the action line $\mathbf{a}_\perp$ that originates in $\mathbf{p}_t$ of the considered target and is indicative of multiple target framing. An explicit camera placement is where it is placed outside the area created by the perpendicular lines, see Figure 4-2. This allows for both targets to be framed on the image plane. An implicit camera placement is where the camera is placed within the area between the two targets. This restricts the camera to only focus

on a single target. The camera placement can be set in the configuration element for each agent in $\mathbf{u}_{cdc}$, where Explicit is inferred when omitted.

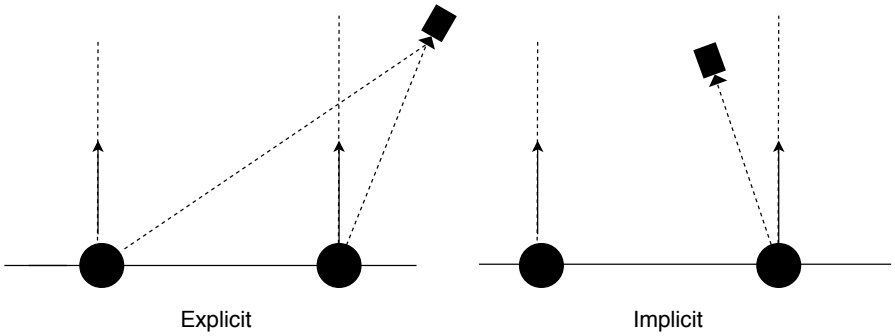


Figure 4-2: Explicit and Implicit camera placement

Dominant vs. Submissive

As there are two targets, there should be a possibility to specify that one target has a higher importance in the scenario. This implies that a camera can focus mainly on one target, ignoring the second target to be framed. This term is relevant one when more than one agent is utilised. There are labels defined, namely: Dominant, Submissive and Equal. When considering two cameras, if one target is set as Dominant, the other target is per definition Submissive. The Submissive camera will frame both targets to create a master shot. When three cameras are used, both targets could be set to Dominant as the third agent will create a master shot. The default setting that the system infers if the status element for the target in $\mathbf{u}_{cdc}$ is omitted, is that the targets are of Equal importance. This will create a Submissive placement for both targets.

4-2 Configuration: Single Person

When considering the camera placement $\mathbf{p}_{c_d}$ with respect to a single person, there is a limited amount of positions that make sense. The distance from the person is mainly determined by the target size σ_{I_d} on the image plane. Depending on the number of agents used, these following positioning are considered. The position computations are relative to the targets. This allows for the target to be either static or dynamic. Ideal desired positions are computed for the following configurations. These configuration can be set in the spatial relation element of $\mathbf{u}_{cdc}$.

Front view

Having up to three agents available, the front view can be specified in Figure 4-3. The agents are placed in a line facing the front of the target subject to the orientation ψ_t.

When one camera is specified in $\mathbf{u}_{cdc}$, the camera is placed along the orientation ψ_t at a distance determined by Equation (4-1). For multiple cameras, an angle variable α_{fr} is defined for the operator to tune the positioning $\mathbf{p}_{c_d}$ with respect to the target orientation ψ_t.

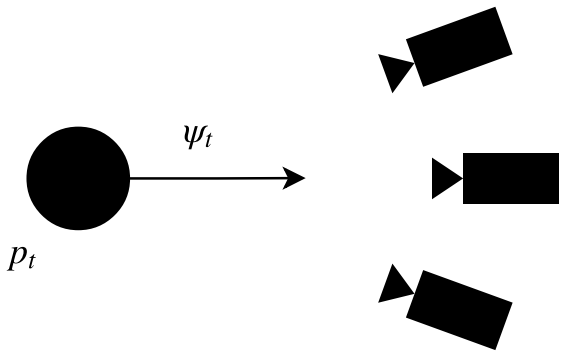


Figure 4-3: Front view configuration for a single target

Circle Configuration

A second configuration is placing the cameras n in a circle around the target $\mathbf{p}_t$, see Figure 4-4. A fixed angle α_{circ} interval is defined based on the number of agents used. The relative distance from the target position $\mathbf{p}_{ip}$ is set by the desired size σ_d and image size σ_{I_d} set in $\mathbf{u}_{cdc}$.

$$\alpha_{circ} = \frac{2\pi}{n} \tag{4-5}$$

Cinematic Configuration

A front, side and back view are defined with respect to a target to create the cinematographic view in Figure 4-5. The cameras are placed consecutively in this order based on the number of agents specified in $\mathbf{u}_{cdc}$. The camera position $\mathbf{p}_{c_d}$ is subject to the orientation of the target ψ_t, whereby the configuration mimics the change in orientation.

4-3 Configuration: Two People

The tracking of two people on an image plane adds terms, variables and parameters that need to be considered. This increases the possible solution space and adds constraints to the problem. Setting the number of targets element in $\mathbf{u}_{cdc}$ to two will make tracking of two targets possible. Also, as the configurations are highly dependable on the orientation of both targets, a single spatial relation label is set for the $\mathbf{u}_{cdc}$, namely Conversation.

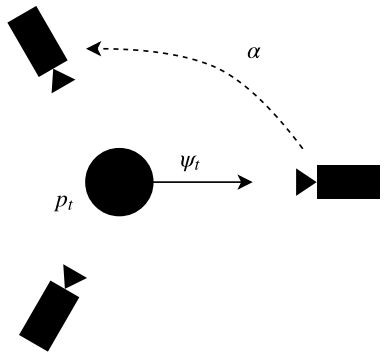


Figure 4-4: Circle configuration for a single target

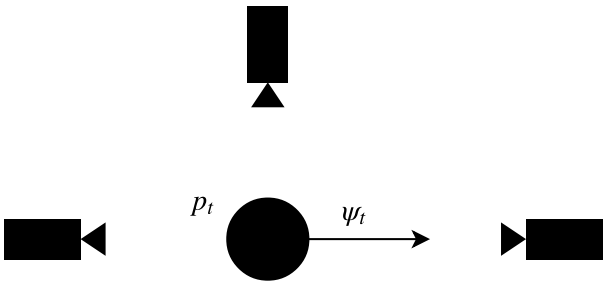


Figure 4-5: Cinematographic view for a single target

When considering two targets, three base configurations can be defined to describe the relative orientation of the targets. A visual representation is presented in Figure 4-6, along with a description in Table 4-3. There are three base configurations, namely Parallel, Face to Face and Right Angle. The angle between the heading vectors $\boldsymbol{\psi}_t = \mathbf{R}_z(\psi_t)[1\ 0]^T \in \mathbb{R}^2$ allows for categorization into the base configurations, making use of Equation (4-6). When both orientations are equivalent, the base configuration is Parallel. The Right Angle base configuration occurs when the orientations are approximately orthogonal. Finally, the Face to Face base orientation is when targets are oriented in opposite directions.

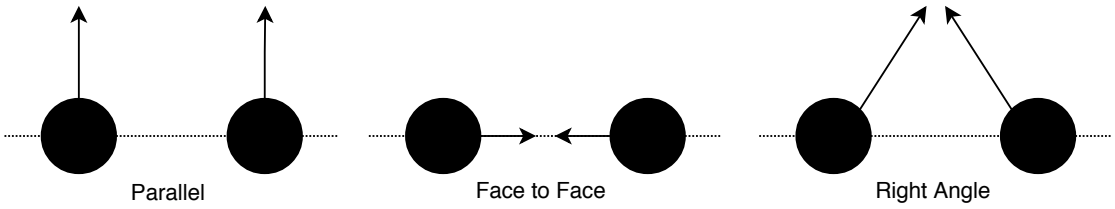


Figure 4-6: Two target base configurations

Table 4-3: Base Configurations for two target scene

Configuration	Description	ψ Range [rad]
Parallel	Orientations are facing the same direction	$[0, \frac{\pi}{3}\rangle$
Right Angle	Orientations are approximately orthogonal	$[\frac{\pi}{3}, \frac{2\pi}{3}]$
Face to face	Orientations are facing the opposite direction	$\langle\frac{\pi}{3}, \pi]$

$$\gamma = \cos^{-1}\left(\frac{\boldsymbol{\psi}_t^j \cdot \boldsymbol{\psi}_t^i}{|\boldsymbol{\psi}_t^j||\boldsymbol{\psi} i_t|}\right) \tag{4-6}$$

4-3-1 Master Shot

This configuration is considered when there is a single camera to place within the environment. A top-down view is adhered for positioning the camera $\mathbf{p}_c$. The objective for this configuration is to maintain the two targets on the image plane $\mathbb{I}$. Therefore, the orientation of the targets ψ_t are neglected and the positioning is computed based on the positions of the targets $\mathbf{p}_t$. As the orientation is not considered, the master shot is applicable for all base configuration types. The geometry is presented in Figure 4-7.

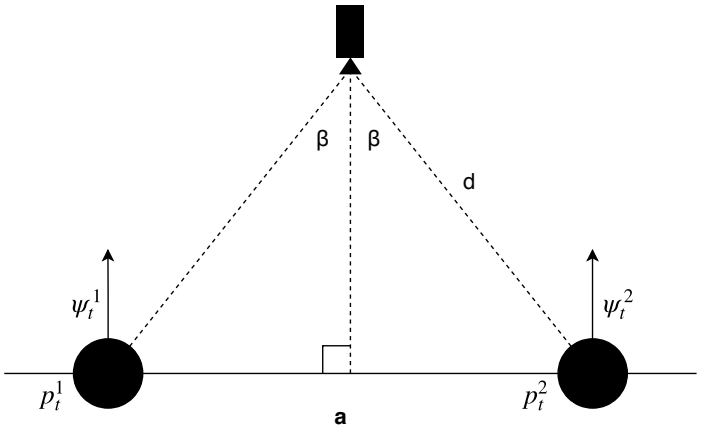


Figure 4-7: Master shot configuration

Within this configuration, the camera position $\mathbf{p}_c$ can be determined by making use of a isosceles triangle. The image positions $\mathbf{m}_d$ determine where the targets are desired to be on the image plane $\mathbb{I}$. An angle β can be defined to compute the distance d at which the camera has to be positioned. The main influence on this is the relative distance $\|\mathbf{p}_t^j - \mathbf{p}_t^i\|$ between the targets.

$$\beta = \tan^{-1}\left(\frac{\mu_x}{\|f\|}\right) \qquad d = \frac{\|\mathbf{p}_t^j - \mathbf{p}_t^i\|}{2\sin(\beta)} \tag{4-7}$$

4-3-2 Parallel Configuration

When two or more cameras are considered for placement, three distinct configurations of the targets can be determined for the Parallel base configuration with respect to the action line, see Figure 4-8. Examples of mainly the second configuration are given in [3, 4]. However, to quantitatively define this base configuration, the addition of the two other configurations is required. When positioning the cameras, the positioning $\mathbf{p}_{c_d}$ is performed in a counter-clockwise manner on the considered action side. Therefore, although mirrored, the first and third configuration are defined. This allows for the

definition of distinct target status combinations characteristic to certain configurations that override the operator input $\mathbf{u}_{cdc}$.

Assumption 10. *The camera positioning is performed in a counter-clockwise direction on the determined action side.*

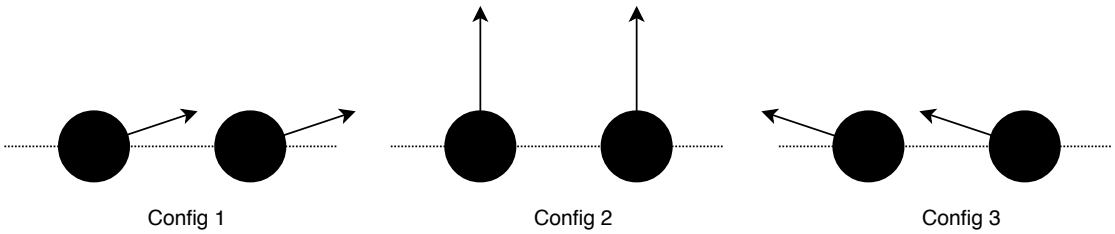


Figure 4-8: Parallel configurations

As the target orientations are parallel, only the relative angle γ between the heading vector $\boldsymbol{\psi}_t$ and the action line $\mathbf{a}$ needs to be determined using Equation (4-8). The characterisation of the configurations is presented in Table 4-4.

$$\gamma = \text{atan}_2 \left(\frac{\|\boldsymbol{\psi}_t \times \mathbf{a}\|}{\boldsymbol{\psi}_t \cdot \mathbf{a}} \right) \tag{4-8}$$

For the first configuration, the target status for the first considered target for camera placement is set to Dominant, with the second being Submissive. This is done such that the second camera generates a master shot of the targets. This overrides the operator input $\mathbf{u}_{cdc}$ if provided. The second configuration has no overriding influence, making use of $\mathbf{u}_{cdc}$. Finally, the third configuration has the first target status set to Submissive and the second to Dominant.

Table 4-4: Parallel configuration classification

Configuration	$\lvert\gamma\rvert$[rad]	Target Status
1	$[0, \frac{\pi}{4}\rangle$	[Dominant, Submissive]
2	$[\frac{\pi}{4}, \frac{3\pi}{4}]$	$\mathbf{u}_{cdc}$
3	$\langle \frac{3\pi}{4}, \pi]$	[Submissive, Dominant]

4-3-3 Face to Face Configuration

The Face to Face base configuration has the premise that there is a π rad difference between the target orientations ψ_t. This results in four configurations for this base configurations, see Figure 4-9. The first configuration is where the targets are actually

facing each other and the fourth configuration has the targets with their backs to one another, expressed in [3, 4]. The second and third are configurations where each target has their orientation ψ_t on alternative sides of the action line **a**.

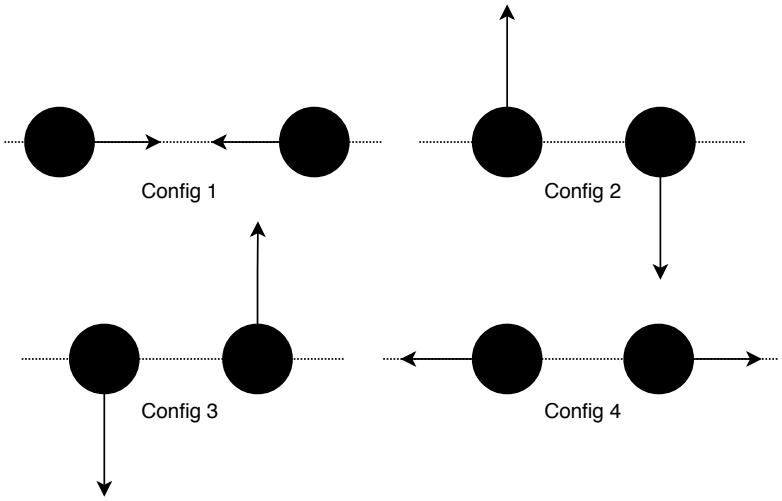


Figure 4-9: Face to Face configurations

To distinguish between the configurations, Equation (4-8) is used. As seen in Table 4-5, for the second and third configuration, the input $\mathbf{u}_{cdc}$ is overwritten. The second configuration sets the first target status to Submissive, as the orientation is on the incorrect side of the action line. The second target is set to Dominant as a master shot already is present. The same logic is applied to the third configurations, setting the first target to Dominant and the second to Submissive.

Table 4-5: Face to Face configuration classification

Configuration	$\lvert\gamma\rvert$[rad]	Target Status
1	$[0, \frac{\pi}{4}\rangle$, $\langle\frac{7\pi}{4}, 2\pi]$	$\mathbf{u}_{cdc}$
2	$[\frac{\pi}{4}, \frac{3\pi}{4}]$	[Submissive, Dominant]
3	$\langle\frac{5\pi}{4}, \frac{7\pi}{4}]$	[Dominant, Submissive]
4	$\langle\frac{3\pi}{4}, \frac{5\pi}{4}]$	$\mathbf{u}_{cdc}$

4-3-4 Right Angle Configuration

The Right Angle base configuration has the most configurations possible with respect to the action line **a**. There are eight configurations defined in Figure 4-10. The main configuration treated in [3, 4] are the second and the fifth. However, the additional

configuration are required to define all the possible Right Angle configurations with respect to the action line **a**.

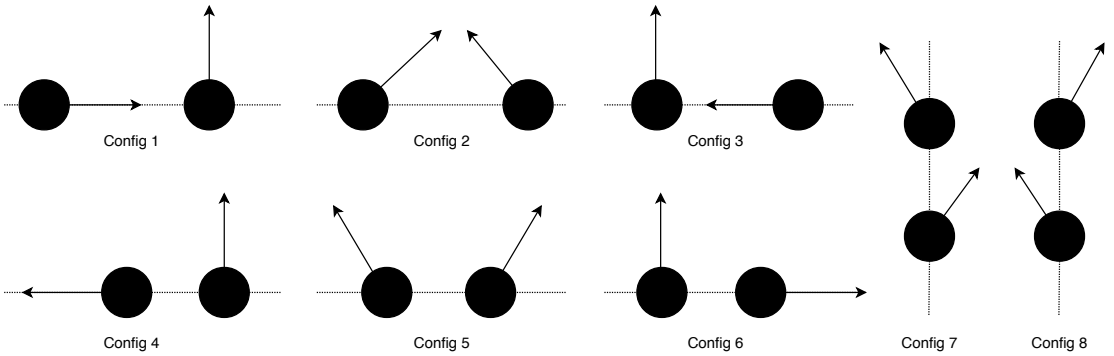


Figure 4-10: Right Angle configurations

The heading vector $\boldsymbol{\psi}_t$ of both targets is summed, creating a vector $\boldsymbol{\psi}_{sum}$ which allows for categorization of the configurations by computing the relative angle γ with the action line **a**, see Equations (4-8) and (4-9). An additional step has to be performed for explicit classification, because for example an equivalent vector is computed for configuration three and four. To distinguish, the relative angle γ_{ra} is computed between $\boldsymbol{\psi}_{sum}$ and the target orientation of the first target $\boldsymbol{\psi}_t^i$. The result is presented in Table 4-6.

$$\boldsymbol{\psi}_{sum} = \boldsymbol{\psi}_t^i + \boldsymbol{\psi}_t^j \tag{4-9}$$

Table 4-6: Right Angle Orientation classification

Configuration	$\boldsymbol{\psi}_{sum}$[rad]	γ_{ra}[rad]	Target Status
1	$[\frac{\pi}{8}, \frac{3\pi}{8}]$	Positive	[Dominant, Submissive]
2	$[\frac{3\pi}{8}, \frac{5\pi}{8}]$	Positive	$\mathbf{u}_{cdc}$
3	$\langle\frac{5\pi}{8}, \frac{7\pi}{8}]$	Positive	[Submissive, Dominant]
4	$\langle\frac{5\pi}{8}, \frac{7\pi}{8}]$	Negative	[Dominant, Submissive]
5	$[\frac{3\pi}{8}, \frac{5\pi}{8}]$	Negative	$\mathbf{u}_{cdc}$
6	$[\frac{\pi}{8}, \frac{3\pi}{8}]$	Negative	[Submissive, Dominant]
7	$[0, \frac{\pi}{8}\rangle$, $\langle\frac{7\pi}{8}, \pi]$	-	$\mathbf{u}_{cdc}$

Chapter 5

Set-points

Given the operator input $\mathbf{u}_{cdc}$, environmental information and cinematographic taxonomy, a Finite State Machine (FSM) can be defined for the two person situation to convert $\mathbf{u}_{cdc}$ into a goal state $\mathbf{s}_{goal}$, as stated in Equation (2-4). A flowchart of the steps required to determine the state is presented in Figure 5-1. An assumption is made that the two targets are interacting with each other. The use of interaction is taken as a more abstract term in this assumption, meaning that actual interaction between two targets does not have to be present, i.e. touching or talking. A walk-through of the computation methodology is presented in this chapter, starting with the action line, base configuration, specific configuration and target orientation computations in Sections 5-1 and 5-2. The relative placement of a single camera with respect to two targets is presented in Section 5-3. Two and three camera placement is considered in Sections 5-4 and 5-5, respectively. Finally, the feasibility of the computed set-points is evaluated in Section 5-6.

5-1 Action Line & Configuration

The first step is to determine the base configuration, action line, preferred side of the action line and the specific configuration. Making use of Equations (4-4) and (4-6), the base configuration and the action line $\mathbf{a}$ are computed. A side of the action line needs to be selected for the cameras to be placed on. Side selection is performed to achieve the most prominent information of the target on the image plane. The action side is determined by making use of Equation (4-9) and computing the relative angle γ_a with respect to the action line. If the angle is negative $\gamma_a < 0$, the Left side is chosen. When the angle is positive $\gamma_a \geq 0$, the Right side is chosen. If the targets would swap position, the action line would change direction and the opposite side of the action line would be

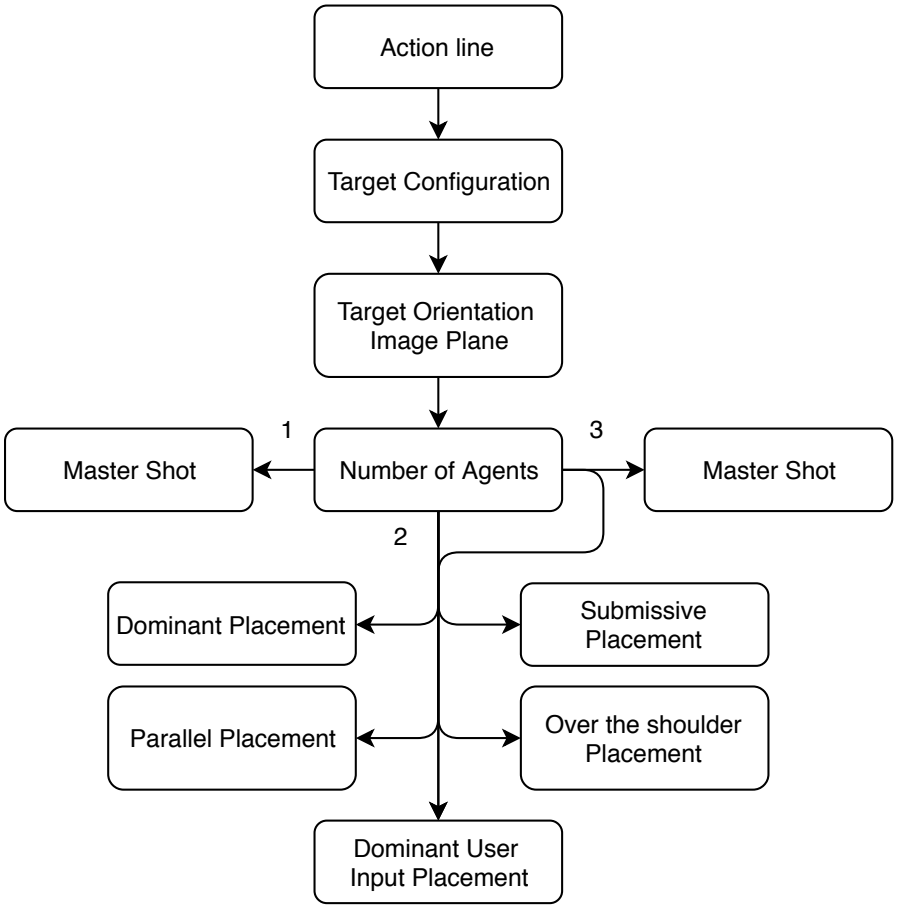


Figure 5-1: Set-point computation flowchart

chosen. However, from a global perspective the considered area to position the cameras is equal. An action range ϵ that characterises the action side is also defined. Finally, the configuration can be determined making use of the methods defined in Sections 4-3-2 to 4-3-4, respectively.

5-2 Target Orientation

With the determination of the action side, the side of the image plane the targets will be occupying needs to be determined to adhere to for continuity. The equidistant point $\mathbf{p}_{at}$ on the action line with respect to the targets is determined and a unit vector $\mathbf{n}_{a_\perp}$ in the direction of the desired action side is defined perpendicular to the action line $\mathbf{a}$. Next, vectors between the point $\mathbf{p}_{at}$ and the targets $\mathbf{p}_t^i$ are computed as $\mathbf{d}_{at}^i = \mathbf{p}_t^i - \mathbf{p}_{at}$. The vector angles γ_{at} are computed with Equation (4-8) between $\mathbf{n}_{a_\perp}$ and $\mathbf{d}_{at}^i$ for the respective target. A positive angle γ_{at} sets the target on the right side of the image

plane and a negative angle sets the target on the left side.

5-3 Placement Single Camera

This step considers the number of agents n in $\mathbf{u}_{cdc}$. For a single agent, the master shot is implemented. Given the desired image positions $\mathbf{m}_d$ as defined in Table 4-2, a master shot is determined as specified in Section 4-3-1. Making use of the distance d, camera placement $\mathbf{p}_{c_d}$ can be determined. The computations make use of the action line $\mathbf{a}$ and side, but omits the configurations of the targets. The desired size σ_d and image size σ_{I_d} is derived from $\mathbf{u}_{cdc}$. The size σ_d is projected onto the image plane using Equation (4-1). If the projected size $\sigma_I > \sigma_{I_d}$, then the size σ_d and $\mathbf{p}_{ip}$ are adapted accordingly. Otherwise, σ_d is adhered to and σ_{i_d} is adjusted.

5-4 Placement Two Cameras

The approach for camera positioning when two agents are specified in $\mathbf{u}_{cdc}$ is more complex. A set of approaches is defined to place the cameras $\mathbf{p}_{c_d}$ based on the configuration of the targets. First each camera is assigned a main target to focus on in a counter-clockwise direction on the desired action side, see Assumption 10. Next, the target status is given by Tables 4-4 to 4-6. This is then combined with the implicit or explicit placements of the cameras in accordance with $\mathbf{u}_{cdc}$. It has to be stated that the camera placement element of the $\mathbf{u}_{cdc}$ is not relevant for the Right Angle base configuration.

Given the configuration, action line $\mathbf{a}$, target orientation on the image plane and camera placement, one of five placement methods is used to position the cameras sequentially. First, an optimisation to determine a general dominant and submissive placement is presented. Next, a dominant placement is presented that can be adapted by the operator. Finally, a placement specific to the Parallel and Face to Face orientation are presented. As each camera is assigned a main target i, the initial set-up for each method is determination of the target size σ_d^i, image size $\sigma_{I_d}^i$, position $\mathbf{p}_{ip}^i$ and image position m_d^i given $\mathbf{u}_{cdc}$. The relative distance d from the target $\mathbf{p}_{ip}^i$ to $\mathbf{p}_{c_d}$ can be computed with Equation (4-1). The optimisations are solved using Sequential Quadratic Programming (SQP).

Dominant

For the placement of a dominant camera, the objective is to determine a camera position $\mathbf{p}_{c_d}$ for which the assigned target $\mathbf{p}_{ip}^i$ is at the desired image position $\mathbf{m}_d^i$, while the second target $\mathbf{p}_{ip}^j$ is occluded from view. A non-linear constrained optimisation is defined in Equation (5-1). The cost function minimizes the image position $\mathbf{m}_d^i$ of the assigned target $\mathbf{p}_{ip}^i$ and the cost term c_{vis} that specified visibility on the image plane $\mathbb{I}$. The states for the optimisation are a rotation α around the target i and the orientation of

the camera ψ_c. Limitations on the states are given by the action range ϵ, along with the specification of an initial condition $\hat{\mathbf{x}}_0$.

$$\begin{aligned} \underset{\alpha,\psi_c}{\arg\min}\ & w_\mu(\mathbf{m}_d^i - \mathbf{m}_i)^2 + w_i c_{vis} \\ \text{s.t. } & \mathbf{x}_0 = \hat{\mathbf{x}}_0 \\ & \mathbf{x} \in \begin{bmatrix} \epsilon_{min},\ \epsilon_{max} \\ \epsilon_{min} + \pi,\ \epsilon_{max} + \pi \end{bmatrix} \end{aligned} \tag{5-1}$$

Given the desired size σ_d^i and image size σ_I^i in $\mathbf{u}_{cdc}$, the magnitude of the distance d with respect to the target can be computed with Equation (4-1). The camera position $\mathbf{p}_c$ is determined by the rotation α around $\mathbf{p}_{ip}^i$ in Equation (5-2). The distance to the target in the camera frame $\mathbf{d}_{ct}^c$ is determined by rotating the global relative distance $\mathbf{p}_{ip}^i - \mathbf{p}_c$ into the camera frame making use of ψ_c, see Equation (5-3). The image position $\mathbf{m}$ can be determined using Equation (5-4) and evaluated if the target is visible c_{vis} with Equation (5-5).

$$\mathbf{p}_c = \mathbf{p}_{ip} + d_\sigma \begin{bmatrix} \cos\alpha \\ \sin\alpha \\ 0 \end{bmatrix} \tag{5-2}$$

$$\mathbf{d}_{ct}^c = \mathbf{R}_{cam}\mathbf{R}_z^T(\psi_c)(\mathbf{p}_{ip}^i - \mathbf{p}_c) \tag{5-3}$$

$$\mathbf{m}_i = \|f\|\frac{\mathbf{d}_{x,y}^c}{d_z^c} + C_{x,y} \tag{5-4}$$

$$c_{vis} = \begin{cases} |d_z^c| & \text{if } d_z^c < 0 \\ 0 & \text{otherwise} \end{cases} \tag{5-5}$$

The non-linear constraint for the occlusion of the second target $\mathbf{p}_{ip}^j$ is shown in Equation (5-6). It considers occlusion based on the angles θ of the tangent lines for each target and the view angle β of the camera. The tangent angles θ_{tan} are determined by computing the sine angle of $\|\mathbf{p}_{ip} - \mathbf{p}_c\|$ and maximum top-down radius of the target $\max\left(\frac{l}{2}, \frac{w}{2}\right)$. The angle range considered is $\theta = \theta_t \pm \theta_{tan}$, where θ_t is the angle of the target with respect to the view direction. The evaluation is made whether the second target is either outside the camera view or behind the assigned target. The relative angle with respect to the closest occlusion possibility is considered as the constraint violation.

$$c_{occ} = \begin{cases} \min\left(|-\beta - \theta_{max_j}|, |\theta_{min_i} - \theta_{min_j}|\right) & \text{if} -\beta < \theta_{max_j} \text{ and } \theta_{min_i} > \theta_{min_j} \\ \min\left(|\beta - \theta_{min_j}|, |\theta_{max_i} - \theta_{max_j}|\right) & \text{if} \beta > \theta_{max_j} \text{ and } \theta_{min_i} > \theta_{min_j} \\ 0 & \text{otherwise} \end{cases} \tag{5-6}$$

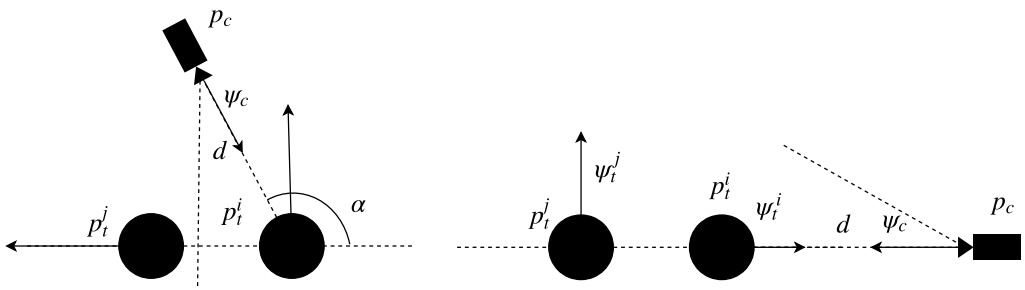


Figure 5-2: Dominant placement of Camera excluding $\mathbf{p}_{ip}^{j}$ from view

This optimisation is used for the following configurations and camera placements, see Table 5-1. For the configuration, the second value indicates whether it applies to the first or second camera to be placed into the environment. Three main initial conditions $\hat{\mathbf{x}}_0$ are stated for the rotation angle α, being the minimal action range η_{min}, the maximum η_{max} and the target orientation ψ_t^i. The initial camera orientation ψ_c is a rotation of πrad with respect to the initial rotation α. A representation of camera placements is presented in Figure 5-2.

Table 5-1: Dominant Placement Use Cases

Configuration		Camera Placement	Initial conditions $\hat{\mathbf{x}}_0$
Parallel	1.1	Extrinsic	$[\epsilon_{min}, \epsilon_{min} + \pi]$
Parallel	3.2	Extrinsic	$[\epsilon_{max}, \epsilon_{max} + \pi]$
Parallel	[1, 2, 3]	Intrinsic	$[\psi_t^i, \psi_t^i + \pi]$
Face to Face	[1.2, 4.1]	Extrinsic	$[\epsilon_{min}, \epsilon_{min} + \pi]$
Face to Face	[1.1, 4.2]	Extrinsic	$[\epsilon_{max}, \epsilon_{max} + \pi]$
Face to Face	[2, 3, 4]	Intrinsic	$[\psi_t^i, \psi_t^i + \pi]$
Right Angle	[6.1, 7.1, 8.2]	-	$[\epsilon_{min}, \epsilon_{min} + \pi]$
Right Angle	[4.2, 7.2, 8.1]	-	$[\epsilon_{max}, \epsilon_{max} + \pi]$

Submissive

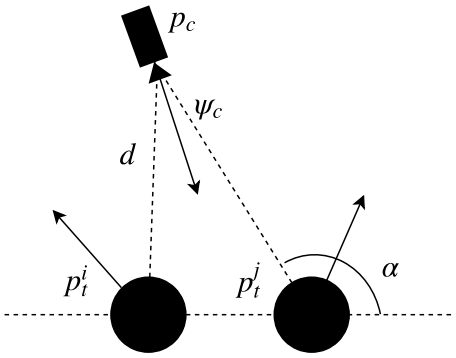
A submissive camera placement is performed using an unconstrained optimisation that tries to set both targets $k \in \{i, j\}$ on the desired image position $\mathbf{m}_{d_k}$ as specified by $\mathbf{u}_{cdc}$. The cost function is defined in Equation (5-7) and is the summation of the image position error and the target visibility c_{vis}. The states $\mathbf{x}$ for the optimisation are the rotation angle α around the assigned target $\mathbf{p}_{ip}^{i}$ and the camera orientation ψ_c. Use is made of Equations (5-2) to (5-5) and the initial condition $\hat{\mathbf{x}}_0$ to determine the camera position $\mathbf{p}_c$, image positions $\mathbf{m}$ and visibility c_{vis}.

$$\begin{aligned} \underset{\alpha,\psi_c}{\arg\min} \quad & \sum_{k=1}^{2}(\mu_{d_k} - \mu_k)^2 + \sum_{k=1}^{2} w_i c_{vis} \\ \text{s.t. } & \mathbf{x}_0 = \hat{\mathbf{x}}_0 \end{aligned} \tag{5-7}$$

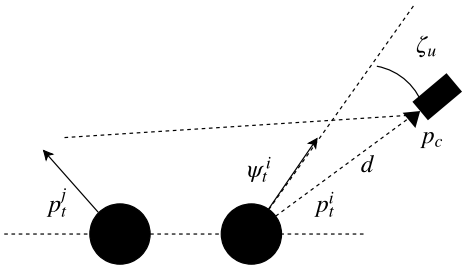
The optimisation does not guarantee that the targets are on the desired image positions $\mathbf{m}_d$. Therefore, these have to be updated given $\mathbf{p}_c$. Table 5-2 presents the cases in which the submissive optimisation is used, along with the initial conditions $\hat{\mathbf{x}}_0$ it is subjective to. For the configuration, the second value indicates whether it applies to the first or second camera to be placed into the environment. The initial angle for the rotation α is set equidistant from the extremes of the action range ϵ and the camera orientation ψ_c is an additional rotation of π rad, respectively. A possible camera placement for the submissive optimisation is presented in Figure 5-3a.

Table 5-2: Submissive Placement Use Cases

Configuration		Camera Placement	Initial conditions $\hat{\mathbf{x}}_0$
Parallel	[1.2, 3.1]	Extrinsic	$[\epsilon_{min} + \frac{\pi}{2}, \epsilon_{min} + \frac{3\pi}{2}]$
Face to Face	[2.1, 3.2, 4]	Intrinsic	$[\epsilon_{min} + \frac{\pi}{2}, \epsilon_{min} + \frac{3\pi}{2}]$
Right Angle	all	-	$[\epsilon_{min} + \frac{\pi}{2}, \epsilon_{min} + \frac{3\pi}{2}]$



(a) Submissive placement of Camera



(b) Dominant User Placement of Camera

Figure 5-3: The submissive and dominant camera placements

Dominant User Input

When considering the explicit placement of a camera, occlusion of the second target $\mathbf{p}_{ip}^j$ on the image plane $\mathbb{I}$ could become impossible while limiting the deviation from

the assigned target orientation ψ_t^i. Therefore, an angle ζ_u is defined for $\mathbf{u}_{cdc}$ such that the operator can influence the assigned target orientation on the image plane and the positioning of the second target $\mathbf{m}_j$. The camera position $\mathbf{p}_c$ is determined using Equation (5-8) given the assigned target orientation ψ_{t_i} and the relative distance d to the target computed with Equation (4-1). Depending if the camera is the first or second placed, ζ_u is either subtracted or added, respectively. The use cases are presented in Table 5-3, where the index next to the configuration specified the number of the camera to be placed in the environment. The camera placement is shown in Figure 5-3b.

$$\mathbf{p}_c = \mathbf{p}_t^i + d\mathbf{R}_z \left(\psi_t \pm \zeta_u\right) \tag{5-8}$$

Table 5-3: Dominant User Placement Use Cases

Configuration		Camera Placement
Parallel	[2]	Extrinsic
Face to Face	[2, 3]	Extrinsic
Right Angle	[1, 2, 3, 5, 4.1, 6.2]	-

Parallel

For the second configuration, where ψ_t is approximately perpendicular to the action line, this approach is applied when an explicit camera placement and a submissive target status are specified in $\mathbf{u}_{cdc}$. The following geometric approach is applied, see Figure 5-4. Known is the relative target distance $d_{ij} = \|\mathbf{p}_t^i - \mathbf{p}_t^j\|$, the camera distance d given Equation (4-1) and the angle between the two targets on the image plane $\beta_{ij} = \beta_i + \beta_j$ given Equation (4-7). Making use of the sine rule, α_1 can be computed. Using a right angle triangle, α_2 can be used to determine α_3. Finally, α_4 is determined, setting the rotation from the perpendicular line with respect to the action line $\mathbf{a}_\perp$, which defined the position of the camera $\mathbf{p}_c$. The rotation direction depends on the image position of the assigned target, being clockwise for the left target and counter-clockwise for the right target.

Face to Face

The first configuration creates an over-the-shoulder view of the second target, with a focus on the assigned target as seen in Figure 5-5. An unconstrained optimisation is constructed such that the amount of overlapping η_d of the targets on the image plane can be set in $\mathbf{u}_{cdc}$. The optimisation is presented in Equation (5-9) for which the state $\mathbf{x}$ is the rotation angle α around the assigned target. The camera position $\mathbf{p}_c$ is computed using Equation (5-2) and d_σ based on $\mathbf{u}_{cdc}$ and Equation (4-1). As the optimisation is

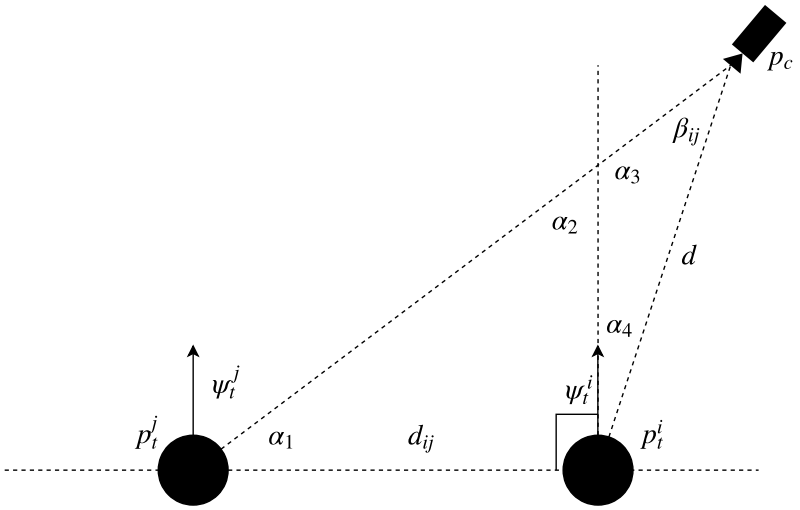


Figure 5-4: Geometric parallel placement of camera

unconstrained, the initial condition $\hat{\mathbf{x}}_0$ is the action range limit $(\epsilon_{min}, \epsilon_{max})$ closest to the assigned target orientation ψ_t^i with a slight offset δ on the determined action side.

$$\begin{aligned} \arg\min_{\alpha} \quad & (\eta_d - \eta)^2 \\ \text{s.t. } & \mathbf{x}_0 = \hat{\mathbf{x}}_0 \\ & \mathbf{x} \in \left[\epsilon_{min}, \epsilon_{max}\right] \end{aligned} \tag{5-9}$$

Based on the camera position $\mathbf{p}_c$, the relative target distance to the targets (d_{ci}, d_{cj}) is computed. The angle γ between the two targets on the image plane is computed with Equation (5-10). To compute the angles of the tangent lines with respect to the target ellipsoids, a top-down view is adhered to and only $\sigma_t = \max(l, w)$ is considered. The angles θ_{tan} are computed using Equation (5-11). Finally, the overlap η on the image plane of two closest tangent lines of the targets is computed using the desired angle β_i of the assigned images, see Equation (4-7), and Equation (5-12).

$$\gamma = \cos^{-1}\left(\frac{d_{ci}^2 + d_{cj}^2 - d_{ij}^2}{2 d_{ci} d_{cj}}\right) \tag{5-10}$$

$$\theta_i = \sin^{-1}\left(\frac{\sigma_t}{2 d_{ci}}\right) \tag{5-11}$$

$$\eta = \frac{\tan(\beta_i + \theta_i) - \tan(\beta_i + \gamma - \theta_j)}{\tan(\beta_i + \theta_i) - \tan(\beta)} \tag{5-12}$$

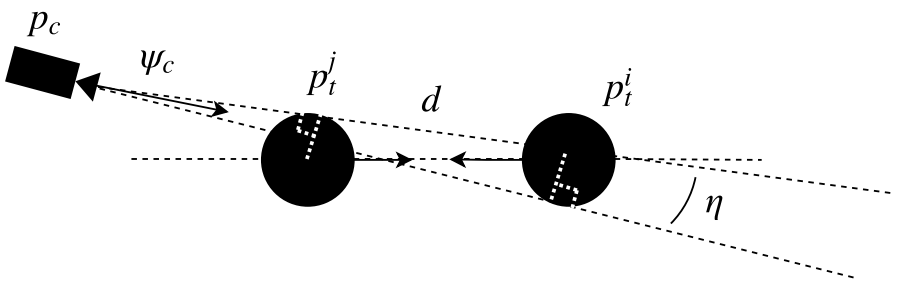


Figure 5-5: Over the shoulder view camera placement

Final Step

For all the optimisations a top-down view is adhered. The z-coordinate of the camera position is set to that of the projected image position $\mathbf{p}_{c_z} = \mathbf{p}_{ip_z}$ as a vertical image position is assumed, see Assumption 4. After the computation of the camera placements $\mathbf{p}_c$, the image position $\mathbf{m}_d$ and image size σ_{I_d} for the second target can be computed based on σ set in $\mathbf{u}_{cdc}$. A flag f_{use} is defined for the targets. It is set to true when a target is on the image plane and the relevant set-points need to be used for further computations. For dominant placement, only the assigned target is set to true. Both target flags are set to true for the submissive placements.

5-5 Placement Three Cameras

For the placement of three cameras in the environment, the single and double camera placements are combined. Making use of the master shot, there will always be a coverage of the two targets. Therefore, the setting of the target status presented in Sections 4-3-2 to 4-3-4 can be omitted, making use of $\mathbf{u}_{cdc}$. The main advantage is the possibility to set the target status of both targets to Dominant.

5-6 Feasible Setpoints

Once the ideal set-points are computed, the feasibility needs to be evaluated. A situation might arise that a position $\mathbf{p}_{c_d}$ is computed that resides outside of the flight envelope, within an obstacle or crossing the action line. Therefore, the following non-linear constrained optimisation is formulated in Equation (5-13), which is subject to collision avoidance constraints in Equations (5-14) and (5-15) and target occlusion in Equation (5-16). The cost function minimizes all camera deviations from their desired position $\mathbf{p}_{c_d}$, image position $\mathbf{m}_d$ and image size σ_I. The initial condition $\mathbf{x}_0$ is a concatenation of $\mathbf{p}_{c_d}$ and ψ_{c_d} for n cameras as set by $\mathbf{u}_{cdc}$. The position is limited to the flight

envelope $\mathbb{W}$, image position to the image plane $\mathbb{I}$ and yaw to a full rotation clockwise and counter-clockwise.

$$\begin{aligned} \underset{\mathbf{p}_c,\sigma_I,\mathbf{m}}{\arg\min} \ & \sum_{i=1}^{n} (w_{pos}\|\mathbf{p}_{c_d} - \mathbf{p}_c\| + w_\mu\|\mathbf{m}_d - \mathbf{m}\| + w_\sigma(\sigma_{I_d} - \sigma_I)) \\ \text{st. } & \mathbf{x}_0 = \begin{bmatrix} \mathbf{p}_{c_d} & \phi_{c_d} \end{bmatrix}^T \qquad \mathbf{p}_c \in \mathbb{W} \\ & \psi_c \in [-2\pi, 2\pi] \qquad\quad \mathbf{m} \in \mathbb{I} \end{aligned} \tag{5-13}$$

The collision avoidance constraint in Equation (5-14) stipulates that the agent size spheres $S_{a_{(i,j)}}$ are not allowed to intersect. The distance between camera positions $p_{c_{(i,j)}}$ has to be larger than the summed radii of the agent size r_a. This constraint is applied to all remaining cameras j with respect to camera i. Target collision avoidance, see Equation (5-15), ensures that the camera is places outside of the target ellipse Ω_t for all targets. When the desired position $\mathbf{p}_{c_d}$ is within a target, this constraint will place the feasible position $\mathbf{p}_{c_{feas}}$ on the targets ellipsoid. However, as the agents have a size sphere S_a, an intersection still occurs. Therefore, the target ellipse Ω_t is augmented by the agent size r_a to ensure that at most the boundaries connect of an agent and a target.

$$0 < \|\mathbf{p}_{c_i} - \mathbf{p}_{c_j}\| - 2r_a \qquad i \neq j \tag{5-14}$$

$$0 < \sqrt{\frac{p_{c_x} - p_{t_x}}{a + r_a} + \frac{p_{c_y} - p_{t_y}}{b + r_a} + \frac{p_{c_z} - p_{t_z}}{c + r_a}} - 1 \tag{5-15}$$

Occlusion of the image targets (i, j) due to static obstacles k is also considered for each agent. Initially, the relative distance $d = \mathbf{p}_t - \mathbf{p}_c$ to the image targets and static obstacles is rotated into the camera frame $\mathbb{C}$. Two checks are performed to evaluate whether occlusion needs to be taken into account. The first is to determine if the camera could view the static targets $d^c_{k_z} > 0$. The second is to evaluate if the obstacle is closer to the camera than either of the image targets $\|d_k\| < \{\|d_i\|, \|d_j\|\}$. When both checks have passed, a top-down view is adhered too. The angles $\alpha_{(min,max)}$ of the tangent lines to the target ellipsoid $\mathbf{h}_{tng}$ with respect to the view direction of the camera $\mathbf{r}_{view} = \mathbf{R}_{\psi_c} \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T$ making use of Equation (4-8). Finally, the constraint in Equation (5-16) is applied for each image target. It checks for overlapping of the angles $\alpha_{(min,max)}$ of each image target with respect to the considered static obstacles. Dynamic obstacles are omitted based on Assumption 2, under the premise that they are intentionally visible per operator influence.

$$c_{occ} = \begin{cases} \max\left(|\alpha_{i_{max}} - \alpha_{k_{min}}|, |\alpha_{i_{min}} - \alpha_{k_{max}}|\right) & \text{if } \alpha_{i_{max}} > \alpha_{k_{min}} \text{ or } \alpha_{i_{min}} > \alpha_{k_{max}} \\ 0 & \text{otherwise} \end{cases} \tag{5-16}$$

The optimisation is sensitive to the initial conditions, not being able to find an appropriate feasible solution if initialized with a large deviation with respect to the computed desires set-points.

Chapter 6

Assignment & Global Planner

The set-points are computed for ideal cameras and have to be assigned to physical agent. An assignment needs to be performed. Two assignment algorithms are considered, namely the Hungarian Algorithm in Section 6-1 and the Linear-Bottleneck Algorithm in Section 6-2. The performance of both algorithms is evaluated in Section 6-3. During the assignment phase, collision avoidance is not taken into account. Local and global collision avoidance will be taken into account in Chapter 7 and Section 6-4, respectively.

6-1 Hungarian Algorithm

The Hungarian algortihm is for linear assignement problems and was first presented in [28, 29]. Their assignement problem is formulates as n individuals $i \in \{1, ..., n\}$ that need to be assigned to n jobs $j \in \{1, ..., n\}$ based on a rating matrix $\mathbf{R}$, where each element is required to be positive $r_{ij} \in \mathbb{R}^+$. The objective is to maximise or minimise the sum of ratings r_{ij}, respectively. The assignment $\mathbf{X} = (x_{ij})$ is states as a bijective mapping of a finite set into itself [30]. Considering the problem at hand, the formulation can be rewritten to n cameras $i \in \{1, ..., n\}$ that need to be assigned to n agents $j \in \{1, ..., n\}$. A cost matrix $\mathbf{C}$ is determined based on the relative distance between current agent position $\mathbf{p}_a$ and feasible camera position $\mathbf{p}_{c_d}$. The minimisation is stated in Equation (6-1), minimising the total distance to be travelled by all agents. The cost is restricted to the relative distance $\|\mathbf{p}_{c_d}^i - \mathbf{p}_a^j\|$, omitting image set-points to minimise computational complexity of the cost terms. Also, they are accounted for during set-point computations and will be taken into account when computing agent inputs $\mathbf{u}_a$. Addition to the relative distance due to collision avoidance is also omitted, as this would require the computation of a collision-free trajectory for each agent to each camera positions, which is computationally expensive.

$$\begin{aligned} &\min_{i,j} \sum c_{ij} x_{ij} \\ &\text{s.t.} \sum_{i=1}^{n} x_{ij} = 1, \quad 1 \geq j \geq n \\ &\sum_{j=1}^{n} x_{ij} = 1, \quad 1 \geq i \geq n \\ &x_{ij} \in \{0, 1\}, \quad 1 \geq (i, j) \geq n \end{aligned} \tag{6-1}$$

Implementation

A description of the algorithm implementation can be found in [28, 29, 31]. The starting point is the cost matrix $\mathbf{C}$. The first step is determining the minimum entry c_j for each column and subtracting this from each entry in the column. The same approach is applied to each row c_i, respectively. Duality allows for the modification to $\mathbf{C}$ to occur, stating that the assignment remains unaltered by the subtraction of constants. This process creates zero entries within the cost matrix $\mathbf{C}$. The summation of the subtracted minima is the solution of the problem if an assignment is found. The columns and rows are evaluated to determine an unique assignment based on the zero entries. If there are insufficient zero entries, the assignment of entries that provide a minimal addition to the total cost are selected.

It has to be noted there is a possibility that the amount of agent n exceeds the number of computed camera positions m. This creates a rectangular cost matrix $\mathbf{C}$. The Hungarian algorithm works only on square matrices. In this situation, the largest cost index $c_{ij_{max}}$ is taken and inserted in each entry of the additional rows or columns required to create a square matrix. As minimization is the objective, the added entries will always be the largest. Therefore, these will not be considered in the solution assignment.

6-2 Linear-Bottleneck Algorithm

Where the Hungarian Algorithm only takes into account the relative distance $\|\mathbf{p}_{c_d}^i - \mathbf{p}_a^j\|$ between the cameras and the agents the Linear-Bottleneck Algorithm (LBA) allows for the additional consideration of time t. The general formulation for this assignment problem is the minimisation of the maximum cost $\sum c_{ij}$ given the assignment $\mathbf{X} = (x_{ij})$ [32, 33]. Relating this to the camera assignment to agents, it can be formulated as minimising the overall time required for the agents n to achieve the camera positions $\mathbf{p}_{c_d}$ given velocity limits $\mathbf{v}_a^j$, while maximising the total distance travelled by all agents. This will result in an assignment where agents travel an equivalent distance. The term bottleneck is used, because the assignment is limited by the time required for the slowest agent to achieve its goal.

Considering the LBA from a cinematographic perspective, maximising the total distance travelled by all agents is not desired. Consider the scenario where there are three adjacent agents for which two camera positions $\mathbf{p}_{c_d}$ are on the current position $\mathbf{p}_a$ with a third position $\mathbf{p}_{c_d}$ further away. The assignment will cause one agent to manoeuvre to the furthest position, while the two remaining exchange positions as this maximises the distance. This behaviour is not desired, preferring the agents to maintain their viewing angles when possible. Therefore, the assignment problem for the LBA is formulates in Equation (6-2) as finding the assignment that minimised the total time t required, while subsequently minimising the distances required.

$$\begin{aligned} &\min_t \min_{ij} \sum c_{ij} x_{ij} \\ &\text{s.t.} \sum_{i=1}^{n} x_{ij} = 1, \quad 1 \geq j \geq n \\ &\sum_{j=1}^{n} x_{ij} = 1, \quad 1 \geq i \geq n \\ &x_{ij} \in \{0, 1\}, \quad 1 \geq (i, j) \geq n \\ &t = \frac{c_{ij}}{\|\mathbf{v}_a^j\|} \end{aligned} \tag{6-2}$$

Implementation

The implementation of the algorithm is present in Algorithm 2. The input is a cost matrix $\mathbf{C}$ and the assumption that in this implementation all agents are homogeneous, i.e. the limit velocity $\mathbf{v}_a$ is equivalent. The first step is to find the maximal element c^* given all the minima of the rows and columns. If an assignment is found, this is equivalent to the minimum time it would take for the system to achieve the objective. Each element $c_{ij} > c^*$ is set to infinity. Given this modification to the cost matrix $\mathbf{C}$, the Hungarian Algorithm is run to find an assignment that minimised the total distance. If ans assignment cannot be found, this process is repeated with the next $c_{ij} > c^*$ until an assignment is found.

6-3 Assignment Comparison

Assignment of camera positions $\mathbf{p}_{c_f}$ is performed when the operator specifies the use of multiple agents $n > 1$ in $\mathbf{u}_{cdc}$. Although for only a few agents, the assignment is not computationally expensive, it is still essential to achieve set-points in either a minimal distance, Hungarian algorithm, or minimum time and distance, LBA. As the cost determination for the assignment methods is almost identical, the overall performance is most

Algorithm 2 Linear-Bottleneck Algorithm [34]

function LBA(C)
  Input: Cost matrix $\mathbf{C} = [c_{ij}]$
  Output: Assignment agn
  $c_{sort} = sort(\mathbf{C})$
  $c^* = \max(\min_i c_{ij}, \min_j c_{ij})$
  while $sol = false$ **do**
    $\mathbf{C}_{mod} = \mathbf{C}$
    if $c_{ij} > c^*$ **then**
      $c_{mod_{ij}} = \infty$
    end if
    $[agn,\ sol] = HunAlg(\mathbf{C}_{mod})$
    if $sol = false$ **then**
      $c^* = find(c_{sort} > c^*)$
    end if
  end while
end function

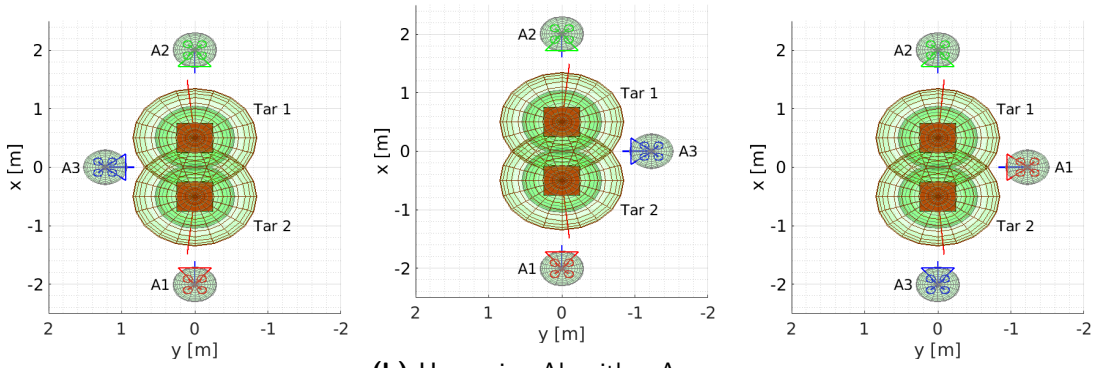
situations is equal. However, to illustrate there are differences, the following examples are presented making use of three agents.

The first example illustrates the fourth Face to Face configuration in Figure 6-1. A master shot is defined, along with two dominant placements set by $\mathbf{u}_{cdc}$ that each focus on a target. An action side shift is induced to generate new set-points for the agents that result in the distance matrix $\mathbf{C}$ in Table 6-1. The Hungarian algorithm determines a cost $\sum c_{ij}x_{ij} = 2.45$ with assignment (3, 1, 2). The first and second agent are already on their set-points, while the third agent has to transition past the first agent to reach the set-point, see Figure 6-1b. Assuming a homogeneous velocity, the bottleneck elements are $c^* = 2.35$ which sets all higher values to infinity. The assignment becomes (1, 3, 2) with a cost $\sum c_{ij}x_{ij} = 4.702$. This results in the third agent shifting from a master shot to the dominant position of the third target and the first becomes the new master shot, see Figure 6-1c.

Table 6-1: Face to Face assignment information

Initial Positions $\mathbf{p}_a$			Feasible Positions $\mathbf{p}_{c_f}$			Distance Matrix $\mathbf{C}$		
A1	A2	A3	1	2	3			
$\begin{bmatrix}-2.01\\0.00\\1.43\end{bmatrix}$	$\begin{bmatrix}2.01\\0.00\\1.43\end{bmatrix}$	$\begin{bmatrix}-1.47\\0.90\\1.54\end{bmatrix}$	$\begin{bmatrix}0.00\\1.23\\1.48\end{bmatrix}$	$\begin{bmatrix}-2.01\\0.00\\1.43\end{bmatrix}$	$\begin{bmatrix}2.01\\0.00\\1.43\end{bmatrix}$	$\begin{bmatrix}2.351 & 0.000 & 4.011\\2.351 & 4.011 & 0.000\\2.452 & 2.351 & 2.351\end{bmatrix}$		

The second example presents the seventh Right Angle configuration. A Submissive target status is defined for the first target and Dominant for the second in $\mathbf{u}_{cdc}$ as depicted in



(a) Initial configuration **(b)** Hungarian Algorithm Assignment **(c)** LBA Assignment

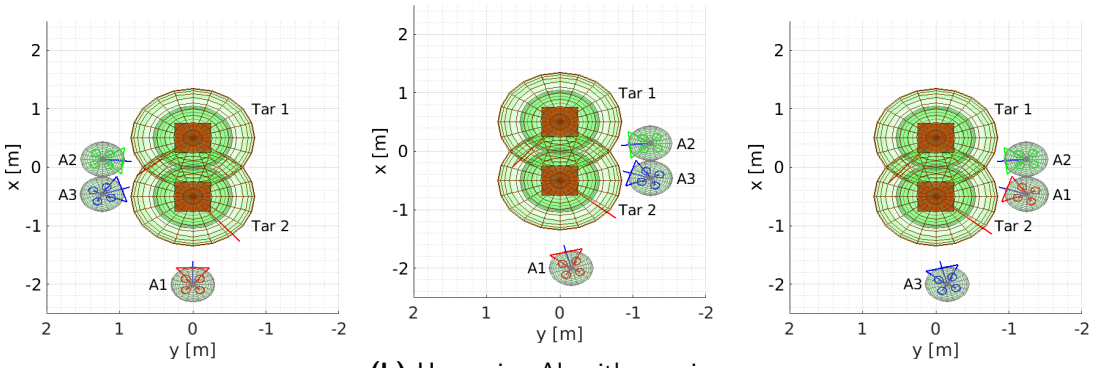
Figure 6-1: The fourth Face to Face configuration to demonstrate Hungarian Algorithm and LBA assignment

Figure 6-2a. A slight change in orientation could cause the action line to shift. The distance matrix $\mathbf{C}$ can be found in Table 6-2. The Hungarian Algorithm determines the assignment (2, 1, 3) with a cost $\sum c_{ij}x_{ij} = 4.956$. It assigns the second and third agent to the new set-points, while maintaining the position of the first agent, see Figure 6-2b. Similar to the face to face example, the third agents had to traverse through the view of the first agent. The LBA in Figure 6-2c determines that $c^* = 2.473$ is the limiting distance regarding time. However, an assignment cannot be determined and the next higher value $c^* = 2.474$ is used. This results in the assignment (2, 3, 1) with $\sum c_{ij}x_{ij} = 6.436$. The first and third agent shift around the second target. For both assignments, the second agent has to manoeuvre around the first target.

Table 6-2: Right Angle assignment information

Initial Positions $\mathbf{p}_{a_{init}}$			Revised Positions $\mathbf{p}_{c_{feas}}$			Distance Matrix $\mathbf{C}$		
A1	A2	A3	1	2	3			
$\begin{bmatrix} -2.01 \\ 0.00 \\ 1.42 \end{bmatrix}$	$\begin{bmatrix} 0.13 \\ 1.24 \\ 1.49 \end{bmatrix}$	$\begin{bmatrix} -0.46 \\ 1.24 \\ 1.41 \end{bmatrix}$	$\begin{bmatrix} 0.13 \\ 1.24 \\ 1.49 \end{bmatrix}$	$\begin{bmatrix} 2.01 \\ 0.00 \\ 1.43 \end{bmatrix}$	$\begin{bmatrix} -0.46 \\ -1.24 \\ 1.42 \end{bmatrix}$	$\begin{bmatrix} 2.473 & 0.001 & 1.981 \\ 2.474 & 2.473 & 2.550 \\ 2.549 & 1.981 & 2.482 \end{bmatrix}$		

Based on the examples, the LBA results in a higher cost. However, the field of view crossing by agents is reduced. An argument can be made that the Hungarian maintains the master shot with the same agent for continuity. The draw-back is that the agent is unable to maintain the master shot during a transition to the desired action side. LBA shifts the master shot to an other agent, allowing for it to be achieved faster. Although having alternative agents be a master shot decreases the continuity of the recording, it is preferred over agents crossing field of view. Therefore, the decision is made to make use of LBA for assigning n camera positions $\mathbf{p}_{c_d}$ to m agents.



(a) Initial configuration

(b) Hungarian Algorithm assignment

(c) LBA assignment

Figure 6-2: The seventh Right Angle configuration to demonstrate Hungarian Algorithm and LBA assignment

6-4 Global Planner

Up until this point, the local environment relative to the targets has been considered for the agents to function in. To ensures collision-free motion through the environment, a global trajectory is required for each agent. The A* graph search algorithms, first presented in [35], is used to determine a collision-free trajectory through the environment. The environment is sampled, which creates a list of nodes n through which an optimal path needs to be determined based on the cost function in Equation (6-3). The cost function considers to current cost $g(n)$ for the start node n_{start} to n and the optimal cost $h(n)$ to the goal n_{end}. At each step it evaluates the nodes surrounding it and selects the node that provides a minimal addition to the cost function until the goal is reached. The algorithm has gained popularity for robot path planning, where examples for UAVs can be found in [36, 37].

$$f(n) = g(n) + h(n) \tag{6-3}$$

The initial step is the sampling of the three-dimensional environment into cubic nodes with dimensions δ. Each node n will represent a feasible space an agent can occupy and traverse. A node n will have 26 connections to its neighboring which creates the graph for the A* algorithm to search for an optimal trajectory. The number of nodes is determined by the dimensions of the flight envelope $\mathbb{W}$. The nodes containing an intersection with the obstacle set $\mathbb{O}$ and the $\mathcal{A}^j(\mathbf{p}_a^j)$ agent set, excluding the current agent considered, are omitted from the graph.

This evaluation is performed each cycle to account for the dynamic motion of the targets and agents. Nodes containing static obstacles are excluded at the initialization of the feasible node list. For safety, the height of targets h is extended to the vertical dimension

of the flight envelope $z_{w_{max}}$ to prevent the planning of a trajectory over the target heads. Also, to prevent a trajectory between the two targets considered in the set-point computation, a plane is defined along the action line. It is constrained by the respective target positions $\mathbf{p}_t$. An additional plane intersection for the nodes is performed to exclude these nodes. The starting node contains the current position of the agent $\mathbf{p}_a$. The goal node contains the assigned agent position $\mathbf{p}_{a_f}$. In the event the goal position resides in an excluded node, the nearest feasible node is set as the goal node. An example of the sampling of the environment is presented in Figure 6-3 in which the a static obstacle and two targets are considered.

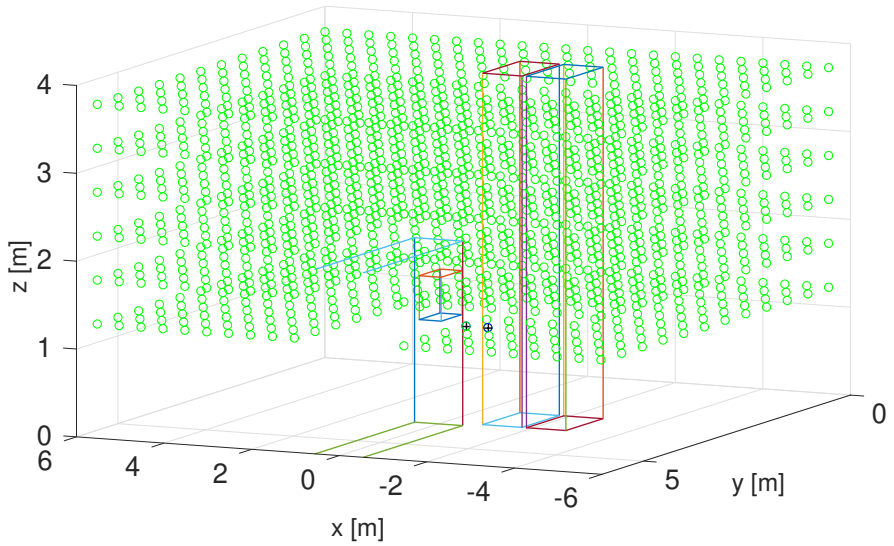


Figure 6-3: Sampling an environment containing two targets and a static obstacle.

Given the list of nodes, a cost function is defined and given in Equation (6-4). The cost function only takes into account the relative distance between the current node and the gal node. As the camera orientation, image placement of the targets and the size are already taken into account by the set-point computations and the MPC, it is sufficient to only take into account the distance. Also, reduction of complexion within the cost function allows for smaller computation time.

$$f = n - n_{goal} \tag{6-4}$$

The output of the algorithm is a list of nodes connecting the start to the goal node. It would be possible to supply this list as way-points for the agents to follow. However, this will create jagged movement. Therefore, a four-order polynomial is fit to the list to create a continuous and smooth trajectory. The value of the terminal node is substituted for the assigned position $\mathbf{p}_{a_f}$.

The trajectory computation is performed at a lower frequency than the set-point computation as coarse trajectory is sufficient for global navigation. The local navigation will be provided by the MPC. Between computations, the current position of the agent is projected onto the trajectory. From this point, a section of the trajectory is determined based on the time horizon N of the MPC and the velocity $\mathbf{v}_{lim}$ of the agent. If the trajectory terminates prior to the time horizon length, the feasible agent position $\mathbf{p}_{a_f}$ is augmented to achieve the required length.

A final evaluation is performed for each point in the trajectory section. Each set-point is evaluated with respect to the action line $\mathbf{a}$. The global angle γ of the vector from the action point to the way-point is compared with the action range η. In the event that the agent is on the incorrect side of the action line $\mathbf{a}$, the target $\mathbf{p}_t^i$ closest to the agent when the trajectory crosses the action line is determined. This target is used as a focal point during crossing, while the alternative target $\mathbf{p}_t^j$ is neglected until the correct action side is achieved. This neglection is achieved by setting the use of the alternative target to zero $f_{use} = 0$. It results in the target not being taken into account in th MPC.

Chapter 7

Model Predictive Control

Model Predictive Control, also known as receding horizon control, is a control method that solves an open-loop optimal control problem for a finite time-horizon online given the current state of the system. This creates a sequence of input commands for the system for a set time-horizon. Differing from offline control methods, the first input is implemented on the system creating a new measurement of the state of the system [38]. Closed-loop feedback is achieved by solving the optimal control problem at the next time step given the new measured initial condition [39].

Given the assigned feasible set-points for each agent, along with the global path, control commands $\mathbf{u}_a$ for each agent need to be computed. Use is made of the MPC presented in [1, 2] to compute the control inputs $\mathbf{u}_a$. The original formulation as stated in [1] is presented in Equation (7-1). The state of the system $\mathbf{x}$ are the states defined in Section 2-1, along with the agent dynamics. A prediction horizon $\Delta_t N$ is used for optimization. Limitations are set for the states $\mathbf{x}_k \in \mathcal{X}$ and inputs $\mathbf{u}_k \in \mathbb{U}$ for any given time step k. Weights (w, λ) can be set by the operator at run-time to adjust the importance of framing constraints. Soft collision avoidance constraints are applied to account for uncertainties in position determination of agents. A slack variable $s_{coll} \geq 0 \in \mathbb{R}$ is used to ensure collision avoidance, along with a slack variable for maintaining the flight envelope $s_{env} \geq 0 \in \mathbb{R}$.

$$\min_{\mathbf{x},\mathbf{u},\mathbf{s}} \; w_N^T c\left(\mathbf{x}_N, \mathbf{u}_N\right) + \sum_{k=0}^{N-1} w^T c\left(\mathbf{x}_k, \mathbf{u}_k\right) + \lambda \|\mathbf{s}_k\|_\infty \tag{7-1}$$

$$\begin{aligned}
\text{s.t. } & \mathbf{x}_0 = \hat{\mathbf{x}}_0 \\
& \mathbf{x}_{k+1} = f\left(\mathbf{x}_k, \mathbf{u}_k\right) \\
& r_{ct}^T \Omega_s r_{ct} > 1 - \mathbf{s}_k \\
& r_{ct} = g\left(\mathbf{x}_k\right) \\
& \mathbf{x}_k \in \chi \\
& \mathbf{u}_k \in \mathcal{U} \\
& \mathbf{s}_k \geq 0
\end{aligned}$$

The original formulation makes the consideration that the defined obstacle set can also function as targets, since an obstacle can be treated as a target with only the collision avoidance constraints active [1]. During computations, all cost terms are computed for all obstacles and only the relevant cost terms are taken into account. A revision of this approach is implemented, separating the cinematographic and collision avoidance cost terms. This reduces the computations performed, only computing the cinematographic cost terms for the user selected targets. Collision avoidance is computed for all obstacles, including the selected targets.

The slack variable element of the cost function $\lambda\|\mathbf{s}_k\|_\infty$ in Equation (7-1) states that only the largest slack variable $s_{k_{max}}$ is taken into account. The actual implementation is a summation of the weighted slack variables $\sum \lambda \mathbf{s}_k$, taking all the slack variables into account.

7-1 Cost Terms

The minimization function consists of six main aspects that contribute to the overall cost, namely: collision avoidance, the input $\mathbf{u}_a$, agent position $\mathbf{p}_a$, target size on the image plane σ_I, target position on the image plane m and mutual visibility of agents. In [1], a cost term is also implemented to minimize the occlusion targets have on each other. This term is omitted given Assumption 2, assuming there is a degree of guidance on the obstacles within the environment. The set-point computations in Chapter 5 makes use of intentional target occlusion in a few desires set-point determinations and the feasibility optimization takes static obstacle occlusion into account. Any occlusion due to dynamic obstacles is a style choice of the operator or an occurrence of unforeseen circumstances.

Collision Avoidance

The first cost term is for collision avoidance c_{coll}, see Equation (7-2). For safe operation, this term is always active for all agents and obstacles. A potential field is mimicked

around each instance, which is activated once an agent enters the boundary. This terms helps to maintain a safe distance from moving obstacles and leaves some buffer for the un-modelled dynamics of the agent and target motion.

$$c_{coll} = \sum w_{coll_a} c_{coll_a} + \sum w_{coll_t} c_{coll_t} \tag{7-2}$$

A buffer range r_b is defined and summed to the radius of the agent or radii of the ellipsoid, $r_{pot} = r_a + r_b$ and $(a, b, c) = (l_t, b_t, h_t) + r_b$, respectively. To compute the cost for inter-agent collision, the intersection of the potential field spheres is computed, see Equation (7-3). The obstacle collision avoidance is computed by the intersection of the agent and the obstacle ellipsoid in Equation (7-4)

$$\begin{aligned} c_{coll_a} &= \begin{cases} d_{coll}^2 & \text{if } d_{coll_a} < 0 \\ 0 & \text{otherwise} \end{cases} \\ d_{coll_a} &= \|\mathbf{p}_{a_i} - \mathbf{p}_{a_j}\| - 2r_{pot} \end{aligned} \tag{7-3}$$

$$\begin{aligned} c_{coll_t} &= \begin{cases} d_{coll_t}^2 & \text{if } d_{coll_t} < 0 \\ 0 & \text{otherwise} \end{cases} \\ d_{coll_t} &= \sqrt{\frac{(p_{a_x} - p_{t_x})^2}{a^2} + \frac{(p_{a_y} - p_{t_y})^2}{b^2} + \frac{(p_{a_z} - p_{t_z})^2}{c^2}} - 1 \end{aligned} \tag{7-4}$$

Input

Although not mentioned in [1, 2], a cost term c_{input} is set on the inputs for an agent $\mathbf{u}_a$. This term is introduced to minimize the inputs to prevent sudden or aggressive behaviour of the agents caused by abrupt change in the input. When operating around humans, this behaviour can compromise safe operation. The cost term is presented in Equation (7-5).

$$c_{input} = w_{\theta,\phi} \|\mathbf{u}_{\mathbf{a}_{\theta,\phi}}\| + w_{v_z} u_{v_z}^2 + w_{\dot{\psi}} u_{\dot{\psi}}^2 \tag{7-5}$$

Image Position

The first cinematographic cost term is the image position of a given target, see Equation (7-6). The relative position rotated into the camera coordinate frame r_{ch}^c between

the agent and the target is computed. The cost term is computed to be the error d_m with respect to a desired image position $\mathbf{r}_d^c$.

$$c_{image} = \sum w_\mu \|d_m\| \tag{7-6}$$

$$\mathbf{d}_m = \frac{\mathbf{r}_{ch}^c}{\|\mathbf{r}_{ch}^c\|} - \frac{\mathbf{r}_d^c}{\|\mathbf{r}_d^c\|} \qquad \mathbf{r}_d^c = \begin{bmatrix} \mathbf{m}_d \\ 1 \end{bmatrix} \qquad \mathbf{r}_{ch}^c = \mathbf{R}(\psi_c)^T \mathbf{R}_{cam} \cdot (\mathbf{p}_t - \mathbf{p}_c)$$

Size

The second cinematographic cost term is the size of a given target on the image plane σ_I. In the original formulation, the head size h_h is projected onto the image plane and the desired size σ_d formulated accordingly. This approach is revised and use is made of a proportional size of the target as desired by the user input $\mathbf{u}_{CDC}$, see Table 4-1. Use is made of Equation (4-1) to compute the image size σ_I.

$$c_{size} = \sum w_\sigma \|\sigma_I - \sigma_d\| \tag{7-7}$$

Position

With the image parameters taken into account, the position of the agent $\mathbf{p}_a$ in the global coordinate system can be taken into account with respect to the global trajectory for each time-step k. The cost term c_{pos} is presented in Equation (7-8). It is revised, whereby the vertical direction p_{a_z} is considered separate to discourage vertical movement during collision avoidance, preferring horizontal movement.

$$c_{pos} = w_{pos}\|\mathbf{p}_d - \mathbf{p}_a\| \quad \rightarrow \quad c_{pos} = w_{pos_{xy}}\|\mathbf{p}_{d_{xy}} - \mathbf{p}_{a_{xy}}\| + w_{pos_z}\|p_{d_z} - p_{a_z}\| \tag{7-8}$$

Mutual Visibility

When utilizing multiple agents within an environment, it is undesirable for an agent to be visible on the image planes of other agents. Therefore, a cost term c_{mv} for mutual visibility between pairs of agents is presented in [2]. The cost term is presented in Equation (7-9). Initially, the view vector of the agent $\mathbf{r}_{view}$ is determined in the global reference frame $\mathbb{W}$, along with the relative distance between agents $\mathbf{r}_j$. The intersection vector $\mathbf{p}_{int}$ is computed, after which the intersection d_{surf} of the viewing cone and the agents sphere can be computed. Two main alterations have been performed with respect to the original formulation regarding the computation of d_{surf}. The first is the added consideration of the agent dimensions r_a, where initially only the centre was taken into

account $\mathbf{p}_{a_j}$. The second is the change of $\|\mathbf{r}_j - \mathbf{p}_{int}^T \mathbf{r}_{view}\|$ into $\|\mathbf{r}_j - \mathbf{p}_{int}\|$. The former attempts to subtract the multiplied norm of the codirectional vectors $\mathbf{p}_{int}^T \mathbf{r}_{view}$ from the relative distance r_j. This results in a dimensional mismatch. By performing the latter implementation, the relative vector between the intersection and the relative distance.

$$c_{mv} = \sum w_{mv} c_{mv_j} \tag{7-9}$$

$$c_{mv_j} = \begin{cases} d_{surf}^2 & \text{if } d_{surf} < 0 \\ 0 & \text{otherwise} \end{cases} \tag{7-10}$$

$$\begin{aligned} d_{surf} &= \|\mathbf{r}_j - \mathbf{p}_{int}\| - r_{cone} - r_a & \mathbf{p}_{int} &= c_{int} \mathbf{r}_{view} \\ r_{cone} &= \frac{\max(C_x, C_y)}{\max(f_x, f_y)} & c_{int} &= \mathbf{r}_j^T \mathbf{r}_{view} \\ \mathbf{r}_{view} &= \mathbf{R}_z(\psi_a) \mathbf{R}_{cam}^T \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T & \mathbf{r}_j &= \mathbf{p}_{a_j} - \mathbf{p}_a \end{aligned}$$

When agents are operating on the same action side, this cost term will cause a slight deviation from the feasible set-points. However, when a transition of action side occurs, there is a high probability that the agent will cross on opposite sides of the considered targets. This will cause the visibility of both agent on the respective image planes. If this cost term were active, this would cause unpredictable or undesired behaviour whereby the target framing and agent orientation are not achieved at the cost of preventing mutual visibility. Therefore, mutual visibility is only taken into account when the agent pair are both on the correct side of the action line. An addendum to this is that mutual visibility is not taken into account for the placement of dominant agent for the fourth face to face configuration when considering three agents. The reason is that occlusion due to obstacles is not taken into account.

7-2 Implementation

Each cost term for the MPC considers a distinct aspect of autonomous cinematography and the relative importance needs to be set in decreasing order of importance. Collision avoidance for the agents is of paramount importance, as safe operation has to be achieved around people. To ensure the inner collision avoidance constraints are maintained, the slack variable weight λ has the highest penalty. Next, to maintain a minimum safe distance from people the target collision weight w_{coll_t} has the highest value. This is followed by inter-agent collision avoidance w_{coll_a}. The image position $w_\mu F$ is the fourth as this is the main focus of the cinemagrahic application. Fifth is the mutual visibility w_{mv} of agent on the image plane which needs to be avoided, but not to the detriment of collision avoidance. To ensure that the global position set-points are followed when a shift in action side occurs or dynamic situation, the position weight $(w_{pos_{xy}}, w_{pos_z})$. The

latter is set slightly higher to set a preference to horizontal movement around an obstacle to closer maintain the desired image set-points. The vertical velocity input weight w_{v_z} adheres to the same logic. The weight for the angle inputs $w_{\theta,\phi}$ set so that larger deviations from the zero input are discouraged and therefore reducing unpredictable behaviour. The penultimate weight is on the yaw rate $w_{\dot{\psi}}$ as the orientation ψ_a is implicitly taken into account by the image positioning of the targets. The final weight is the size of the targets on the image plane w_σ as this term is expressed in pixels and therefore magnitudes larger than the other terms that are largely expressed in meters, radians and radians per second. The quantities used for the weights can be found in Table B-1.

Chapter 8

Results

Three simulations are performed to evaluate the capability, flexibility and limitations of the system. Information pertaining to the set-up of the simulations is presented in Section 8-1. The first simulation is set up to showcase camera configurations for the agents when the targets have a fixed position and variate their orientations, see Section 8-2. The second simulation considers a dynamic environment in which a pair of targets perform a set of manoeuvres with obstacles present in Section 8-3. The third simulation will showcase the performance of the system when multiple groups of targets are considered and a larger number of agents is used, see Section 8-4.

8-1 General Set-up

The simulations are performed on a standard desktop PC running Ubuntu 16.04 with a Quadcore Intel i5 CPU @ 3.2GHz. Two instances of MATLAB are utilised for the simulations, where communication is performed through ROS [40]. One instance runs the visualisation tool along with the set-point computations. The second instance computes the inputs $\mathbf{u}_a$ for n agents sequentially using the MPC. For formulating and code generation of the MPC optimization problem, use is made of FORCES Pro [41, 42] to generate code for solving in real-time.

The agents are modelled to represent the Bebop Parrot 2[1] quadcopters. The sphere $\mathbf{S}_a$ has a radius $r_a = 0.2$m and a collision buffer $r_{coll} = 0.3$m. The parameters for the agent dynamics are presented in Table 8-1, along with the drag coefficient $c_d = 0,28$. The camera parameters are set to $f_x = 529.8$px, $f_y = 519.9$px, $C_x = 428.1$px and $C_y = 240.6$px [1].

[1] Bebop Parrot 2, www.parrot.com/us/drones/parrot-bebop-2 (accessed: 30 Oktober 2018)

Table 8-1: Parameters for Agent Dynamics [23]

Name	Variable			
Time Constant τ	$\tau_\theta = 0.57$	$\tau_\varphi = 0.54$	$\tau_{v_z} = 1.88$	
Gain k	$k_\theta = 1.61$	$k_\varphi = 1.45$	$k_{v_z} = 1.57$	
Limits	$\lvert\theta_u\rvert \leq 15^\circ$	$\lvert\phi_u\rvert \leq 15^\circ$	$\lvert v_z\rvert \leq 1\text{m}\,\text{s}^{-1}$	$\lvert\dot{\psi}_u\rvert \leq 120^\circ\,\text{s}^{-1}$

The values for the variables in $\mathbf{u}_{cdc}$ that need to be inferred when not specified are the vertical buffer $\mu_{y_{buff}} = 150$px, single target angle for front view $\alpha_f = 20^\circ$, angle for dominant positioning $\zeta_u = 0^\circ$ and overlapping factor $\eta = 0.2$. The MPC makes use of a time horizon $N = 20$, $\Delta t = 0.05$s and weights defined in Table B-1. The simulation computes $\mathbf{u}_a$ for the agents at 20Hz. The A* search algorithm samples the environment with $\delta = 0.5$m and is run at a frequency of 2Hz. Given the limiting velocity $v_{lim} = 1\text{m}\,\text{s}^{-1}$, this accounts for a transition to a new node every computation cycle.

Simulation: Dynamic Target Orientation

The first simulations consists of two static targets in an obstacle-free environment to demonstrate the performance of the set-point computations, assignment and global planner. The flight envelope is defined as $x \in [-5, 5]$, $y \in [-5, 5]$ and $z \in [1, 4]$, whereby the assumption is made that the agents start in flight after a vertical take-off. The position of the targets is set at $p_t^1 = [0.5\ 0\ 1.9]^T$ and $p_t^2 = [-0.5\ 0\ 1.9]^T$. The dimensions for the ellipsoid are defined as $(l, w) = 0.5$m, $h = 1.9$m, along with a head radius of $h_h = 0.15$m. The radius of the potential field is set at $r_{pot} = 0.6$ and the minimum safety radius of $r_{coll} = 0.3$. An orientation ψ_t profile is defined for each target and can be found in Appendix B-3. The collective orientations cycles through a range of configurations from Section 4-3. Each configuration is held for $t = 5$s, after which a transition of $t = 2$s occurs to the next configuration. The operator command $\mathbf{u}_{cdc}$ states the minimal required information of $n \in \{1,\ 2,\ 3\}$ agents that needs to track two targets, namely Target 1 and Target 2. Additions to the command $\mathbf{u}_{cdc}$ are introduced when applicable.

Simulation: Dynamic Motion

The second simulation illustrates the performance of the collision avoidance and the limitations of a rapid succession of configurations the agents have to achieve. The workspace is extended to $x \in [-6, 6]$, $y \in [-6, 6]$ and $z \in [1, 4]$. Two targets with the same dimensions as above manoeuvre through the environment. A dynamic obstacle mimicking a person heading in an opposite direction with respect to the main focal targets is defined for the first section of the simulation. An additional static obstacle is defined for static collision avoidance at $p_t^2 = [-0\ 5\ 1.9]^T$ with $l = 0.3$m, $w = 2.0$m and $h = 1.7$m. The trajectories through the environment and the orientation can be found in Appendix B-4.

Simulation: Multiple Commands

The third simulation extends the functionality of the system to demonstrate the performance when a sequence of operator commands is specified, collectively focussing on more that two targets. A command is limited to two targets, but this limitation can be overcome with multiple commands. The simulation consists of five targets mimicking a situation where people would be interacting with each other. The system will process the commands sequentially based on the order they are specified in. All targets have the dimensions $(l, w) = 0.5$m and $h = 1.9$m. The commands and trajectories can be found in Appendix B-5.

8-2 Simulation: Dynamic Target Orientation

The simulation is performed threefold, increasing the number of agents for each instance. A situational representation of the simulation is presented in Figures 8-1 and 8-2 for a single agent and two agents, respectively. The targets are represented by a red bounding box, along with the target ellipsoid Ω_t in green and potential field in lighter green. The orientation ψ_t of each target is represented by a red line. The agents, each with a distinctive colour, have their potential field show in green, orientation ψ_a as a blue line and a structure representing the viewing angles of the agent.

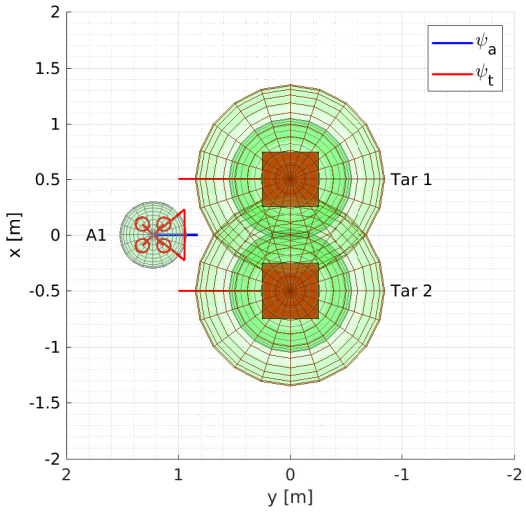


Figure 8-1: Example of single agent for parallel configuration

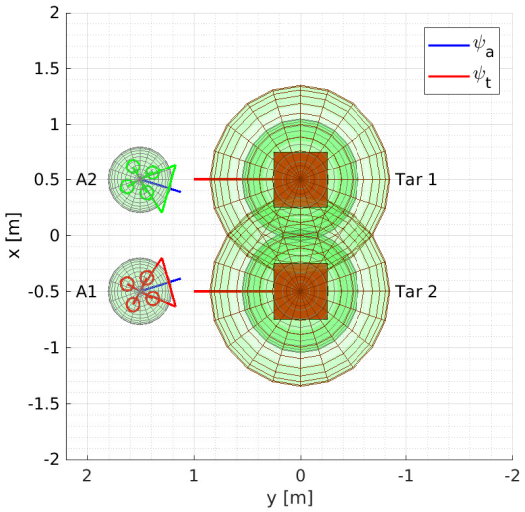


Figure 8-2: Example of two agents for parallel configuration

Single agent

The single agent functions as a master shot throughout the simulation, only taking the side of the action line into account. It does not consider the base and specific configuration of the target, only on which the side the base configuration occur. During the defined orientation cycles, the action side changes four times. Each change requires the agent to perform a manoeuvre. The Root-mean Squared (RMS) error e_p of the position $\mathbf{p}_a$ with respect to desired $\mathbf{p}_{a_d}$ and feasible $\mathbf{p}_{a_f}$ set-points are presented in Figure 8-3.

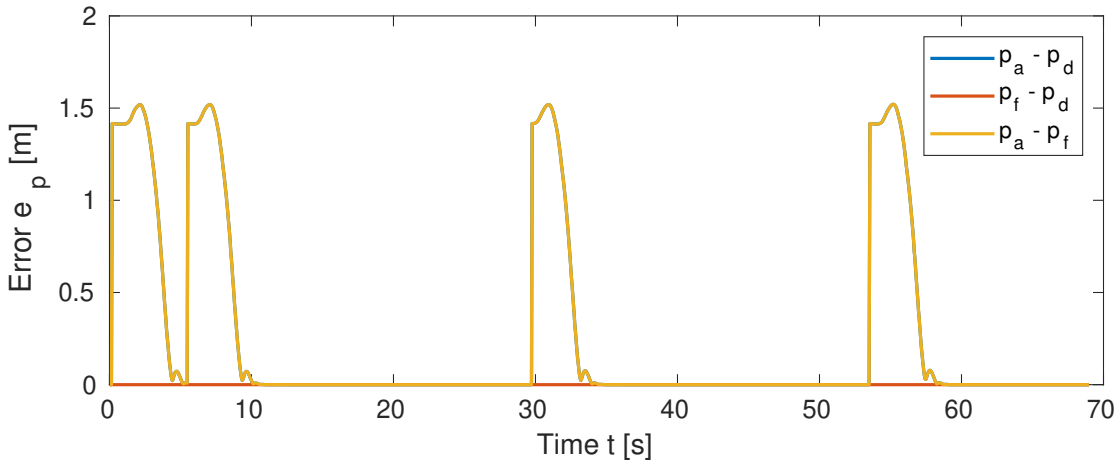


Figure 8-3: RMS position error e_p for single agent

When a switch occurs, a new desired $\mathbf{p}_{a_d}$ and feasible position $\mathbf{p}_{a_{feas}}$ are determined, creating an increase in the error $e_p = 1.42$m. As the agent is discouraged to manoeuvre over the targets, it has to fly around which causes a slight increase to $e_p = 1.52$. When the agent arrives at the new set-point $\mathbf{p}_{a_{feas}}$, a slight overshoot occurs due to the first-order agent dynamics. The time required is execute the manoeuvre is $t \approx 4.5$s. The trajectory generated by the global planner for the third transition at $t = 29.7$s is presented in Figure 8-4. As the trajectory does not change height, a top-down view is shown. Figure 8-5 depicts two instances of the agent at the start $A1$ and during the transition $A1'$. The black global trajectory for the initial step is significantly longer than the generated MPC prediction horizon as the drone has to start moving from a stationary position and has to maintain the cinematographic cost terms. Between the two instances the orientation of the second target has shifted from a Face to Face to Right Angle base configuration. The second instance $A1'$ depict the MPC trajectory exceeding the specified global trajectory to adhere to the cinematographic terms. The focus has shifted from only the second target on the incorrect action side to both targets after crossing the action line $\mathbf{a}$.

To illustrate the effect of the action side shift on the image size σ_I of $\sigma_d = 0.95$m, Figure 8-6 depicts the respective target sizes on the image plane. As can be seen, the image size of the second target σ_I^2 has minimal deviation during a manoeuvre. A slight

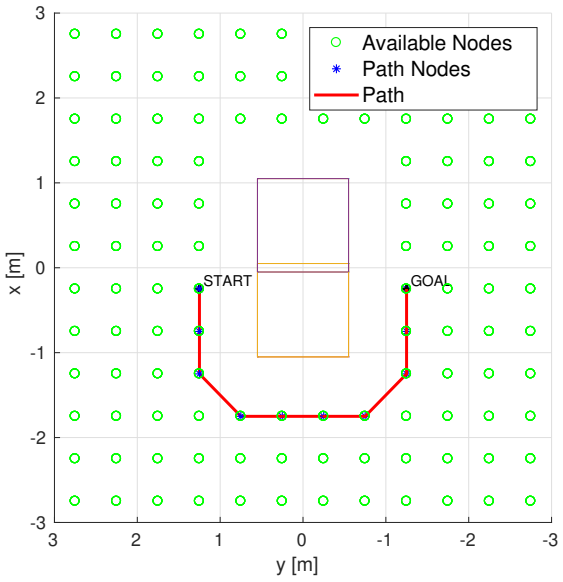


Figure 8-4: Global Trajectory for single agent at $z = 1.25\text{m}$

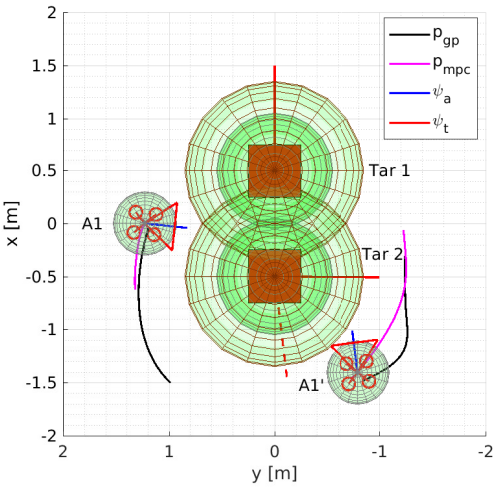


Figure 8-5: Agent transition to set-point $\mathbf{p}_{a_f}$ on opposite side of the targets

decrease is experienced when on the incorrect action side due to motion initialisation and a horizontal shift of the target to the new image position $\mu^2_{x_d} = 214\text{px}$. An increase is visible when both targets are used for cinematographic cost term computations in the MPC on the correct side. The image size of the first target σ_I^1 experiences a decrease as the agent performs the transition on the side of the second target. When the image sizes of both targets cross for the first time post deviation, the agent has reached the position set-point $\mathbf{p}_{a_d}$. However, as stated for the position error e_p, overshoot occurs due to the agent dynamics that converges to a steady state. A steady-state error of 15px occurs between the σ_{I_d} and the actual σ_I for both targets.

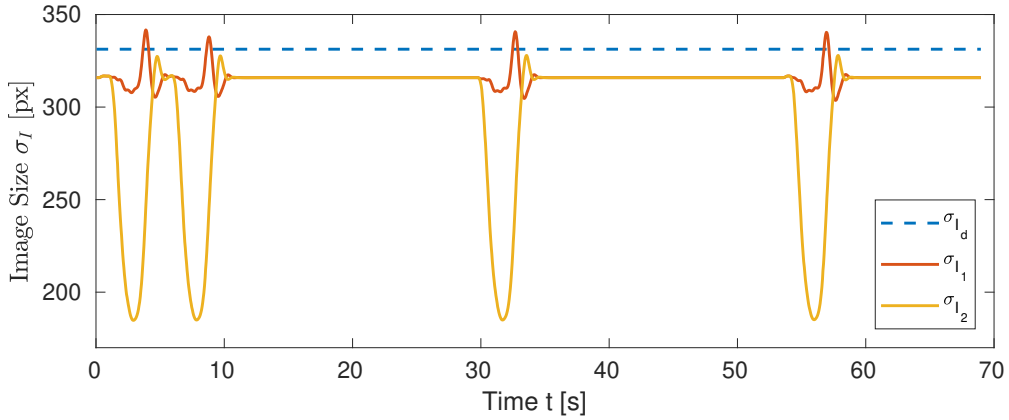


Figure 8-6: Image size σ_I for the target orientation cycle

Two Agents

The configurations are taken into account when determining set-points for two agents as specified in $\mathbf{u}_{cdc}$. The current configuration over time in which the targets are throughout the cycle of nine states is presented in Figure 8-7, referring to Section 4-3 for a depiction of the states. For most transitions, the next state is determined and maintained. However, switching from the fourth Right Angle configuration to the seventh Right Angle configuration at $t = 38$s, a swift transition through the fifth and sixth Right Angle configuration are determined. This causes a succession of desired agent positions $\mathbf{p}_{a_d}$ that are unachievable in the time spent in that configuration. Changing from the seventh Right Angle configuration to the first Face to Face configuration experiences a transition through the first and second Parallel configurations, followed by the first and second Right Angle configuration before settling in the first Face to Face configuration.

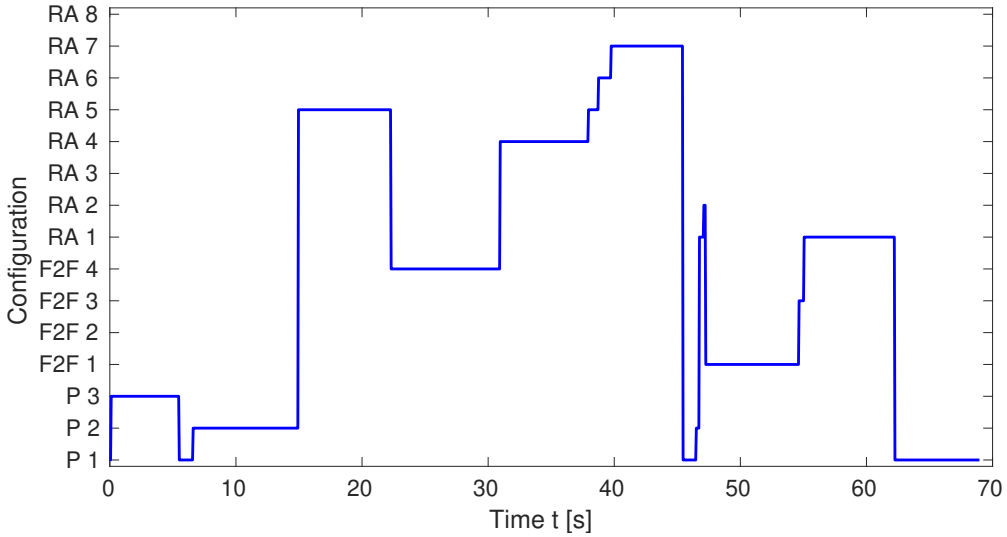


Figure 8-7: The base configuration (Parallel P, Right Angle RA, Face to Face F) combined with the configuration for two static targets

The influence on an unspecified target status in $\mathbf{u}_{cdc}$ is, where Equal is inferred, is presented in Figure 8-8. As can be seen, the inferred target status is adhered to for four of the configurations and overruled for the remaining to provide sensible camera placements $\mathbf{p}_{a_d}$. The Explicit camera placement is adhered to for both agents. At any given time, either one or both agent will frame both targets on the image plane, but neither targets will have a dominant target status simultaneously.

To demonstrate the positioning of agents in the environment, a triad of configurations is presented in Figure 8-9. The first Parallel configuration on the left, whereby a first target has a Dominant view occluding the second target and a Submissive view of both targets. The fifth Right Angle configuration is presented in the middle, whereby the target status is inferred to Equal. The fourth Face to Face configuration is presented on

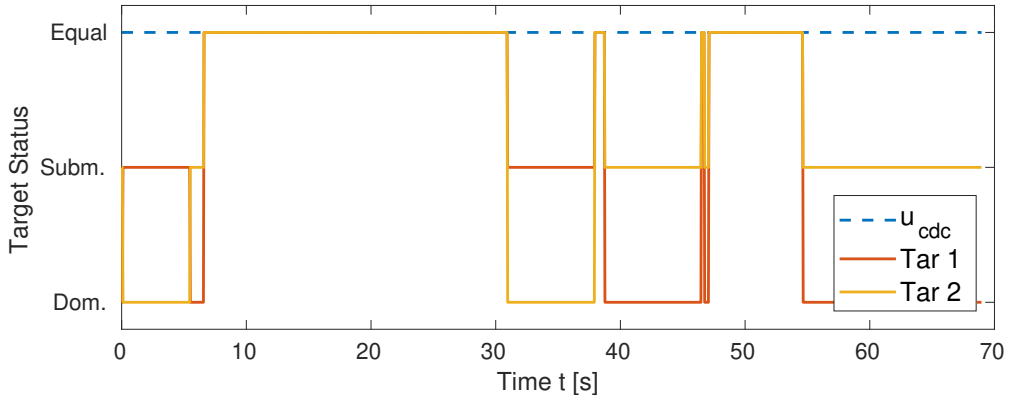
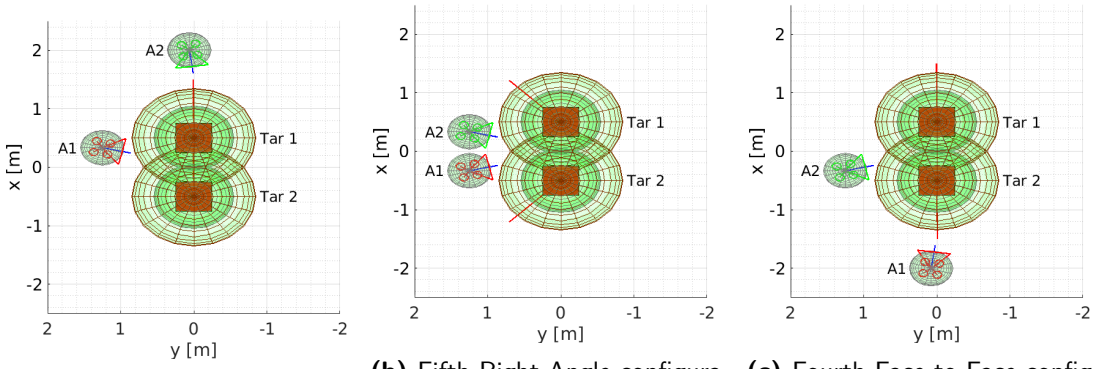


Figure 8-8: Target status for the targets based on the determined configuration

the right, whereby the target status for the second target is assigned a Dominant status in $\mathbf{u}_{cdc}$. This infers that the first target has to be Submissive.



(a) First Parallel configuration **(b)** Fifth Right Angle configuration **(c)** Fourth Face to Face configuration

Figure 8-9: Configuration examples for positioning of two agents

The RMS position errors e_p for the second agent through the configuration cycle are presented in Figure 8-10. The first three deviations are caused by the alternating action side. Around $t = 37.9$s the succession of the Right Angle configurations occurs, resulting in a jagged increase in the error. A convergence to an error $e_{p_{af}} = 0.05$m with respect to the feasible set-point $\mathbf{p}_{a_f}$ is observed due visibility exclusion of the first agent on the image plane $\mathbb{I}$. A steady-state deviation $e_{p_{af}} = 0.15$m for the first Face to Face configuration occurs as the agents are within the potential field of the targets to achieve the over-the-shoulder view. For the penultimate configuration, the first Right Angle configuration, a deviation of the feasible set-point $\mathbf{p}_{a_f}$ with respect to the desired set-point $\mathbf{p}_{a_d}$ to ensure minimal spacing between the agents.

The horizontal image position μ_x of the first target as observed form the second agent is presented in Figure 8-11 for the configuration cycle. Overall the image position μ_x

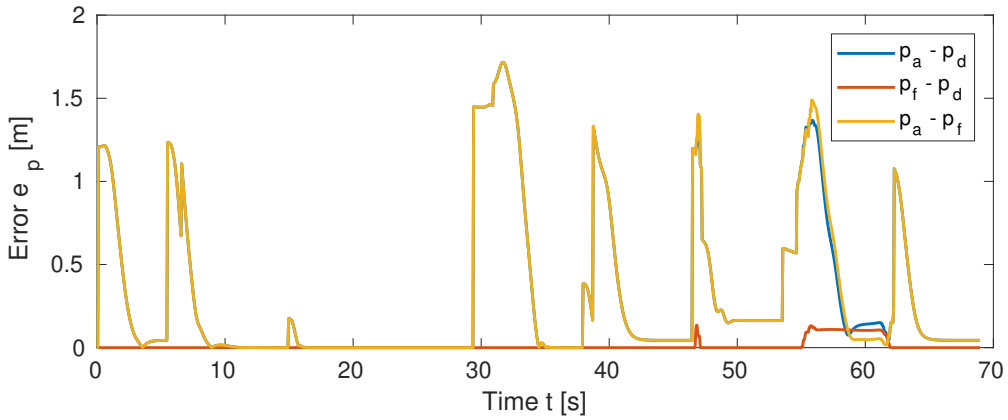


Figure 8-10: RMS errors e_p for the second agent's position $\mathbf{p}_a$, desired set-point $\mathbf{p}_d$ and feasible set-point $\mathbf{p}_f$, respectively

converges to μ_{x_d} with overshoot present due to the agent dynamics when the new set-point $\mathbf{p}_{a_d}$ is reached. The largest deviations occur transitioning to and from the first Face to Face configuration. The agent attempts to achieve an over-the-shoulder view of the second target with respect to the first target. During the transitions, the image position μ_x shift briefly outside the range of view of the agents camera. This is acceptable, as the second target is the main focal point for this configuration.

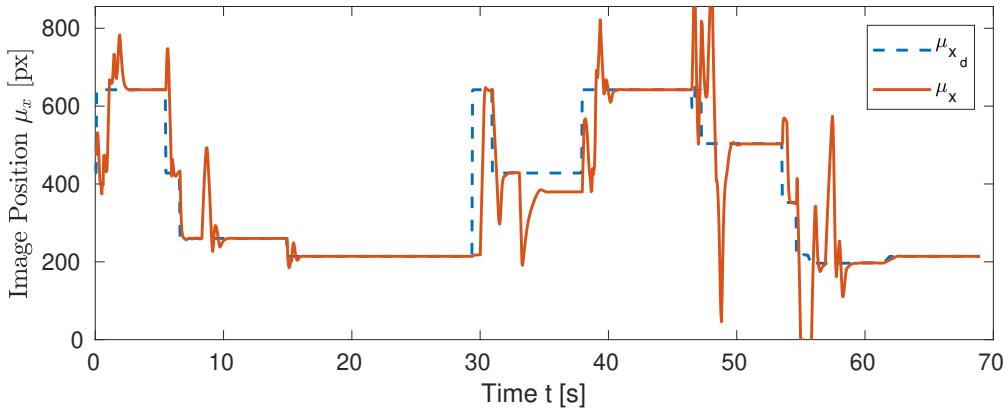
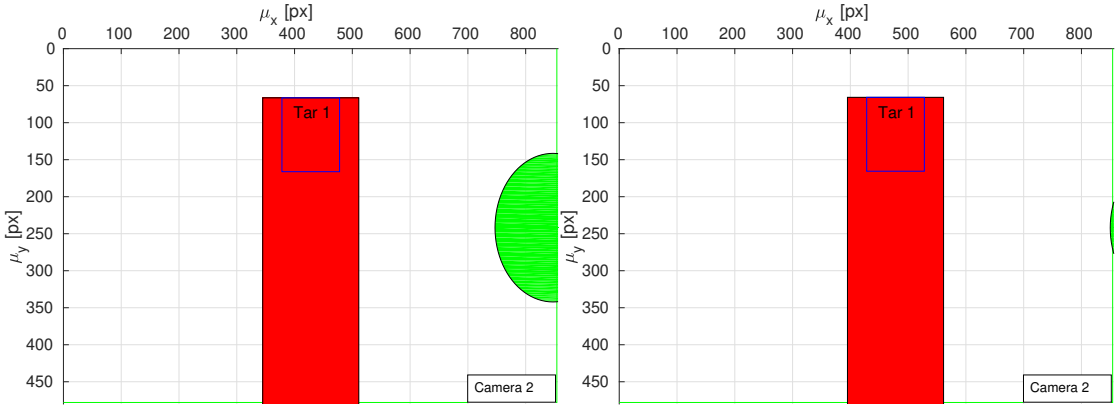


Figure 8-11: Image position of the first target from the second agent

Mutual Visibility

The first Parallel configuration in Figure 8-9a is used to demonstrate the effect of the mutual visibility constraint in the MPC. Figure 8-12a present the camera view of the second agent, whereby the targets are presented by a bounding box in red with a blue inner boarder outlining the head. The agent is positioned with the Dominant optimization, whereby the second target is occluded by the first target. When the mutual visibility

constraint is not taken into account, the first agent can clearly be seen on the right side of the image plane $\mathbb{I}$. However, activating the constraint minimizes the visibility of the first agent on the image plane by altering the yaw angle ψ_a slightly. This shifts the target from the centre of the image plane. A balance is found whereby the target is shifted with respect to the desired position, while the agent is occluded from view.

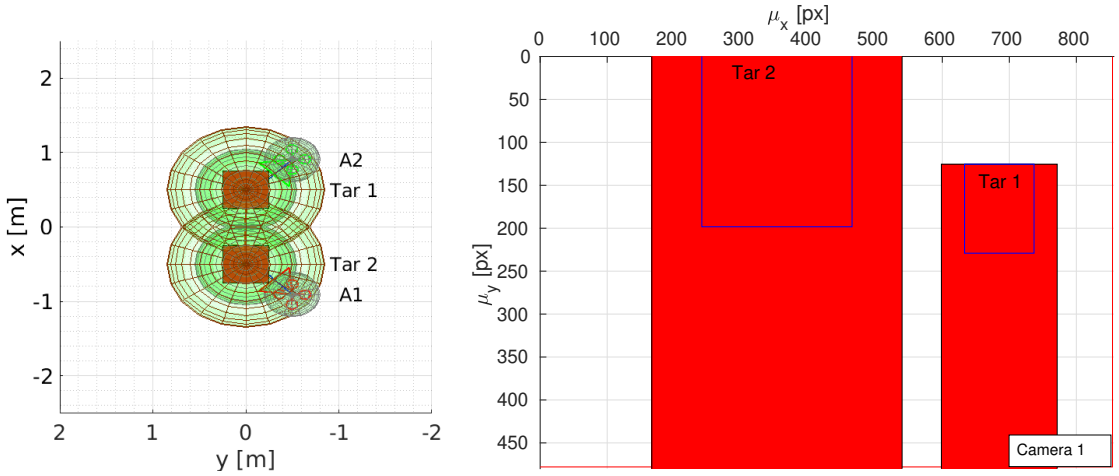


(a) Dominant placement with agent visibility **(b)** Dominant placement omitting agent visibility

Figure 8-12: Camera view of the second agent in the first Parallel configuration

Over-The-Shoulder

A closer look is taken at the first Face to Face positioning, i.e. the over-the-shoulder view of a target. As can be seen in Figure 8-13a, the agents are positioned within the potential field of the targets which created the view in Figure 8-13b. A desired overlap $\eta = 0.1$ is set by the operator in $\mathbf{u}_{cdc}$, which is not achieved due to the collision avoidance constraints that result in a deviation from the desired set-point $\mathbf{p}_{a_d}$. Although Figure 8-13a given the impression that the inner collision-avoidance constraints is violated, this is not the case as can be seen in Figure 8-14. The radius of the target ellipsoid in the direction of the agent $\mathbf{p}_a$ is augmented with the agent radius r_a to acquire the minimum required separation distance. As can be seen, the inner collision avoidance constraint is adhered to within 0.05m. From a theoretical standpoint, this configuration can be achieved. However, considering the practical implementation, this configuration will likely be perceived as unsafe and have to be revised.



(a) Face to Face Configuration **(b)** Image view of the first agent

Figure 8-13: An over-the-shoulder view of the first target

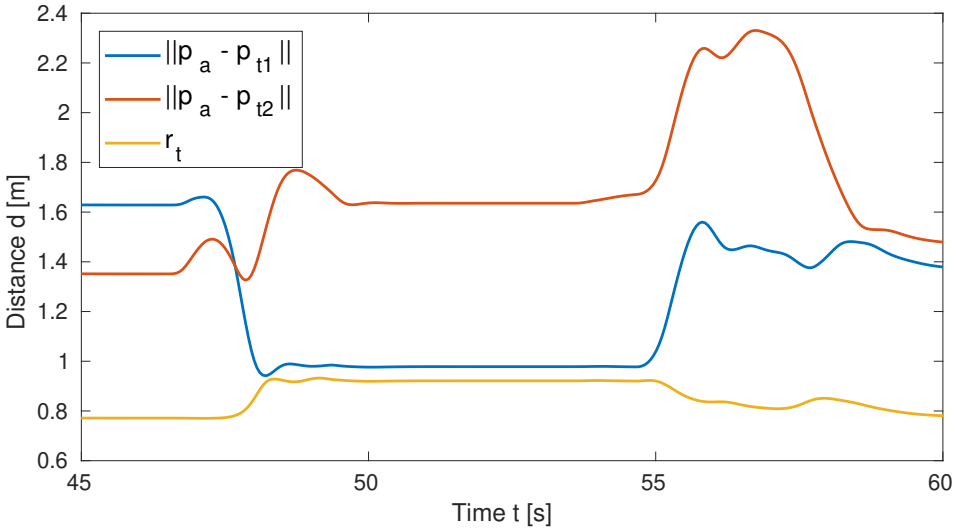
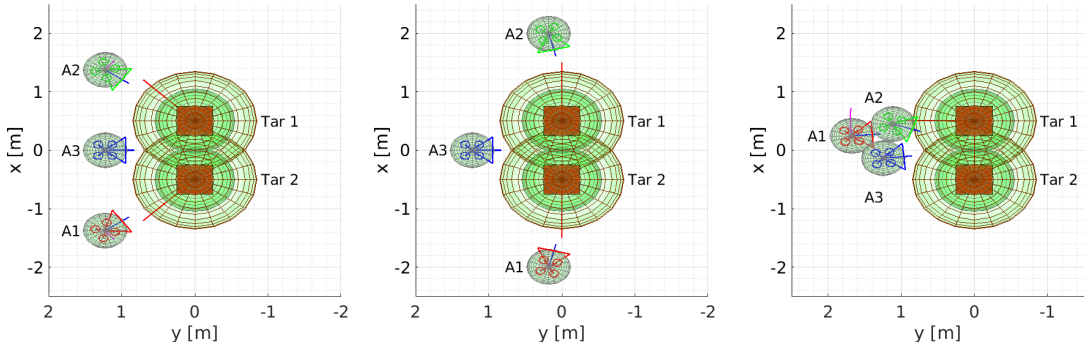


Figure 8-14: Relative distance to the target positions

Three Agents

The addition of a third agent provides additional flexibility regarding the placement of the agent $\mathbf{p}_a$ and target status specification by the operator. The status for both targets is specified as Dominant in $\mathbf{u}_{cdc}$ with $\zeta_u = 0$ rad for the dominant user optimization. This can be specified as one of the agents will always be assigned the master shot. A drawback is that there are more agents manoeuvring through the environment and there is a higher probability of mutual visibility. Three configurations are presented in Figure 8-15, depicting two Right Angle and one Face to Face configuration. As can be seen, the first two examples show the agents are oriented to focus on a single target along with the master shot. However, Figure 8-15c shows conflicting camera objectives as there is an overlap of determined agent positions $\mathbf{p}_{a_d}$ due to the Dominant status provided operator input $\mathbf{u}_{cdc}$.



(a) Fifth Right Angle Configuration

(b) Fourth Face to Face Configuration

(c) First Right Angle Configuration

Figure 8-15: Three agent configurations for two target tracking

The RMS error e_p for the first agent is presented in Figure 8-16, where the influence of the feasibility optimization can be observed. Due to the determination of $\mathbf{p}_{c_d}$ that causes intersections between the agent spheres $\mathbf{S}_a$, feasible set-points $\mathbf{p}_{c_f}$ are determined to prevent such intersections. The three configurations that experience alterations are the third Parallel, first Face to Face and first Right Angle for this agent. This is mainly caused by the proximity of the master shot and a Submissive target status that overrides $\mathbf{u}_{cdc}$.

For the first section of the cycle, the second target is assigned to the Dominant view of the first target, see Figure 8-17. When a switch in action side occurs, the position μ_x experiences a deviation as the agent has to get into motion. The overshoot of the agent when the set-point is reached causes a slight shift on the image position. Due to the prevention of mutual visibility, the image position shifts by $\mu_x \approx 65$px at $t = 24$s. On the left side of the action line, the target adheres to a submissive placement on the right side of the image plane $\mathbb{I}$. During the fast shift in configuration, the image position is set to zero, meaning the target should not be taken into account during MPC computations.

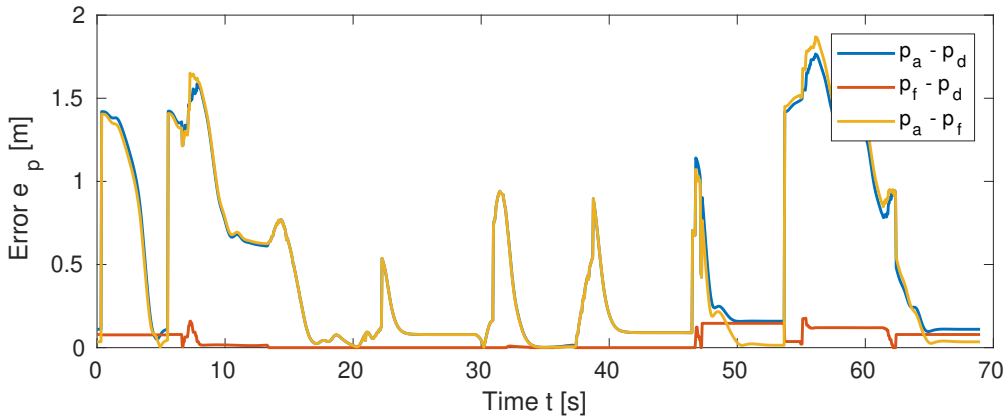


Figure 8-16: RMS errors e_p for the first agent's position $\mathbf{p}_a$, desired set-point $\mathbf{p}_d$ and feasible set-point $\mathbf{p}_f$, respectively

During the final action side shift, the target briefly leaves the image plane on the right side due to a combination of agent dynamics and mutual visibility. The targets ends on the left side of the image plane, whereby the agent has a Submissive position.

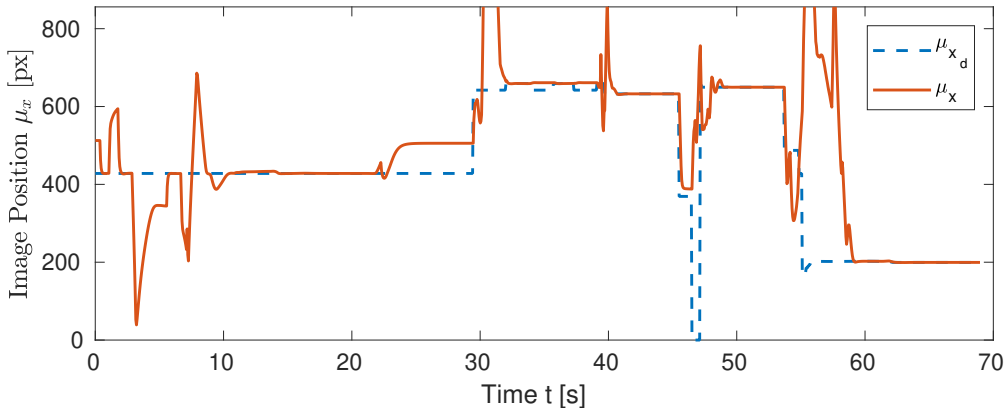
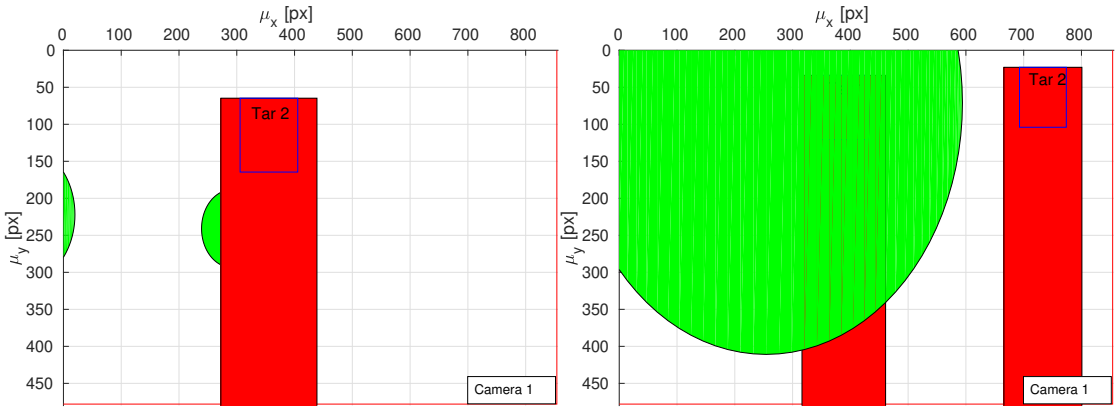


Figure 8-17: Horizontal image position of the first target on the image plane of the second agent

Having a look at mutual visibility of agents for Figures 8-15b and 8-15c, the camera view of the first agent in both configurations is presented in Figure 8-18. For the Face to Face configuration, only the visibility of the third agent is considered along with the positioning of the target. Though visibility of the second agent is not actively taken into account, the second target occludes the first target and partially the agent. The Right Angle configuration is an example where due to the specifications of a Dominant target status along the master shot, the desired positions $\mathbf{p}_{c_d}$ clash with the mutual visibility cost. The second agent is in full view on the image plane $\mathbf{I}$ of the first agent, obscuring the main focal target for the agent which results in a failure case.



(a) Fourth **(b)**

Figure 8-18: Three agent configurations for two target tracking

Computation Time

The computation times for the single, double and triple agent situations can be found in Figure 8-19 for the duration of the configuration cycle. The set-point time t_{sp} pertains to the computation of the desired and feasible set-points for all agents in the simulation. Considering the maxima observer, omitting the outliers, the set-point computations can be run at 500Hz and 24Hz for single and double agent situation, respectively. This significant difference is due the the use of optimizations for the desired and feasible set-points for the double agent situation, where for the single agent a geometric computations is applied. For the three agent situation, the computations require up to $t_{sp} = 0.15$s, with the median at $t_{sp} = 0.09$s due to the increased usage of the feasibility optimization. This implies that the set-point for the MPC, run at 20Hz, would be updated after every two or three iterations. The global planner is run at 2Hz, as it is used to provide a course trajectory through the environment for the agents to adhere to. The time t_{A^*} presented is the total time required to sequentially plan the global trajectories for all agents. The trajectory determination around the targets for a single agent results in 75% of the computations to be within $t_{A^*} < 0.05$s. The two agent situations shows an increase in computations time, having $t_{A^*} < 0.09$s for 75% of the computations. Although generally having shorter distances to traverse when changing configurations, a multiplicity of 2 is observed due to the sequential implementation. With the addition of a third agent, the computation for the global planner increases linearly. The MPC times t_{mpc} presented are also also the summation of the sequential input $\mathbf{u}_a$ computations for each agent. The maxima for each situation, not considering outliers, are $t_{mpc} = 0.04$s, $t_{mpc} = 0.08$s and $t_{mpc} = 0.12$s. For the sequential implementation, the two and three agent situations are outside of real-time operation at 20Hz. Considering the number of agents in the situations, a parallel implementation of both the global planner and the MPC would allow for real-time usage.

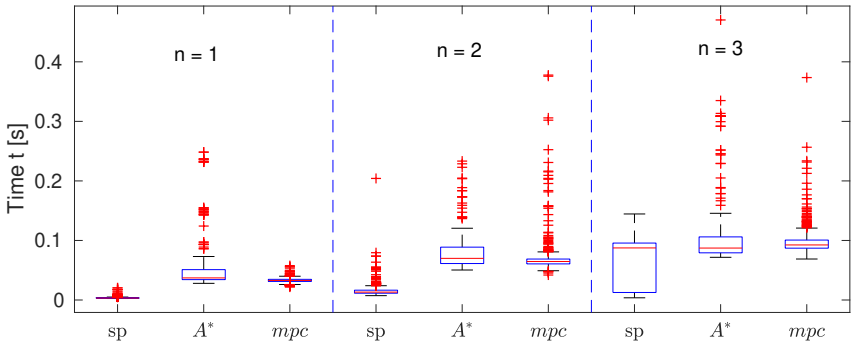
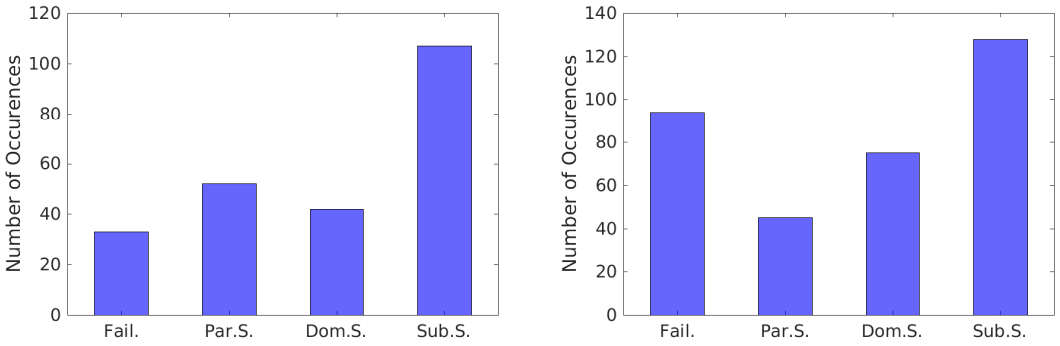


Figure 8-19: Computation times for $n \in \{1, 2, 3\}$ agents for the configuration cycle

Performance

The performance of the camera placements in the quasi-static environment is evaluated for the two and three agent situation. White Gaussian noise is applied to $\mathbf{p}_{t_{xy}}$ to imitate a person shifting position for each configuration. The deviation magnitude is limited to 0.25m, being a large shift in posture. The simulation is run fifteen times for both situations with agent initialisation altering throughout the environment. Five criteria are used to evaluate the performance of individual agents. The first is the achievement of the feasible position $\mathbf{p}_{a_f}$ within 0.3m. The next criteria is achieving the image position $\mathbf{m}_d$ within 75px for the considered targets. The third criteria is achieving the image size σ_{I_d} within 10%. Being on the correct side of the action line is also considered. Finally, undesirable visibility of other agents is taken into account. Successful placement of an agent is achieved if the agents maintain the constraints for longer than $t > 1.5$s, accounting for the a transition every seven seconds and the transition time. Partial success is achieved when either only one of two targets is being tracked prior to the configuration change or the incorrect action side is being maintained while tracking. Partial configurations that occur during the configuration shift are not taken into account as these are unachievable within the time spent in the configuration.

The results for this evaluation is presented in Figure 8-20. Considering two agents, a failure rate of 14% is observed. The main causes are the positioning on the incorrect action side and not maintaining the set-points for the desired time. Partial success of 22% accounts for the Dominant target views within range of position and image constraints, but is on the incorrect action side. Agents during their transition and having to track two targets are also considered here. A success rate of 64% is observed for the tracking of one and two targets, collectively. For the three agent situation, the failure rate increased to 27% due to inter-agent hindrance in achieving the set-points and mutual visibility. The collective success rate decreases to 59%, seeing an increase in dominant success. This is a result of specifying a Dominant target status in $\mathbf{u}_{cdc}$ for both targets in a subset of runs.



(a) Two agent situation **(b)** Three agent situation

Figure 8-20: Performance evaluation for agent placement for Dominant and Submissive placement

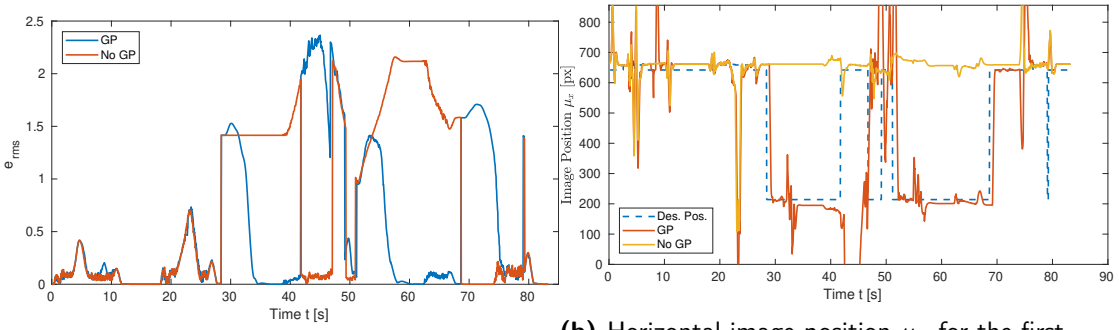
8-3 Simulation: Dynamic Motion

The second simulation is set up to evaluate the performance of the system when targets are in motion through an environment with obstacles present. The targets start stationary and side-by-side inn the second Parallel configurations. A motion occurs along the target orientations during which a dynamic collision avoidance occurs due to an obstacle moving in the opposite direction between the two targets, see Figure 8-22. Next, the first Parallel configuration is adopted by the targets. A following motion is simulated whereby a static collision avoidance occurs as seen in Figure 8-25. Thirdly, a rapid succession of target configurations is performed while the targets are in motion, which can be found in Figure 8-29. Finally, a motion exceeding the dynamic constraints of the agents is executed which ends in the initial position of the targets. A complete situation impression of the trajectories for the target orientations ψ_t and positions $\mathbf{p}_t$ is presented in Appendix B-4. First, the influence of the global planner in a dynamic environment is evaluated.

Global Planner

The influence of the global planner, the A* algorithm, is evaluated with respect to a single agent maintaining a master shot of the two dynamic obstacles. The RMS position error $e_{p_{af}}$ of the agent position $\mathbf{p}_a$ with respect to the feasible set-points $\mathbf{p}_{a_f}$ can be found in Figure 8-21a for the whole dynamic motion through the environment. The horizontal image position μ_x of the first target is presented in Figure 8-21b. During the two straight line motions along the boundaries of the the environment, a steady-state error $e_{p_{af}} = 0.1$m is observed due to the dynamic motion. The static collision avoidance at $t = 23.1$s causes the image position the transition outside of the image boundaries. When a shift in the action side occurs at $t = 28.4$s, the global planner guides the agent

to the feasible position $\mathbf{p}_{a_f}$. The agent absent the global planner maintains the current action side with a steady-state error of $e_{p_{af}} = 1.41$m. This results in the incorrect image positioning of the targets, whereby the prominent target head orientations are not captured. Better continuous framing is achieved without the influence of the global planner during the quick succession of configurations starting at $t = 38.9$s, albeit with the target on the incorrect side of the image. The cause for this is the shifting of the action side, whereby the agent with the global planner is predominantly executing transitions over the action line to adhere to the target configurations.



(a) RMS position error $e_{p_{af}}$

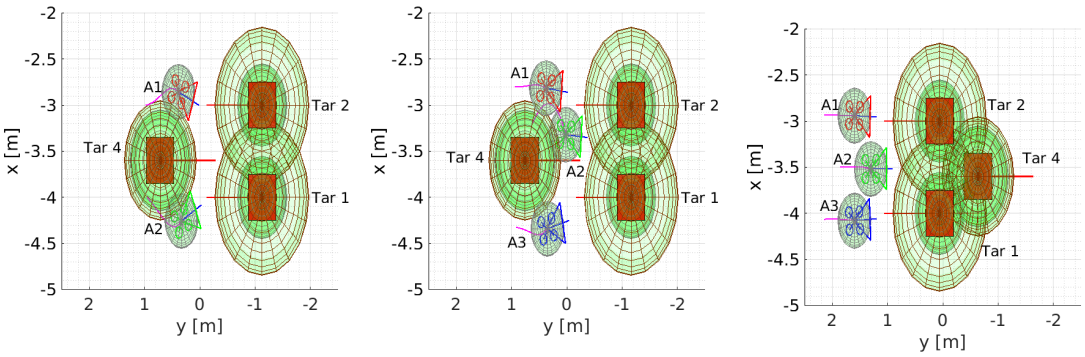
(b) Horizontal image position μ_x for the first target

Figure 8-21: Influence of the global planner on the position $\mathbf{p}_a$ and horizontal image position μ_x for a single agent

Dynamic Collision Avoidance

The dynamic collision avoidance occurs during the motion parallel to the environment boundary x_{min}. The targets are in the second Parallel configurations, representing two people moving side-by-side in a forward motion. The target status is omitted in $\mathbf{u}_{cdc}$, resulting in an Equal label for both targets. Illustrated in Figure 8-22 are the collision avoidance for two and three agents for an obstacle Ω_o^4 heading in the direction counter to the focal targets, respectively. The agents have to deviate from their desired positioning $\mathbf{p}_{a_d}$ to mitigate collision with Ω_o^4. Although vertical movement p_{a_z} and velocity v_{a_z} are more severely weighted in the MPC, the agents descent slightly at first and then move around the obstacle. Considering the three agents in Figure 8-22b, two agents deviate to one side of the targets as the second agent is in the direct path of the obstacle. This result in a further deviation for the second agent to prevent inter-agent collision.

Taking a closer look at agent A1 in the Figure 8-22a and agents (A1, A2) in Figure 8-22b, the RMS error e_p of their positions $\mathbf{p}_a$ with respect to the feasible position $\mathbf{p}_{a_f}$ are presented in Figure 8-23. As can be seen, the first agents A1 have an equivalent deviation due to the impending collision. To account for both the collision with the obstacle Ω_o^4 and agent S_a^1, the second agent A2 performs a lateral motion and a descent. This results in a peak deviation $e_{rms} = 0.39$m at $t = 7.5$s.



(a) Two agent collision avoidance

(b) Three agent collision avoidance

(c) Three agents post avoidance

Figure 8-22: Collision avoidance of a dynamic obstacle for two and three agent situations

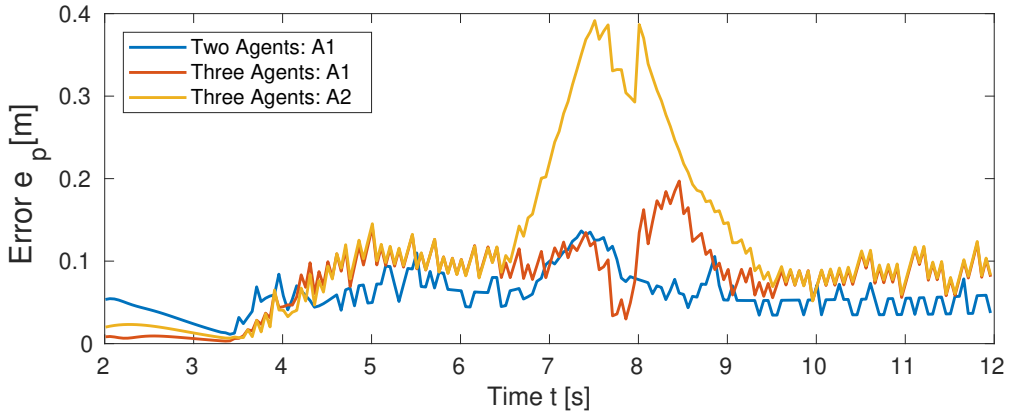
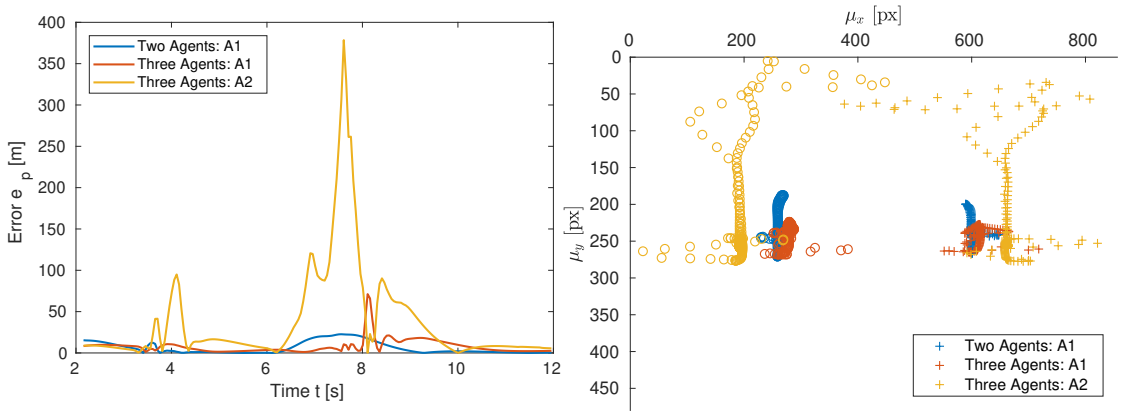


Figure 8-23: Position error e_p for the two and three agent dynamic collision avoidance

Evaluating the RMS error e_m for the image position of the projected point $\mathbf{p}_{ip}^2$ of the second target in Figure 8-24a, the deviation of the first agent in both situations is below $e_m < 91$px. This is classes as an acceptable deviation from the desired set-points. The second agent A2, a master shot, experiences a more significant significant error of $e_m = 380$px at $t = 7.65$s due to the additional vertical motion. The initial peak at $t = 4$s is due to motion initialization of the targets. The image positions $\mathbf{m}$ of both targets as viewed by all considered agents for the collision avoidance are presented in Figure 8-24b. As a Submissive agent placement is stated, both instances of A1 maintain $\mathbf{m}_d^1$ within reason. However, due to the vertical descent of the second agent A2, a transition to the top of the image plane $\mathbb{I}$ occurs. The results in the head of the targets being outside of the viewing range of the camera. Therefore, the master shot provided by the second agent A2 becomes unusable as a result of the collision avoidance.



(a) RMS error e_m of the second target on the image planes

(b) Position of Target 1 + and Target 2 ∘ on the respective image planes

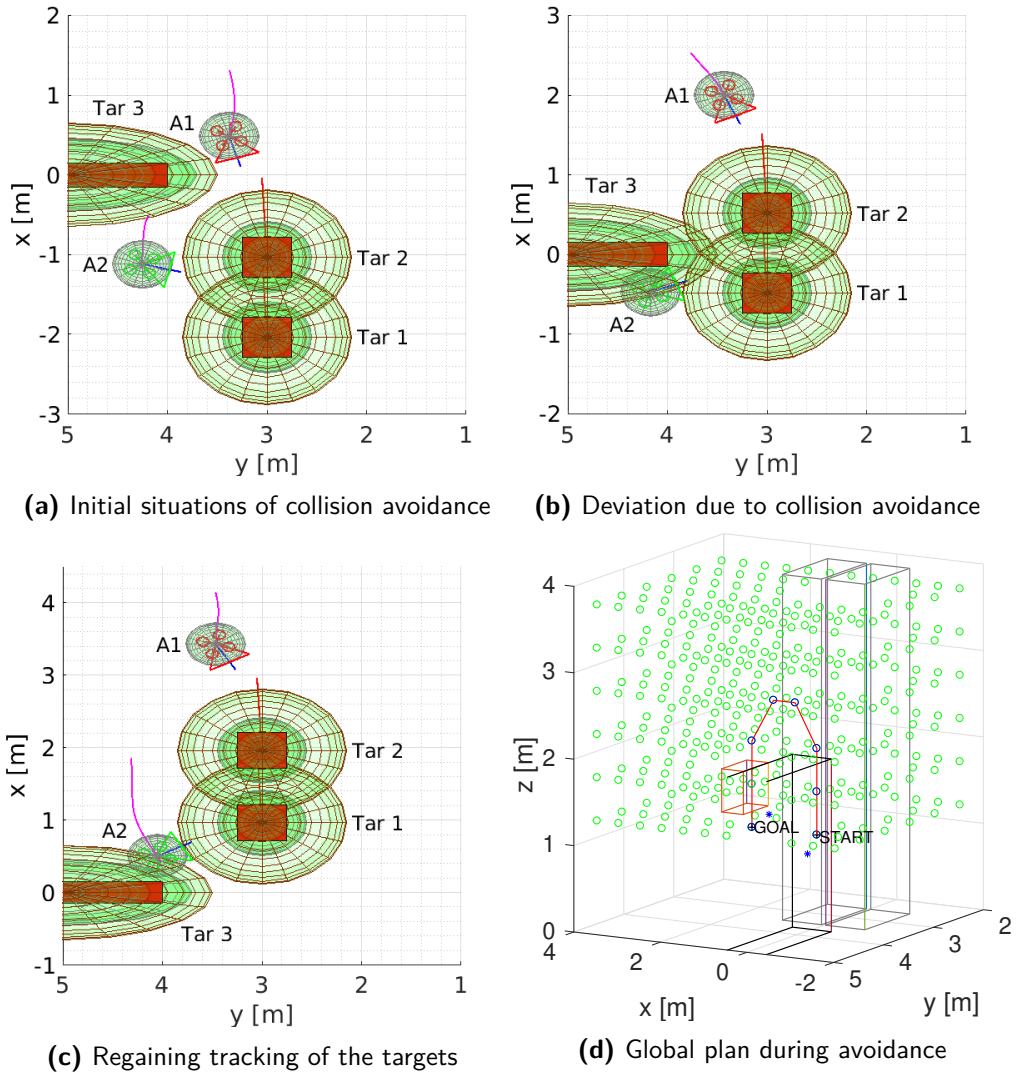
Figure 8-24: The error e_m and image position of the targets for A1 in the two agent and (A1, A2) in the three agent situation

Static Collision Avoidance

The static collision avoidance occurs when the targets manoeuvre parallel to the environment boundary y_{lim} in the first Parallel configuration, during which a static obstacle Ω_o^3 creates an obstruction. Two agents $n = 2$ are specified in $\mathbf{u}_{cdc}$, which results in a Dominant view of the second target and Submissive view of both targets. The manoeuvre is presented in Figure 8-25, consisting of a view prior to, during and after the avoidance. The global trajectory computed during the avoidance is also included.

The Dominant view of the second target achieved by the first agent A1 does not experience a deviation from its trajectory as it transitions past the obstacle Ω_o^3 at an acceptable distance. To account for the mutual visibility of the second target A2, it does experience a deviation from positioning the second target in the centre of the camera view. The second agent A2 experiences a significant deviation as the obstacle Ω_o^3 is directly obstructing its trajectory to achieve the cinematographic requirements. As the targets transition close to the obstacle Ω_o^3, the agent is unable to avoid collision by flying between the obstacles. Therefore, it halts at the boundary of the static obstacle Ω_o^3 until the targets have passed it. For Figure 8-25b, the determined global plan for the second agent can be found in Figure 8-25d. Given the situation, the shortest path to the feasible position $\mathbf{p}_{a_f}$ is an ascension over the target. However, the agent does not adhere to this global trajectory as the targets transition past the obstacle close after initialisation of the motion. This allows the agent to shift around rather than over the static obstacle Ω_o^3.

The second agent A2 provides a Submissive view of the targets, functions as the master shot for the situation. The RMS image position error e_m for both targets is presented in Figure 8-26. The initial influence of the collision avoidance occurs at $t = 23.0$s, where



(a) Initial situations of collision avoidance

(b) Deviation due to collision avoidance

(c) Regaining tracking of the targets

(d) Global plan during avoidance

Figure 8-25: Situations representation of static collision avoidance of two agents

the agent has to reduce its velocity $\mathbf{v}_{xy}$ to mitigate collision. As the target approaches the obstacle, the executed a yaw motion ψ_a to compensate for the reduction in velocity and adhere to the set-points. This results in a position error within $e_m^2 < 110$px for the second target while maintaining tracking of the first target. When the targets are level with the static obstacle Ω_o^3, a significant increase in the error occurs, culmination in the peak error $e_m = 762$px at $t = 26.9$s. During this period the agents is hindered from transitioning around the static obstacle Ω_o^3 by the first target Ω_t^1. The master shot becomes unusable for the static avoidance, as the second target is outside the view of the camera. When sufficient space is available for the agent to manoeuvre around the obstacle, it regains the desired cinematographic set-points. The input sequence $\mathbf{u}_a$ can

be found in Appendix B-4.

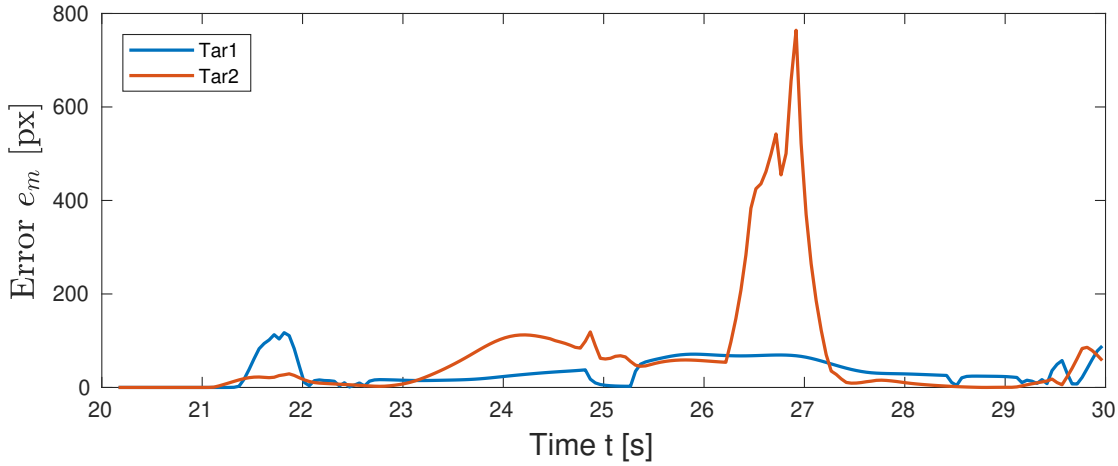


Figure 8-26: Image position RMS error e_m of the targets for the second agent

The relative distances $\|\mathbf{p}_a^i - \mathbf{p}_t\|$ between the agents and the static obstacle Ω_o^3 are presented in Figure 8-27, along the the radius of the ellipsoid as perceived by the agents. The first agent remains $\Delta d = 0.37$m from the safety boundary. The second agent is halted by the static obstacle Ω_o^3 while the targets transition past. When the availability occurs for the second agent to pass the obstacle, a violation of the safety boundary occurs at $t = 26.8$s with a magnitude of $\Delta d = 0.09$m. Although no physical collision occurs as the safety radius is set to $r_t = 0.4$m and the safety boundary is formulated as a soft constraint with a slack variable, the safe operation of the agents is compromised.

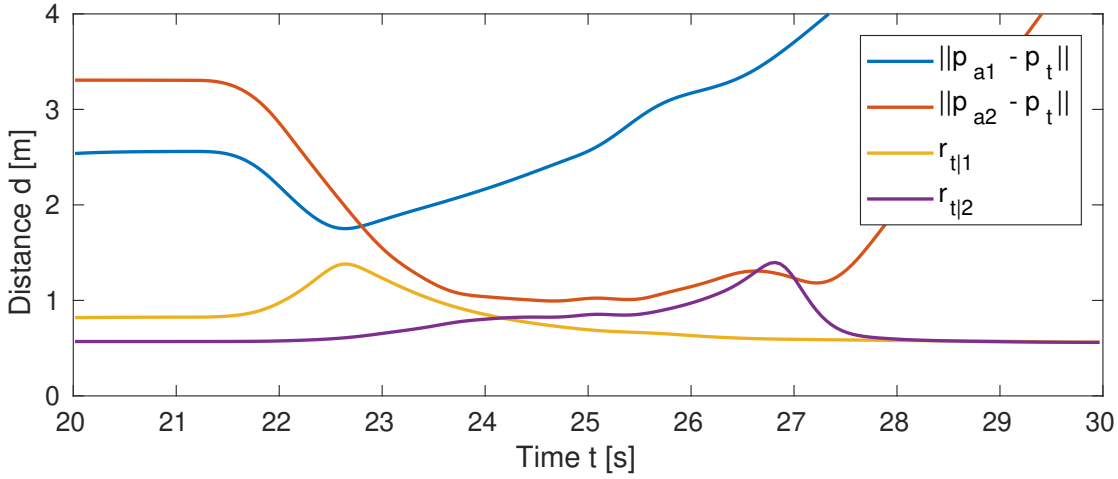


Figure 8-27: Relative distance between the agents and the static obstacle, with the ellipsoid radius r_t

Dynamic Motion

Six configuration changes are performed in the timespan $t \in [36, 62]$s while the targets are also in motion. This will illustrate the limitations of the time required to transition over the action line when a side shift occurs while the targets are in motion. The configuration succession is presented in Figure 8-28. A selection of configuration is presented in Figures 8-29 and 8-30 for the two and three agent specification in $\mathbf{u}_{cdc}$. The yaw ψ_t and position $\mathbf{p}_{t_{xy}}$ can be found in Appendix B-4 for the target pair.

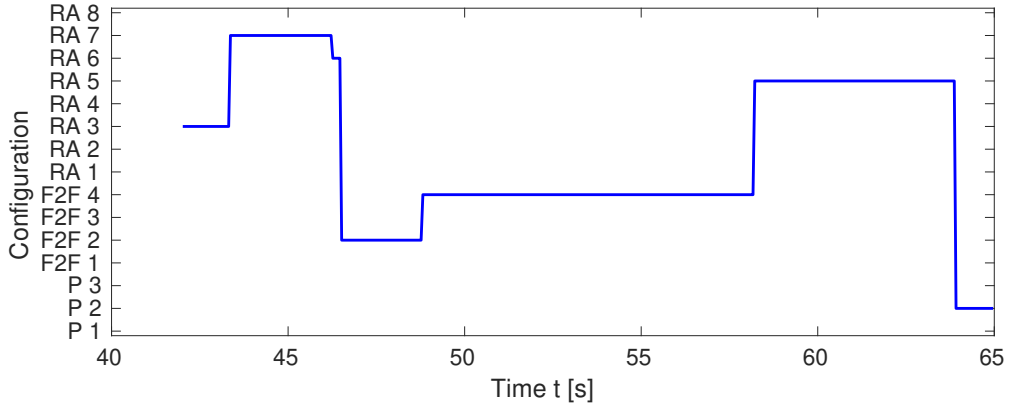
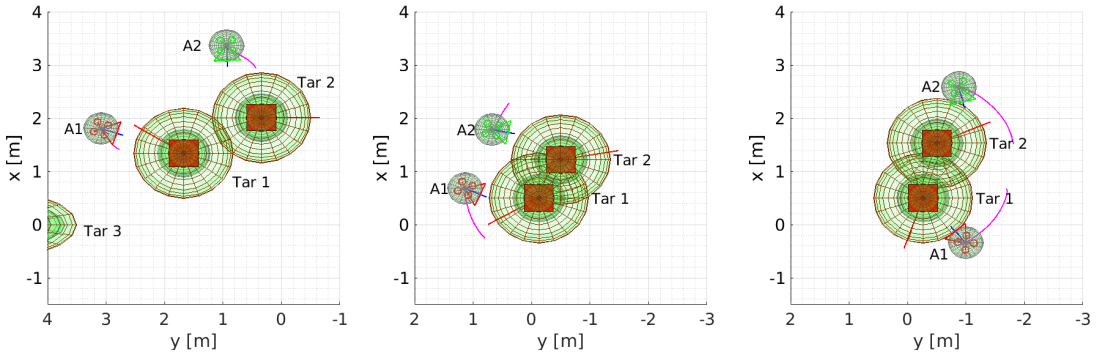


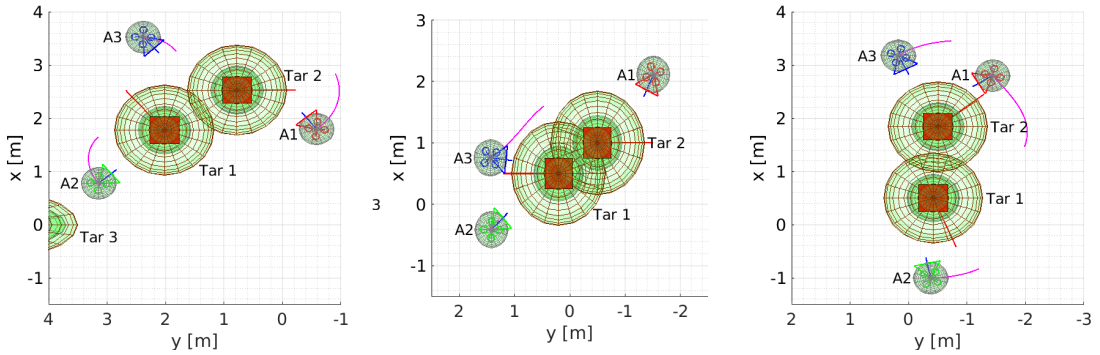
Figure 8-28: Configuration (Parallel P, Right Angle RA, Face to Face F) sequence for dynamic motion



(a) Sixth Right Angle configuration at $t = 43$s **(b)** Second Face to Face configuration at $t = 45$s **(c)** Fifth Right Angle configuration at $t = 58$s

Figure 8-29: Target configuration and agent movement during the dynamic motion for two agents

The agents are in constant motion during the dynamic motion to achieve the set-points as can be seen by the MPC trajectories of the agents in Figures 8-29 and 8-30. The performance of the set-point achievement is evaluated for three cycles of the dynamic motion with the following criteria. A position $\mathbf{p}_{a_f} < 0.3$m has to be achieved and



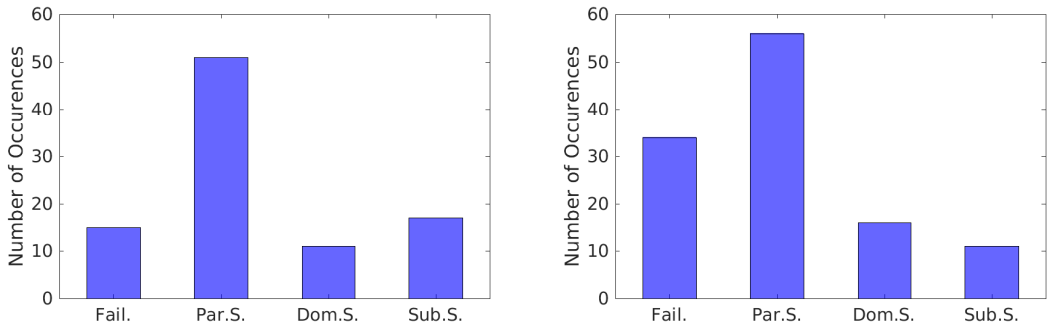
(a) Sixth Right Angle configuration at $t = 43$s **(b)** Second Face to Face configuration at $t = 45$s **(c)** Fifth Right Angle configuration at $t = 59$s

Figure 8-30: Target configuration and agent movement during the dynamic motion for three agents

maintain for $t > 0.5$s. The tracking of either one or both targets has to be achieved within $\mathbf{m}_d < 75$px depending on the target status. The image size σ_I has to be within 10% margin. Finally, the agents have to adhere to the specified action side. Mutual visibility is not taken into account when considering the performance. The performance for the two and three agent situations are presented in Figure 8-31. Considering two agent situation, the agents are predominantly executing a transition across the action line while tracking a single target, having an occurrence of 54%. A failure rate of 16% is observed, whereby for a majority a movement counter to the current direction of motion is required or mutual exclusion prevents correct achievement of the desired cinematographic set-points $\mathbf{m}_d$ and σ_{I_d}. A collective success rate of 30% for the Dominant and Submissive agent placements is observed. Depending on the desires of the operator, approximately half of the captured motion is unusable due to the limitations on the agent dynamics. A failure rate of 30% is observed for the three agent situation. Due to the increase in agents manoeuvring around the targets, the mutual visibility of agent and inter-agent collision avoidance occurs more frequently. One results is incorrect viewing angles where the set-points are not achieved as a transition of an agent through the camera view is required to achieve them. A second is the proximity of $\mathbf{p}_{a_f}$ for multiple agents which causes agents to deviate to mitigate collision.

Computation Time

The computation times for the set-points t_{sp}, the global planner t_{A^*} and the MPC t_{mpc} for $n \in \{1,\ 2,\ 3\}$ agent situations is presented in Figure 8-32. The computational performance of the single and double agent situation is comparable to the first simulation in Figure 8-19. The set-point computation t_{sp} for three agents requires $t_{sp} = 0.15$s for 75% of the computations, the the maximum at $t_{sp} = 0.31$s. This implies that the set-points are updated every 3-6 iterations when the MPC is run at 20Hz. For a dynamic



(a) Two agent performance **(b)** Three agent performance

Figure 8-31: Performance of set-point achievement during dynamic motion

situation as evaluated in the simulation, this causes a delay in the adaptation of the agents to the current situation. This could result in unsafe situations or failure in achieving the set-points.

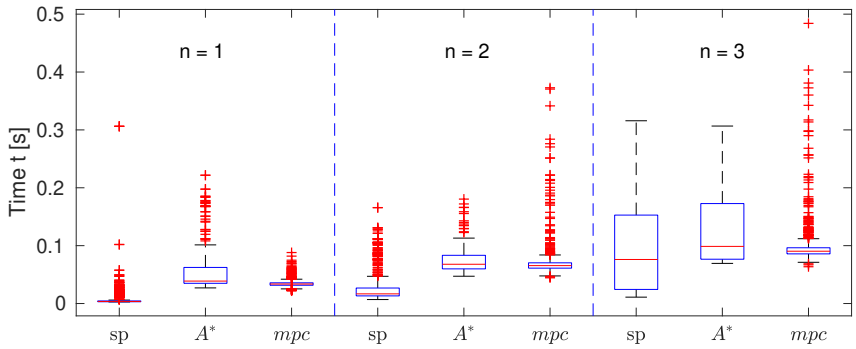
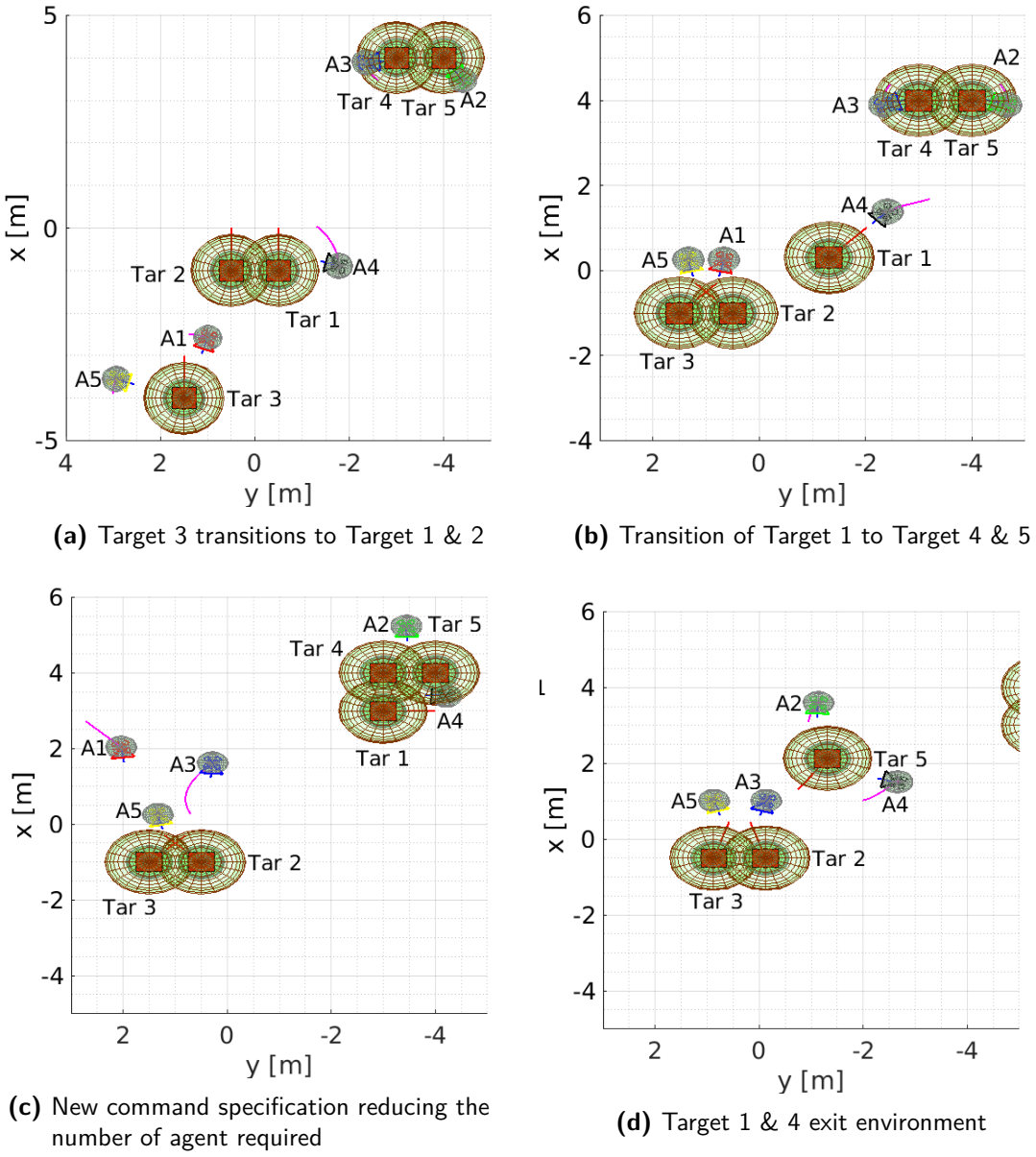


Figure 8-32: Computation times for $n \in \{1, 2, 3\}$ agents for the dynamic simulation

8-4 Simulation: Multiple Commands

This simulation is set up to extend the functionality of the system to make use of more than three agents and collectively focus on more than two targets at a given time. Allowing the operator to specify multiple commands $\mathbf{u}_{cdc}$ provides the opportunity utilise a larger number of agents. Each command would be assigned to a sub-group which is restricted to the use of $n \in \{1,\ 2,\ 3\}$ agents to focus on two targets. Overlapping of the targets specified in $\mathbf{u}_{cdc}$ can create connections between multiple targets. The commands are evaluated sequentially to compute the set-points.

The situation presented in Figure 8-33 will be considered to demonstrate the flexibility, scalability and limitations. Five agents and targets are defined in the environment,



(a) Target 3 transitions to Target 1 & 2

(b) Transition of Target 1 to Target 4 & 5

(c) New command specification reducing the number of agent required

(d) Target 1 & 4 exit environment

Figure 8-33: Overview of the motions occurring for the multiple command simulation

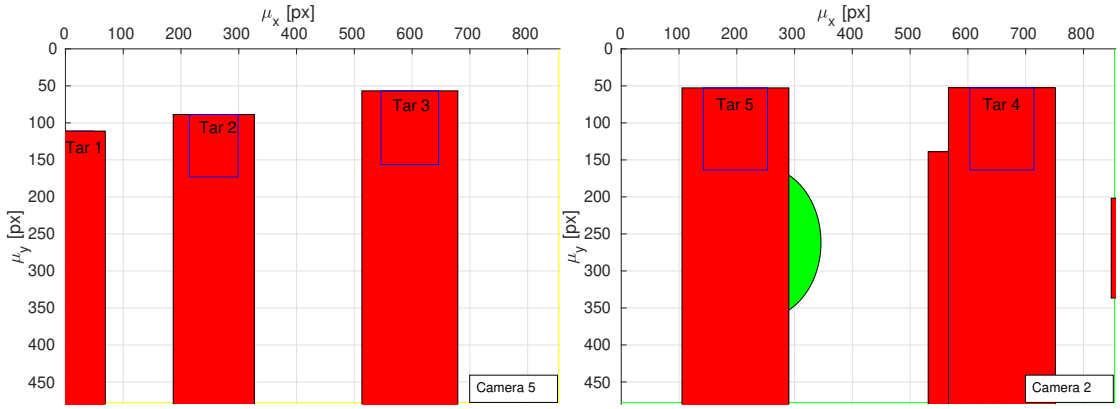
whereby at any given time up to three commands are specified. The commands can be found in Appendix B-5. The situation consist of four major motion elements. The first motion is the third target moving to a position in line with the first and second target. Two agents are assigned to the former and one to the latter. The initialisation of the motion can be seen in Figure 8-33a. The remaining targets are positioned in the right corner of the environment, being assigned two agents to create an over-the-shoulder view.

The second motion is performed by the first target, which manoeuvres to the fourth and fifth target, see Figure 8-33b. It is assigned a single agent to create a Front view. A new command sequence is specified for Figure 8-33b in which the total number of agents to use is reduced to $n = 4$. Two agent remain assigned to each target cluster. Finally, the first and fourth target exit the environment and the fifth target executes a motion towards the second and third target in Figure 8-33d.

Camera Views

The use of multiple command and agents provides additional flexibility in the system when considering the positioning of multiple targets on the image planes . However, with the addition of agents, the probability of mutual visibility also increases. To prevent frequent occurrence of incorrect camera positioning $\mathbf{p}_a$ and orientations ψ_a, mutual visibility is only taken into account for those agents required for a single operator command $\mathbf{u}_{cdc}$.

Consider the culmination of the first motion in Figure 8-33a, where the first three targets are in a line facing the same direction. One agent is assigned to create a master shot of the first two targets and two agents are commanded to frame the second and third target. An overlap of the image planes of the first and fourth agent occurs, whereby the second target is on alternate halves of the plane. The fifth agent creates the view depicted in Figure 8-34a, whereby the second and third target adhere to the set-points, with the addition of the first target implicitly on the edge of the view. A similar implicit occurrence of the first target as seen by the second agent in Figure 8-33c is depicted in Figure 8-34b. It is assigned to create a master shot of the first Face to Face configuration of the fourth and fifth target. The fourth agent, which is partially visible in the view, creates master shot of second Parallel configuration of the first and fourth target.

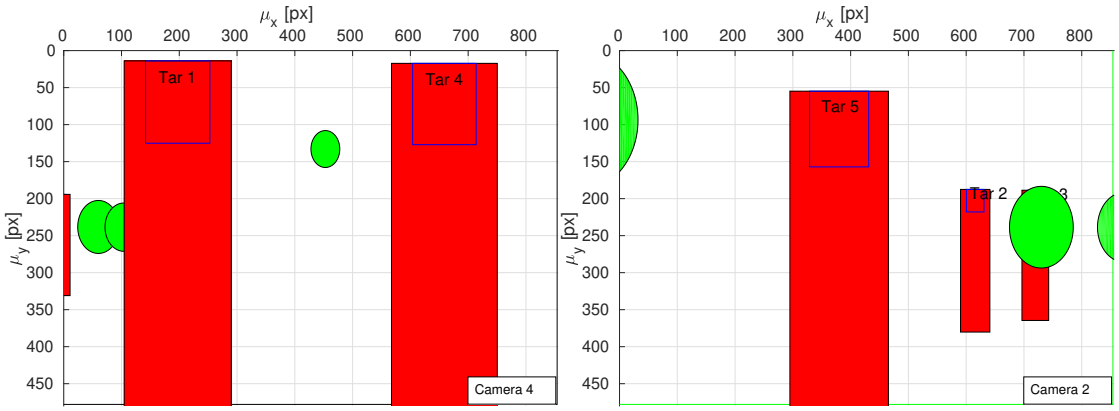


(a) View of the fifth agent of the in-line sequence of targets

(b) Master shot of the second agent for Target 4 & 5

Figure 8-34: Examples of camera views with multiple target implicitly present

Two examples of the mutual visibility of agent assigned an alternative operator command $\mathbf{u}_{cdc}$ are presented in Figure 8-35. The view from the partially visible agent in Figure 8-34b is presented in Figure 8-35a, whereby the agents assigned to the other target cluster are visible on the left side of the image plane $\mathbb{I}$. Figure 8-35b outlines the same agents while being assigned to frame the fifth target as a Front view. Due to collision avoidance during the targets exiting the environment, the agent is unable to achieve the feasible position $\mathbf{p}_{a_f}$ while the fifth target is in motion. The cinematographic aspects of the MPC are adhered to.



(a) Master shot with agent visibility **(b)** Single target view with agent visibility

Figure 8-35: Examples of inter-command agent mutual visibility

Assignment

A closer look will be taken at Figure 8-33c. When target one reaches target four and five, new commands are provided in which the sum of desired agents is less than the number present. One of the agents will remain unassigned to set-points. Therefore, a point $\mathbf{p} = [4\ 4\ 3]^T$ is defined for the unassigned agent to reside and provide minimal hindrance to the scene. This point is augmented to the feasible set-points, creating the distance cost matrix $\mathbf{C}$ in Equation (8-1). The resulting assignment is (5, 3, 2, 4, 1), with the bottleneck element $c^* = 5.35$ and $\sum c_{ij}x_{ij} = 12.64$. The behaviour resulting from this that the third agent changes from target pair and takes the position initially occupied by the first agents. The first agent becomes the unassigned agent.

$$\mathbf{C} = \begin{bmatrix} 0.67 & 0 & 5.87 & 4.86 & 5.26 \\ 7.16 & 6.64 & 0.99 & 1.98 & 8.57 \\ 5.78 & 5.35 & 1.88 & 1.98 & 6.75 \\ 6.48 & 5.91 & 0.27 & 1.05 & 8.53 \\ 0 & 0.67 & 6.43 & 5.44 & 4.86 \end{bmatrix} \tag{8-1}$$

Computation time

The computation times required to process multiple simultaneous command is considered in Figure 8-36 to evaluate the scalability of the system. As the simulations predominantly contains static and slow dynamical movement, a comparison is made with the computational times in Figure 8-19. The maximum computation time for the two agent situation is $t = 0.04$s, whereas the median of the two command t_{2sp} is $t = 0.08$s and a maximum at $t = 0.15$s. This quadruple increase in the maximum is due to the increased use of the feasibility optimisation due to the larger number of agent and targets present in the environment, equivalent to the dynamic three agent situations. The three command computation time t_{3sp} has a median at $t = 0.09$s and a maximum at $t = 0.13$s. The third command is generally a geometrical computation to generate a master shot or track a single target, which results in an equivalent computation time with respect to the two command situation. The maximum is less due to generation of desired set-points that are largely already feasible. The main limiting factor is the sequential implementation of the global planner that computes a trajectory for each agent. The median is at $t = 0.02$s for five agents, which is approximately double the computation time required for three agents. The computation time t_{mpc} for the MPC increases a linear increase with the number of agents considered, being $t = 0.09$s for three $n = 3$ and $t = 0.16$s for five agents $n = 5$.

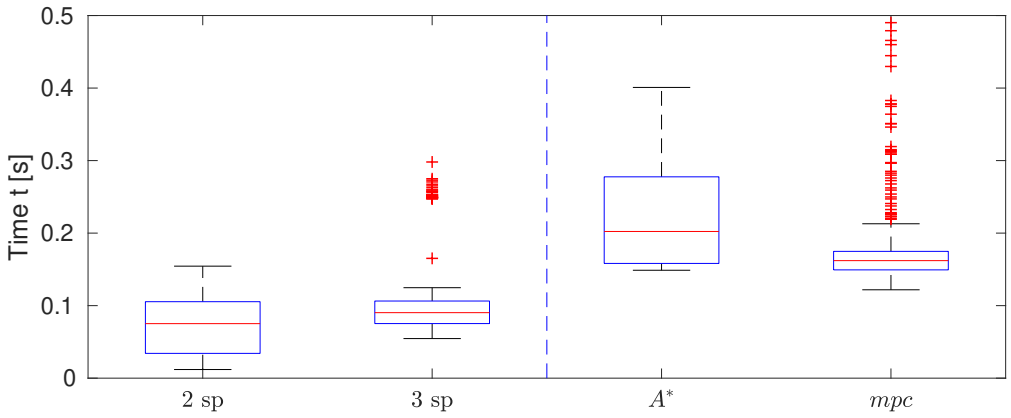


Figure 8-36: Computation times for the set-points (sp) of multiple commands, the global planner (A*) and MPC

Chapter 9

Conclusion

High-level command mapping for multiple aerial agents for a cinematographic application has been considered given a verbal command by an operator. The focus is on autonomously framing of up to two targets on the image plane of a number of agents specified by an operator. As the focus was not on NLP, the assumption was made that processing of the initial verbal command has been formerly performed into the CDC structure. A minimum input approach is applied, taking into account the variation in NL to express a command. Making use of a cinematographic taxonomy, the desired behaviour of the system can be determined from the CDC, resulting in quantifiable set-points for agents to achieve. An assignment of the set-points to the agents by making use of a LBA. The set-points are determined relative to the targets, whereby the computation of a global trajectory through the environment is required given the current agent position. A sample-based A* search algorithm is used to determine a collision-free trajectory. Given the cinematographic set-points and the global trajectory, an MPC computed the inputs for individual agents to achieve the operator command.

The CDC defined in Chapter 3 allows an operator to specify the desired behaviour of the system, accounting for the level of detail provides in the command. The minimum required input is the number of agents, number of targets and an identifier for the targets. Additional information can be provided pertaining to cinematographic taxonomy, which is inferred when omitted. The input allows for the determination of desired image variables.

When capturing a target within the view of a camera, the most dominant feature is the orientation of the head. Considering two targets, both their orientations have to be taken into account, along with their relative orientation in the global reference frame when determining camera positions. A classifications is defined into configurations in Chapter 4, from which set-points can be determined using a selection of five behaviours, see Chapter 5. A selection is made based on the configuration and operator command.

Each agent requires a low-level input to achieve the desired behaviour. A state-of-the-art MPC, with minor modifications, allows for the computation of input commands in real time [1, 2].

The best performance is achieved for the use case two agents within a dynamic and static environment during the verification of the methodology. A single agent is limited to a master shot, having to always track both targets to give an overview of the situation. Two agents provide more flexibility due to the increase in camera views that can be considered with respect to the determined target configuration. Three agents increase the camera positions further. A failure rate of 14% is observed for two agents framing targets with a dynamic orientation given the operator command. This failure rate increases to 27% for three agents.

The dominant limitations that cause deviation from the operator command are the agent dynamics, collision avoidance and mutual visibility. To ensure safe operation in an environment with people, the velocity is limited to $v_{lim} = 1.5\mathrm{m\,s^{-1}}$. When a rapid succession of configurations in motion occurs, the agents are unable to follow the global trajectory and achieve the desired cinematographic set-points in the time spent in the configuration. This results in partial tracking of targets, whereby the set-points are achieved for the target closest to the agent. Collision avoidance of obstacles to maintain safe operation can result in a signification deviation from the set-points, making the camera shot unusable. This can be mitigated by making is of multiple agents, creating a range of camera angles to compensate for these deviations. However, the increase in the number of agents clutters the environment relative to the targets. Proximity of camera positions causes deviations from the set-points to prevent inter-agent collision avoidance. A second undesirable consequence of multiple agents is the mutual visibility of the agents in the camera view. Three agents experience the most deviations from desired set-points as they have to take into account two agents. Deviations occur both due to mitigating the view of other agents or an agent being in full view which makes the shot unusable. When transitioning across an action line on counter sides of the target pair, this is an unavoidable artefact in the recording that is promptly mitigated when the correct action side is achieved.

Real time performance is achieved by the high-level command interpretation for all agent numbers, whereby the limiting computational factor is the determination of feasible set-points when making use of three agent. Generating sequential global trajectories experiences a linear increase in computations time for the number of agents used.

An extension to utilise multiple commands is presented to increase the number of agents that can be used. The combining of commands allows for framing of more than two target, where a single command is still limited to two targets. This increases the flexibility with respect to the use cases for the system, but also introduces new complications. When target pairs have contradicting action sides, mutual visibility of the increased number of agents is unavoidable. Smart occlusion by obstacles or collective viewpoint optimisation for the agents based on cinematographic practices would have to be considered to mitigate this. The number of agents that can be used scales threefold with

the number of commands that can be specified simultaneously. Computations of the set-points is highly dependable on their feasibility and can scale from linear with the number of commands to four times. However, the global planner and MPC scale linearly with the number of agents

An extension is presented to utilise multiple commands to increase the number of agents that can be used and the number of targets that can be framed. The limitation here is that one command is limited to three agents and two targets. This increases the flexibility with respect to the use cases of the system, but also introduces new complications. When contradicting action side of target pairs are determined, mutual visibility of the increased number of agents is unavoidable. Smart occlusion by obstacles in the requirement would have to be considered to mitigate this or collective viewpoint optimisations for all agents based on cinematographic practices. The number of agents that can be used scales threefold with the number of commands that can be specified simultaneously. Computation of the set-points is highly dependable on the feasibility of the setpoints in the environment and can scale from linear with the number of commands to four times. However, the global planner and MPC scale linearly with the number of agents.

The research has successfully determined a high-level command methodology to achieve aerial cinematography with multiple agents given a processed verbal command. The methodology combines cinematographic principles and control methodologies to control agents through a dynamic scene while mitigating collision.

Future Work

As the processing of NL was not the focus during the thesis, a future step could be the implementation of processing of the verbal commands. The approach presented in [21], whereby a CRF is trained for navigation, could be trained on a corpus of cinematographic terms. A open-source speech-recognition program such as Pocketsphinx could be implemented to allow for real-time command input [43].

The addition of a timing element such that a time dependent command structure can be introduced. An initial set-up is used for the final simulations. However, this implementation can be improved and integrated into the GUI or NL structure. This could allow for a sequential command specification by the operator which could be executed at predefined times.

Considering the current software implementation, the MPC computations are performed sequentially in an instance of MATLAB. Use is made of the FORCES Pro [41, 42], which requires a computer-bound license. This does not allow for usage on machines bereft of a license. Also, the python variant of the software does not allow the definition of non-linear constraints essential in the current implementation. Development of an MPCC making use of ACADO [44] is presented in [45]. It demonstrates collision avoidance in a dynamic environment, while adhering to a specified trajectory. Making this transition

would remove the dependence on a license and allows for the creation of ROS nodes to shift from a sequential approach to a parallel approach.

Throughout the thesis, the environment is assumed to be completely defined. An interesting avenue for research would be the implementation of a localization methodology for obstacles and targets within the environment such as presented in [15]. An alternative would be to analyse the camera feed of each agent and make use of image processing software such as OpenCV for image processing to determine the locations of desired targets [46]. This would require an initial description of the target for classification, along with classifications for obstacles.

Appendix A

Cinematographic Elaboration

The following image, see Figure A-1, presents a visual representation of the possible sizes that can be considered when framing a human [4]. Although nine relative size are presented, a selection is made of the most frequently used. These are the Full shot, Medium Shot, Close Shot and Full Close-up.

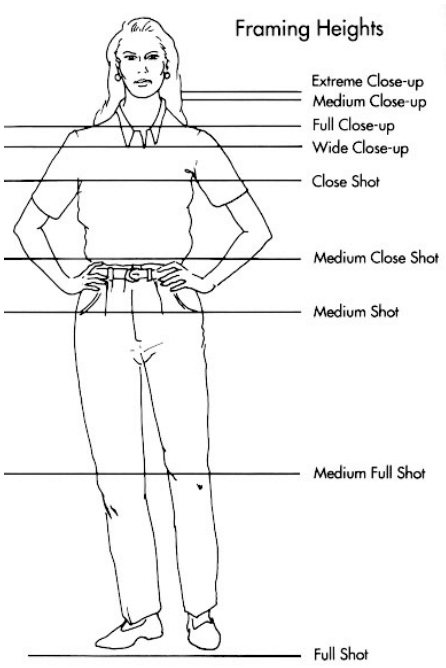


Figure A-1: Farming options for a human as presented in [4]

Appendix B

Simulation Supplement

This Appendix contains additional information pertaining to the simulations presented in Chapter 8. The visualisation tool is presented in Appendix B-2. The weights used for the cost terms in the MPC are presented in Table B-1. Additional information to complement the simulation can be found in Appendices B-3 to B-5, respectively.

B-1 MPC Weights

Table B-1 contains the weights used for the cost terms in the MPC. The slack variable weight is set to $\lambda = 10^4$.

Table B-1: MPC weights

Variable	Value	Variable	Value
w_{coll_a}	20	w_{coll_t}	40
w_{μ}	20	w_{σ}	0.1
$w_{pos_{x,y}}$	8	w_{pos_z}	10
$w_{\theta,\phi}$	2	w_{v_z}	4
$w_{\dot{\psi}}$	0.1	w_{mv}	10

B-2 Visualisation tool

The visualisation tool created to evaluate the simulations is presented below. Figure B-1 shows an overview of the environment, containing the targets and desired number of

agent to perform the simulation. Figure B-2 illustrates the panel created to simulate the CDC, whereby drop-down menus allow for the setting of the labels and values can be input when required.

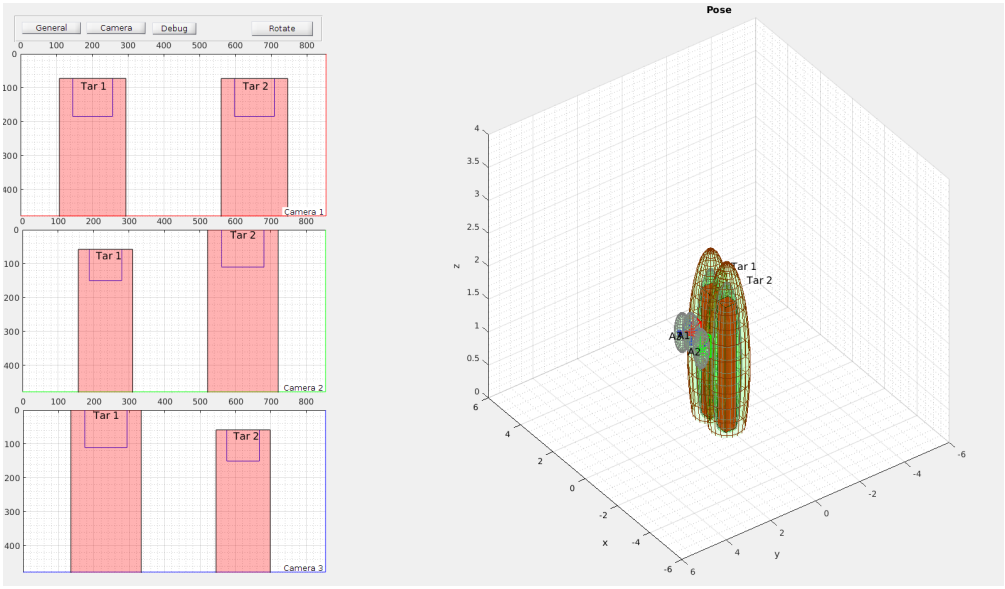


Figure B-1: Visualisation of the environment and the camera views of the agents.

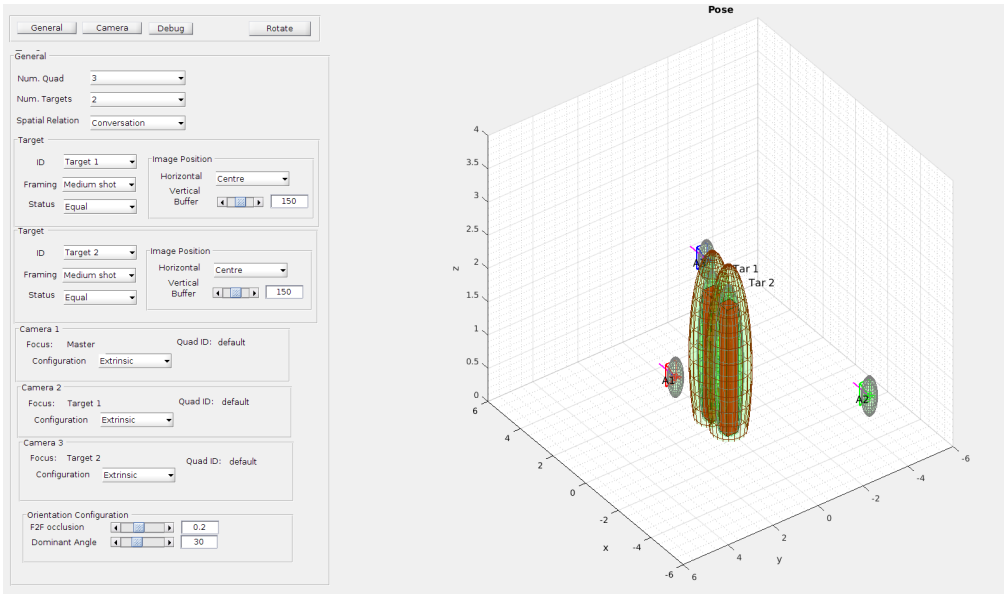
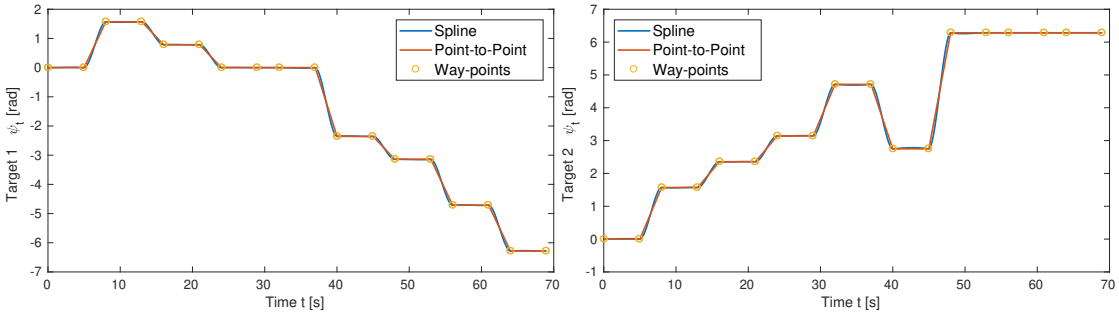


Figure B-2: Visualisation of the environment and the CDC input panel

B-3 Simulation: Dynamic Target Orientation

In the first simulation, use is made of static obstacles for which the yaw transits over time. Figure B-3 present the yaw ϕ_t trajectory for each target, respectively. As illustration, the control inputs for the second agent in the two agent situation are presented in Figure B-4. The inputs θ_u and φ_u show capping due to the limits set on the range. Clearly visible are the occurrences of a change in configuration of the targets.



(a) Yaw ψ_t for Target 1 **(b)** Yaw ψ_t for Target 2

Figure B-3: The yaw ψ_t trajectories for the first simulation

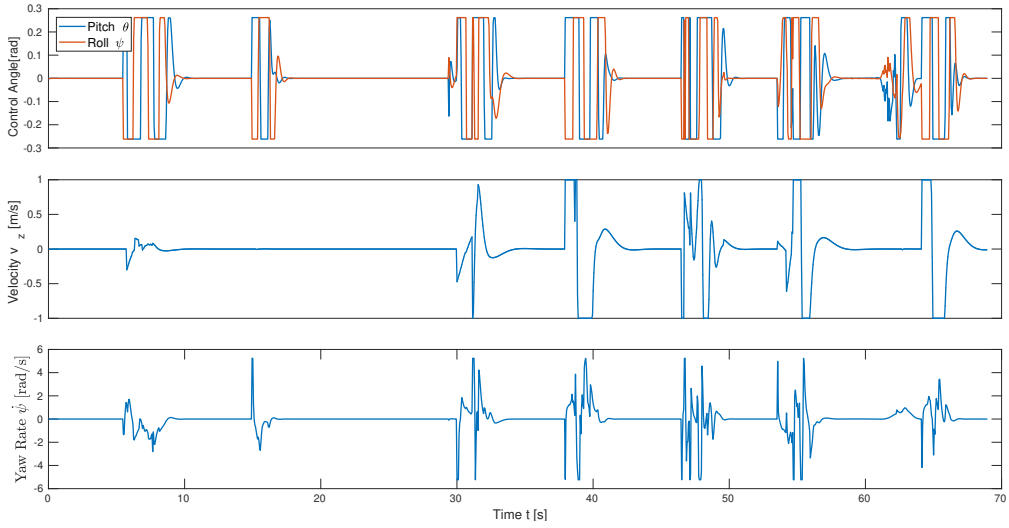
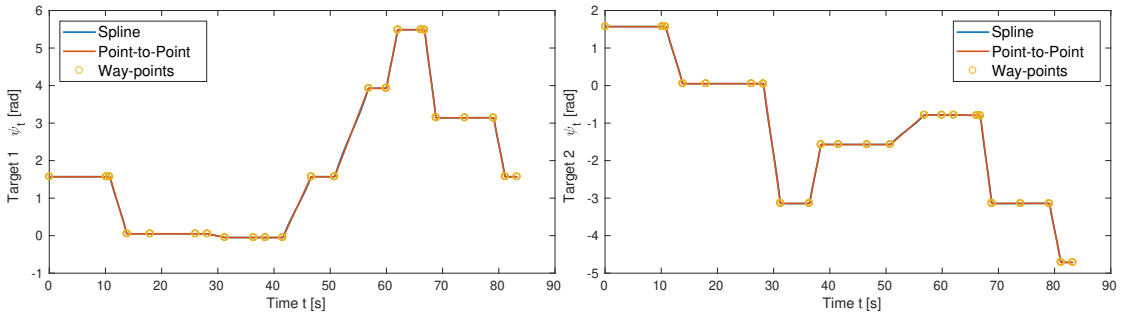


Figure B-4: The input $\mathbf{u}_a$ for the second agent during the dynamic target orientation simulation

B-4 Simulation: Dynamic Motion

In the second simulation, the trajectories of the targets and their motion has to be considered. The respective orientation trajectories of the targets can be found in Figure B-5.



(a) Yaw ψ_t for Target 1 **(b)** Yaw ψ_t for Target 2

Figure B-5: The yaw ψ_t trajectories for the second simulation

The trajectory the targets execute on the horizontal xy-plane is presented in Figure B-6. A clock-wise execution of the trajectories is performed, whereby initialisation and culmination are in the start point indicated for the agents. The configurations the targets are in over the course of the simulation can be found in Figure B-7

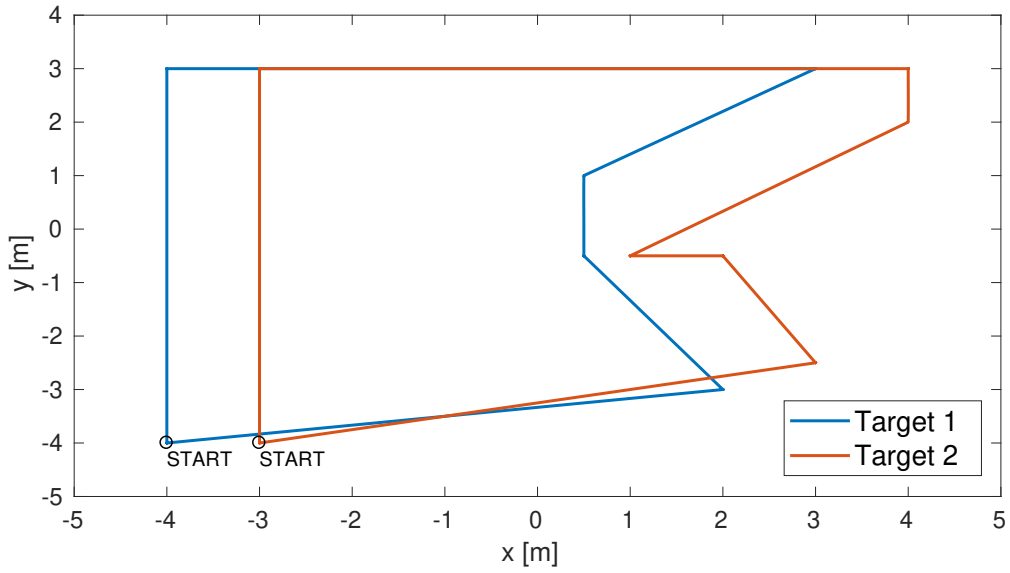


Figure B-6: Target trajectories specifying their dynamic motion through the environment

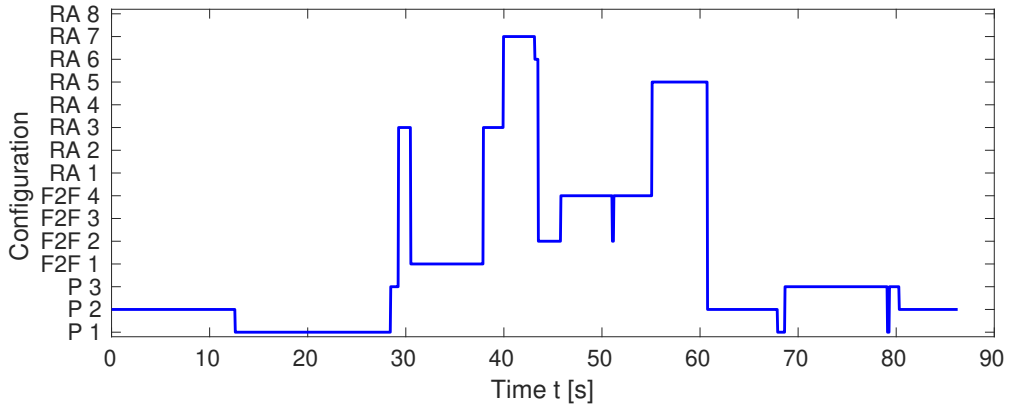


Figure B-7: Configurations of the targets throughout the simulation

B-5 Simulation: Multiple Commands

The commands used for the multiple command simulation are presented in Table B-2. The situation and combination the command occur in is outlined in Table B-3.

Table B-2: Command specification for the multi-command simulation

Command	Agents n	Targets	Spatial Relation	Target Status
1	1	Target 1, Target 2	Conversation	
2	2	Target 3	Cinematic	
3	2	Target 4, Target 5	Conversation	[Dominant, Submissive]
4	2	Target 2, Target 3	Conversation	
5	1	Target 1	Front View	
6	1	Target 1, Target 4	Conversation	
7	1	Target 4, Target 5	Conversation	
8	2	Target 5	Front View	
9	2	Target 5, Target 2	Conversation	[Dominant, Submissive]
10	1	Target 2, Target 3	Conversation	

Table B-3: The command combination over time

Time t [s]	Motion	Commands
0	Stationary initialisation	[1, 2, 3]
3.1	Target 3 moves to join Target 1 & 2	[1, 2, 3]
6.2	Target 3 pauses in line with Target 1 & 2	[2, 3, 4]
11.3	Target 1 moves to join Target 4 & 4	[3, 4, 5]
14.4	The number of agents required is reduces by one	[4, 6, 7]
19.5	A pause of $t = 2$s occurs	[4, 6, 7]
21.6	Target 1 & 4 exit the environment and Target 5 moves towards Target 2 & 3	[4, 8]
27.7	Target 5 has joined Target 2 & 3	[9, 10]

Appendix C

Mambo

A new Parrot drone has been acquired to make use of in the Cognitive Robotics (COR) Lab. Presently, the main drone type used is the Parrot Bebop 2. It is quadcopter that weights 295g and offers 10 minutes of autonomous flight time according to the website[1]. A full High Definition (HD) camera is attached to the main frame and uses Wi-Fi as communication type. The main disadvantage of making use of the Bebop is that there is limited indoor workspace. This restricts the number of drones that can fly simultaneously to at most 3. Therefore, a Parrat Mambo has been purchased.

The Parrot Mambo is a quarter of the size of the Parrot Bebop2. A larger number of drones could be operating within the workspace simultaneously. The Mambo is also a quadcopter that weighs 63g and has a flight time of 10 minutes. For stabilization the drone makes use of an Inertial Measurement Unit (IMU), ultrasound sensor, pressure sensor and camera sensor. The drone is equipped with a LEGO configuration on the top-side, where accessories can be attached. This allows for the addition of a forward facing HD camera or a grabber. Communication is achieved through Bluetooth for the Mambo without any accessories. Flying the Mambo with the camera accessories attached with the mobile application, connection has to be made with the Mambo through Bluetooth and to the camera with Wi-Fi. Finally, the Parrot website indicate that there is a Software Development Kit (SDK) available for use on Linux.

C-1 Linux Setup

Robot Operating System (ROS) is used on the Linux system to connect to the Bebop 2, joystick, Optitrack system and MPC controller. A open-source ROS driver can be found

[1]Parrot mambo FPV: https://www.parrot.com/global/drones/parrot-mambo-fpv (Accessed 22 November 2018)

for the Bebop 2. The Parrot Mambo currently does not have a open-source ROS driver that can be used. Therefore, a usable ROS driver is created to connect to the Mambo. A requirement for the driver is that it can be used modularly, such that switching between the two types of drones can be performed easily.

The first step is to search for a package that allows for connection to the Mambo outside of ROS. The PyParrot package[2] provides a great basis for a ROS driver. It contains the possibility to connect to the Mambo through Bluetooth or Wi-Fi. Also, a Mambo class is defined with all the functions required to control and manoeuvre the drone. To achieve modularity, the message structure for the Mambo ROS driver should match the Bebop ROS driver.

As the PyParrot package is written in Python, the ROS driver that wraps around it is also implemented in Python, contrary to the C++ implementation of the Bebop driver.

C-2 Establish Connection

Depending on the configuration of the Mambo, the connection type and method might vary. Therefore, both the connection through Bluetooth and Wi-Fi are elaborated on.

Starting with the Bluetooth connection, first the address for the Mambo needs to be determined. A convenient script is provided in the Pyparrot package which checks the discoverable addresses to identify a Mambo address. This address can then be set in the launch file for the Mambo. A connection needs to be established through running the driver. Do not pair with the Mambo as this will not establish a correct communication over which commands can be sent. In the launch file, set the use_{wifi} parameter to false.

Depending on the Bluetooth capability of the desktop or laptop used, a Bluetooth dongle is available to connect. A recommendation is to always use the dongle to connect, as generally the drivers implemented in laptops are of Class 2. This means that the connection distance has a limit of 10 meters. This does not take into account possible obstacles between the Mambo and the computer. The dongle is of Class 1, meaning that the distance it can communicate over is limited to 100 meters. This also does not take into account possible obstacles. However, it does provide a stronger connection at the same distance.

When a camera is connected to the Mambo, both connection types can be used. To fly a Mambo with your phone, both connections are required. This does not apply to the ROS driver. The Mambo can be controlled through the Wi-Fi connection alone. Noted has to be that the Wi-Fi connection is provided by the camera accessory and can only be used when it is connected.

The Mambo currently experiences communication loss due to unknown causes. Also, when a landing command is issued, it is unable to take-off again as it remains in the state where it sends the command to land.

[2]PyParrot package: https://pyparrot.readthedocs.io/en/latest/ (Accessed 22 November 2018)

Bibliography

[1] T. Nägeli, J. Alonso-Mora, A. Domahidi, D. Rus, and O. Hilliges, "Real-time motion planning for aerial videography with dynamic obstacle avoidance and viewpoint optimization," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1696–1703, 2017.

[2] T. Nägeli, L. Meier, A. Domahidi, J. Alonso-Mora, and O. Hilliges, "Real-time planning for automated multi-view drone cinematography," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, p. 132, 2017.

[3] D. Arijon, *Grammar of the film language*. Focal Press London, 1976.

[4] S. D. Katz, *Film directing shot by shot: visualizing from concept to screen*. Gulf Professional Publishing, 1991.

[5] M. Schwager, B. J. Julian, and D. Rus, "Optimal coverage for multiple hovering robots with downward facing cameras," in *2009 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3515–3522, IEEE, 2009.

[6] M. Schwager, B. J. Julian, M. Angermann, and D. Rus, "Eyes in the sky: Decentralized control for the deployment of robotic camera networks," *Proceedings of the IEEE*, vol. 99, no. 9, pp. 1541–1561, 2011.

[7] T. Stirling, J. Roberts, J.-C. Zufferey, and D. Floreano, "Indoor navigation with a swarm of flying robots," in *2012 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4641–4647, IEEE, 2012.

[8] F. Augugliaro, S. Lupashin, M. Hamer, C. Male, M. Hehn, M. W. Mueller, J. S. Willmann, F. Gramazio, M. Kohler, and R. D'Andrea, "The flight assembled architecture installation: Cooperative construction with flying machines," *IEEE Control Systems*, vol. 34, no. 4, pp. 46–64, 2014.

[9] R. Ritz and R. D'Andrea, "Carrying a flexible payload with multiple flying vehicles," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3465–3471, IEEE, 2013.

[10] F. Augugliaro, A. P. Schoellig, and R. D'Andrea, "Dance of the flying machines: Methods for designing and executing an aerial dance choreography," *IEEE Robotics & Automation Magazine*, vol. 4, no. 20, pp. 96–104, 2013.

[11] J. Alonso-Mora, M. Schoch, A. Breitenmoser, R. Siegwart, and P. Beardsley, "Object and animation display with multiple aerial vehicles," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1078–1083, IEEE, 2012.

[12] I. Mademlis, V. Mygdalis, N. Nikolaidis, and I. Pitas, "Challenges in autonomous uav cinematography: an overview," in *2018 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6, IEEE, 2018.

[13] C. Gebhardt, B. Hepp, T. Nägeli, S. Stevšić, and O. Hilliges, "Airways: Optimization-based planning of quadrotor trajectories according to high-level user goals," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pp. 2508–2519, ACM, 2016.

[14] Q. Galvane, J. Fleureau, F. Tariolle, and P. Guillotel, "Automated cinematography with unmanned aerial vehicles," in *Proceedings of the Eurographics Workshop on Intelligent Cinematography and Editing*, pp. 23–30, Eurographics Association, 2016.

[15] N. Joubert, D. B. Goldman, F. Berthouzoz, M. Roberts, J. A. Landay, P. Hanrahan, *et al.*, "Towards a drone cinematographer: Guiding quadrotor cameras using visual composition principles," 2016.

[16] A. Gomes, C. Rubens, S. Braley, and R. Vertegaal, "BitDrones: towards using 3D nanocopter displays as interactive self-levitating programmable matter," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pp. 770–780, ACM, 2016.

[17] J. Alonso-Mora, S. H. Lohaus, P. Leemann, R. Siegwart, and P. Beardsley, "Gesture based human-multi-robot swarm interaction and its application to an interactive display," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5948–5953, IEEE, 2015.

[18] J. Nagi, A. Giusti, L. M. Gambardella, and G. A. Di Caro, "Human-swarm interaction using spatial gestures," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),*, pp. 3834–3841, IEEE, 2014.

[19] J. Nagi, H. Ngo, L. M. Gambardella, and G. A. Di Caro, "Wisdom of the swarm for cooperative decision-making in human-swarm interaction," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1802–1808, IEEE, 2015.

[20] S. Pourmehr, V. M. Monajjemi, R. Vaughan, and G. Mori, "'You two! Take off!": Creating, modifying and commanding groups of robots using face engagement and indirect speech in voice commands," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 137–142, IEEE, 2013.

[21] A. S. Huang, S. Tellex, A. Bachrach, T. Kollar, D. Roy, and N. Roy, "Natural language command of an autonomous micro-air vehicle," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2663–2669, IEEE, 2010.

[22] S. Stevšić, T. Nägeli, J. Alonso-Mora, and O. Hilliges, "Sample efficient learning of path following and obstacle avoidance behavior for quadrotors," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3852–3859, 2018.

[23] H. Zhu and J. Alonso-Mora, "Collision avoidance for mavs in dynamic environments using chance-constrained receding horizon control,"

[24] E. Cambria and B. White, "Jumping nlp curves: A review of natural language processing research [review article]," *IEEE Computational Intelligence Magazine*, vol. 9, pp. 48–57, May 2014.

[25] D. Arumugam, S. Karamcheti, N. Gopalan, E. C. Williams, M. Rhee, L. L. Wong, and S. Tellex, "Grounding natural language instructions to semantic goal representations for abstraction and generalization," 2017.

[26] T. Kollar, S. Tellex, D. Roy, and N. Roy, "Toward understanding natural language directions," in *Proceedings of the 5th ACM/IEEE international conference on Human-robot interaction*, pp. 259–266, IEEE Press, 2010.

[27] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.

[28] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[29] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the society for industrial and applied mathematics*, vol. 5, no. 1, pp. 32–38, 1957.

[30] R. E. Burkard and E. Cela, "Linear assignment problems and extensions," in *Handbook of combinatorial optimization*, pp. 75–149, Springer, 1999.

[31] H. Zhu and M. Zhou, "Efficient role transfer based on kuhn–munkres algorithm," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 42, no. 2, pp. 491–496, 2012.

[32] U. Pferschy, "The random linear bottleneck assignment problem," *RAIRO-Operations Research*, vol. 30, no. 2, pp. 127–142, 1996.

[33] M. Turpin, K. Mohta, N. Michael, and V. Kumar, "Goal assignment and trajectory planning for large teams of interchangeable robots," *Autonomous Robots*, vol. 37, no. 4, pp. 401–415, 2014.

[34] P. S. Pundir, S. K. Porwal, and B. P. Singh, "A new algorithm for solving linear bottleneck assignment problem," *Journal of Institute of Science and Technology*, vol. 20, no. 2, pp. 101–102, 2015.

[35] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[36] L. Xia, X. Jun, C. Manyi, X. Ming, and W. Zhike, "Path planning for uav based on improved heuristic aâĹŮ algorithm," in *Electronic Measurement & Instruments, 2009. ICEMI'09. 9th International Conference on*, pp. 3–488, IEEE, 2009.

[37] T. Khuswendi, H. Hindersah, and W. Adiprawita, "Uav path planning using potential field and modified receding horizon a* 3d algorithm," in *Electrical Engineering and Informatics (ICEEI), 2011 International Conference on*, pp. 1–6, IEEE, 2011.

[38] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[39] J. Mattingley, Y. Wang, and S. Boyd, "Receding horizon control," *IEEE Control Systems*, vol. 31, no. 3, pp. 52–65, 2011.

[40] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, p. 5, Kobe, Japan, 2009.

[41] A. Domahidi and J. Jerez, "FORCES Professional." embotech GmbH (http://embotech.com/FORCES-Pro), July 2014.

[42] A. Zanelli, A. Domahidi, J. Jerez, and M. Morari, "FORCES NLP: an efficient implementation of interior-point methods for multistage nonlinear nonconvex programs," *International Journal of Control*, 2017.

[43] D. Huggins-Daines, M. Kumar, A. Chan, A. W. Black, M. Ravishankar, and A. I. Rudnicky, "Pocketsphinx: A free, real-time continuous speech recognition system for hand-held devices," in *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, vol. 1, pp. I–I, IEEE, 2006.

[44] B. Houska, H. Ferreau, and M. Diehl, "ACADO Toolkit – An Open Source Framework for Automatic Control and Dynamic Optimization," *Optimal Control Applications and Methods*, vol. 32, no. 3, pp. 298–312, 2011.

[45] B. Floor, B. Brito, L. Ferranti, and J. Alonso-Mora, "Local model predictive contouring control for dynamic environments,"

[46] G. Bradski and A. Kaehler, "Opencv," *Dr. DobbâĂŹs journal of software tools*, vol. 3, 2000.

Glossary

List of Acronyms

MPC	Model Predictive Control
NLP	Natural Language Processing
NL	Natural Language
HRS	Human-Robot System
SDC	Spatial Description Clause
CDC	Cinematographic Description Clause
GUI	Graphical User Interface
FSM	Finite State Machine
UAV	Unmanned Aerial Vehicle
LBA	Linear-Bottleneck Algorithm
UAV	Unmanned Aerial Vehicle
LQR	Linear-Quadratic Regulator
ORCA	Optimal Reciprocal Collision Avoidance
MPCC	Model Predictive Contouring Control
IQP	Iterative Quadratic Programming
RMS	Root-Mean Squared
ROS	Robot Operating System

IMU	Inertial Measurement Unit
HD	High Definition
SDK	Software Development Kit
COR	Cognitive Robotics
RMS	Root-mean Squared
RTK	Real Time Kinematic
GPS	Global Positioning System
CRF	Conditional Random Field
SQP	Sequential Quadratic Programming

Nomenclature

Greek Symbols

Symbol	Description
α	Rotation angle
β	Camera view angle
$\boldsymbol{\psi}$	Heading vector
δ	Sampling factor
ϵ	Action range
η	Overlapping factor
γ	Relative angle
μ	Image Axis
Ω	Ellipsoid for obstacle
ψ	Yaw angle
σ	Size
τ	Time constant
θ	Pitch angle
φ	Roll angle
ζ	Dominant angle

Roman Symbols

Symbol	Description
a	Action line
C	Cost matrix

m	Image position
n	Normal vector
p	Position vector
q	Orientation quaternion
s	Desired state
u	Input
x	State vector
C	Image centre position
d	Distance
e	Error
f	Focal length
h	Height
k	Time step
l	Length
m	Number of targets
N	Time Horizon
n	Number of agents
r	Radius
S	Sphere around agent
s	Slack variable
w	Width
x	Position along x-axis
y	Position along y-axis
z	Position along z-axis

Subscripts

Symbol	Description
a	Agent
buff	Buffer
c	Camera
cdc	Cinematographic Description Clause
d	Desired
d	Desired
f	Feasible
fr	Front view
gimb	Gimbal
h	Head
I	Image plane
ip	Image projection point

max	Maximum limit
min	Minimal limit
o	Obstacle
op	Operator
proc	Processed (NL command)
ra	Right Angle
t	Target
w	World coordinate frame
x	X-axis
y	Y-axis