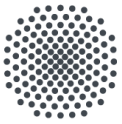

University of Stuttgart
Germany

Studienarbeit

Review of Formation Control Approaches and their Implementation Considering a Realistic Model of Mobile Robots

Hannes Wind

Supervisors: Prof. Dr. rer. nat. habil. Thomas Bräunl
Prof. Dr.-Ing. Dr. h.c. Oliver Sawodny
M.Sc. Florian Morlock
Submitted: 04.05.2016

Declaration of Authorship

I hereby declare that the thesis submitted is my own unaided work. All direct or indirect sources used are acknowledged as references.

Stuttgart, 04.05.2016

.....

Hannes Wind

Abstract

This work investigates the formation control applied on real mobile robots. Therefore, a literature research regarding the formation controller for mobile robots is conducted. Several formation controllers are investigated concerning the ability of implementing them on the real robots. In order to perform these simulations, a dynamic model of the robot is considered. To obtain the parameter of the dynamic model, a model identification is accomplished. Furthermore, several extensions to this model are added, in order to achieve a more realistic model of the robot and the whole system. Once a suitable formation controller is chosen, the implementation on the real robots is carried out. The results of this implementation are highlighted and some practical aspects are discussed. In addition to that, an obstacle avoidance controller is proposed. In order to verify this controller, simulations are performed and to be able to validate the collision avoidance controller, further experiments are carried out.

Acknowledgements

I would like to thank the following people for their support and assistance in this project:

Prof. Thomas Bräunl for his supervision at the University of Western Australia and his contributions to this project.

Prof. Oliver Sawodny for his guidance at the University of Stuttgart and for giving me the opportunity to study abroad.

Florian Morlock for his supervision at the University of Stuttgart and his support.

Marcus Pahm and Franco Hidalgo for their support regarding the robots.

Linda Barbour for her support in any matters and the full proof-read of my thesis.

Contents

Chapter 1

Introduction

There are many advantages of controlling multiple mobile robots. For achieving a task, which might require one sophisticated and expensive robot, a group of simple robots can be sufficient [SB00]. It can also contribute to the robustness and the efficiency of the system [YZ05], [SHP04]. The applications are widespread and range from automated highway systems [RI96], [Ben91] over underwater vehicles [HF10] to achieving formations with satellites [Ser03].

This thesis considers the implementation of formation controllers on given mobile robots. Therefore, the literature is studied in order to obtain different formation controllers. To be able to gain an insight of these formation controllers, several simulations are carried out. Thereby a continuous improvement of the model is performed. One improvement concerns a parametric dynamic model, whereby a model identification takes place in order to identify the parameter. The results of these simulations are reconsidered during the implementation on the real robots. Several experiments are carried out to verify the simulations and the ability of implementing the formation controller on the robots. Furthermore, an obstacle controller is proposed, based on a formation controller. In order to verify this controller, several simulations are carried out. The validation ensues via experiments on the real robots.

There are several control tasks to achieve with mobile robots. The most basic task could be considered as following a given trajectory [dLOS98], known as trajectory tracking. Another motion task is to achieve a group formation. The goal of this method is to change the position of each robot in order to generate a group formation [YB96]. This formation control could be considered an extension to the trajectory tracking problem where robots follow a given trajectory while performing a formation.

The trajectory tracking problem was solved in [SA91] by using a time-varying control law as a feedback. Another way to solve this problem is through the use of dynamic feedback linearization [dNCB95], [De 93] and [OdLV02].

In [ZN99] and [TKCC01] the technique of backstepping is used for trajectory tracking of nonholonomic systems. The approach in [PLNS98] takes advantage of the cascade structure of the robot model.

The group formation control is considered as a control law which controls the mobile robots in a certain formation, e.g. a triangular. In [YB98] the authors proposed an approach to achieve this goal by a time-varying feedback control. Whilst [ZFM05] gives necessary and sufficient graphical conditions for the formation stabilization to a point and to more general geometric pattern.

There are various approaches to formation tracking, like the behavior-based method [BA98], the virtual structure strategy [LT97] and the leader-follower approach [DOK98], [TNO04] and [CSVS03]. The behavior-based approach defines a desired behavior for each robot. In order to derive the control, the relative importance of each behavior is weighted [LBY03]. This approach is used, for example, to apply the social characteristics of insects and animals to multirobot systems [AAC09]. The virtual structure method treats the entire formation as a single entity. To derive the control for each robot, the motion of the virtual structure is considered and transformed into the motion of the robot [GSM08] and [LJ12].

In this thesis the focus is on the leader-follower approach. To accomplish the leader-follower strategy there are several approaches. The method taken [LA06] made use of the cascade approach as in [PLNS98] to achieve a consensus-based controller. In [DOK98], feedback linearization was used to exponentially stabilize the distance and the orientation between the leader and the follower. In addition to these approaches, model predictive control (MPC) has become an accepted method in solving the formation control problem. In [HYC$^+$08] a nonlinear MPC is applied. Since the computational effort is higher than applying a linear MPC, [KY15] uses a linear MPC in combination with an input-output feedback linearization to achieve a leader-follower formation control.

Related works, where an implementation of a formation controller takes place are [FDKO01], [TO03], [CY04] and [CC12]. These papers also perform simulations of the formation controller. However, non of them considers either a dynamic model or uncertainties of the robot. Therefore, this work takes an extended model of the robot into account and performs further simulations. The outcome of these investigations shows the ability of implementing a formation controller on the given robots.

Chapter 2

Robot Identification

In this section an investigation of the given robot takes place regarding its sensors, actuators, computational power and communication ability. The results of these investigations are required for the subsequent sections. The possibility of implementing a formation controller depends highly on the ability of the robot. The appearance of the mobile robot can be seen in Figure 2.1.

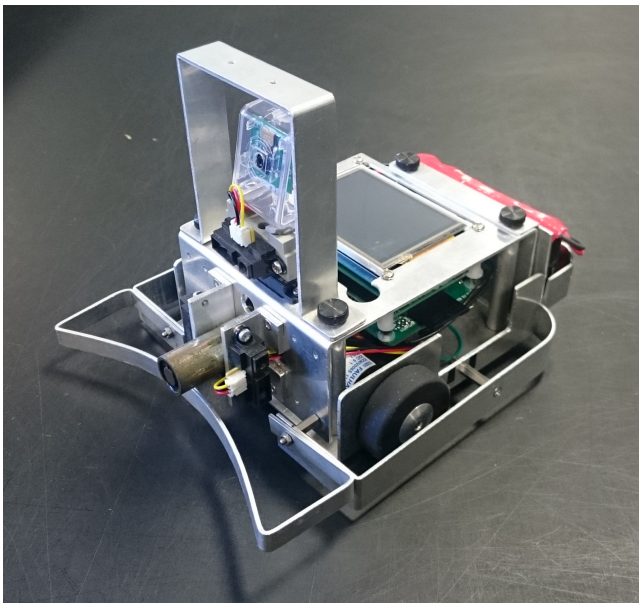

Figure 2.1: Appearance of the mobile robot.

2.1 Hardware

The hardware in the sense of the computational ability is divided into two parts. On the one hand there is the high level which is basically a Raspberry Pi board. On the other hand the sensors and actuators are operated by the low level.

2.1.1 High Level

As mentioned above, the high level is made up of a Raspberry Pi. This board uses a Linux kernel-based operation system with a 700 MHz single core processor and a 256 MB memory. For implementing functions on the high level, the programming language C++ is used. There are already some existing functions for exchanging data with the low level e.g. for setting the speed of the motors in the low level or for requesting the position. Since the high level is operated by an operating system, a constant sample time cannot be guaranteed. Considering the computational power and the given interface with the low level, the high level is made for executing the formation controller.

2.1.2 Low Level

The low level is designed to request and process the data of the sensors and to drive the actuators. Therefore, a processor with 16 MHz and 8 MB memory is used. The board also contains the required hardware driver for the actuators. Since the low level has less computational power compared to the high level, it is necessary to keep the software simpler. Nevertheless the low level supports interrupts, which can be used for achieving a constant sample time.

2.2 Sensors

A crucial part of the formation controller is either to know the global position $\boldsymbol{q}(t) = [x(t), y(t), \phi(t)]^T$ or to determine the position between two robots. These requirements depend on the specific formation controller. The robots have basically two different types of sensors with which to obtain either the global position or the relative position between two robots. In the following sections the sensors are discussed in detail.

2.2.1 Position Sensitive Device

Each robot is equipped with three Position Sensitive Devices (PSD). They are mounted on the head of the robot, where one points into the head direction

and the two others to the left and to the right respectively. A PSD senses the distance to another object. However the output is a nonlinear function. This disadvantage can be solved via an inverse function, implemented in software. This linearization has to be done before the data are used. A further drawback is the mounted position of the PSDs in combination with the spot-like measurement of the distance. Thus, exists a dead space between each PSD where a robot cannot be detected. This could be avoided by adding more PSD to the robot. Since the robot should not be manipulated, another method to get the position is considered.

2.2.2 Position Calculation

The robots have a motor on each wheel with an attached encoder. This encoder detects the displacement of the wheel in each sample time of the low level. With this information a calculation of the increment position based on geometric relations can be done:

$$\begin{aligned} \Delta pos_{\mathrm{L}} &= \frac{e_{\mathrm{L}} - e_{\mathrm{old,L}}}{TPM_{\mathrm{L}}} \\ \Delta pos_{\mathrm{R}} &= \frac{e_{\mathrm{R}} - e_{\mathrm{old,R}}}{TPM_{\mathrm{R}}} \\ \Delta pos &= \frac{\Delta pos_{\mathrm{R}} + \Delta pos_{\mathrm{L}}}{2} \\ \Delta angle &= \frac{\Delta pos_{\mathrm{R}} - \Delta pos_{\mathrm{L}}}{d} \end{aligned} \tag{2.1}$$

with the current and previous encoder value $e_{\mathrm{L}}, e_{\mathrm{old,L}}$ of the left and $e_{\mathrm{R}}, e_{\mathrm{old,R}}$ the right encoder respectively. The constants TPM_{L} and TPM_{R} define the ticks per meter of the left and the right encoder. The parameter d is the distance between both wheels. The first two equations in (2.1) calculate the increment position of each wheel compared to the last sample. Δpos and $\Delta angle$ are the increment position and the increment angle of the robot. Considering these pre-calculations, the global position for time $k+1$ can be calculated as:

$$\begin{aligned} x(k+1) &= x(k) + \Delta pos(k) \cdot \cos(\phi(k)) \\ y(k+1) &= y(k) + \Delta pos(k) \cdot \sin(\phi(k)) \\ \phi(k+1) &= \phi(k) + \Delta angle(k) \\ x(0) &= x_0, y(0) = y_0, \phi(0) = \phi_0 \end{aligned} \tag{2.2}$$

For these calculations a few crucial things have to be considered. One consideration is of the parameters in (2.1). For practical purpose it is necessary to distinguish between the ticks per meter (TPM) of the left and the right wheel, since it is not guaranteed that both wheels have the same perimeter.

These parameters have to be determined for each robot separately. The other parameter is d. Since the wheels of the given robot are relatively wide, it is crucial to determine this distance for each robot.

Another part to consider is that of the encoder values. It is necessary that each encoder tick is registered. In the given configuration of the low level it could occur that ticks are missing. This is because of the reading of the encoder values which had been done via an interrupt. This interrupt service routine was called when the encoder had passed one tick. If there is a higher prior interrupt service routine in the execution while the encoder does more than one tick, there will be only one registered. In order to avoid this, the encoder values are counted via a hardware counter. With this change the accuracy of the position calculation could be improved especially at higher speed.

2.3 Actuators

In order to be able to drive the robot, two Direct Current (DC) motors are used. The output of the processor used for this purpose is the Pulse Width Modulation (PWM) output. Since the processor is not capable to drive the motor directly, a hardware driver is needed. This hardware driver supplies the needed electrical power. By varying the duty cycle of the PWM signal, the speed of the motor can be varied.

2.4 Communication

Besides the need of knowing the position of the robot to control the formation, it is also necessary for some formation controlling approaches to interact between the robots. The details regarding the required information are shown in Section 4. Since the robots did not have a working communication, this must be created.

For this approach, communication is considered between two robots and can be scaled afterwards. For the purpose of information exchange a wireless communication via WiFi is chosen, whereby the robots have to connect to a common router. The UPD-protocol is used as a communication protocol. This is an unconnected communication where packet loss can occur. This must be investigated, to ascertain if the occurrence of a packet loss can be tolerated by the specific formation controller. In order to choose a communication partner, unique identification numbers (ID) are given to each robot. Therefore, the last byte of the IP-address of the robot is mapped to the ID. To be able to communicate with another robot, only the ID of it has to be known. On the other hand if a message is received, the IP-address and hence the ID of the

sending robot can be determined. Therefore, functions in the high level for sending and receiving messages are provided.

Chapter 3

Model Setup

3.1 Kinematic Model

In this work, differentially driven wheeled mobile robots are considered. Each robot has two wheels and each wheel can be actuated independently. The description of a robot in a two dimensional space can be done by the current position (x, y) and the orientation angle ϕ. Hence the state vector results to $\boldsymbol{q}(t) = [x(t), y(t), \phi(t)]^T$. The kinematic equations of the robot are represented in (3.1):

$$\begin{aligned} \dot{x} &= v\cos(\phi) \\ \dot{y} &= v\sin(\phi) \\ \dot{\phi} &= \omega, \\ x(0) = x_0, y(0) &= y_0, \phi(0) = \phi_0 \end{aligned} \tag{3.1}$$

with the velocity v and the angular velocity ω as an input $u = [v, \omega]^T$. The connectedness of the velocity and angular velocity of the robot to the angular velocity of both wheels $(\omega_{\mathrm{W,R}}, \omega_{\mathrm{W,L}})$ is given by (3.2) with the radius r of a wheel.

$$\begin{aligned} v &= r\frac{\omega_{\mathrm{W,R}} + \omega_{\mathrm{W,L}}}{2} \\ \omega &= r\frac{\omega_{\mathrm{W,R}} - \omega_{\mathrm{W,L}}}{d} \end{aligned} \tag{3.2}$$

3.2 Dynamic Model

3.2.1 Unreduced Model

In this section the dynamics of a mobile robot are considered. Figure 3.1 shows the parameter of a robot, where B is the wheel baseline center, G is the center of gravity, C is the position of the castor, L is the position of the left wheel and R is the position of the right wheel. The geometric distances are expressed by b, c and d.

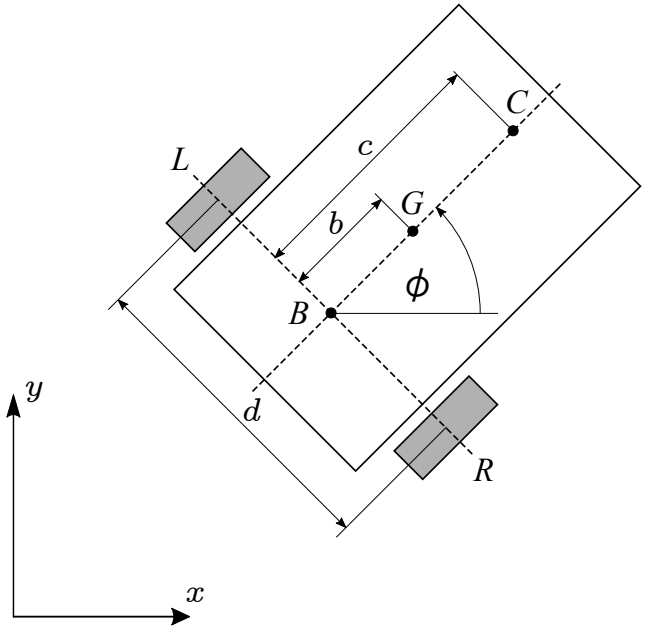

Figure 3.1: Parameter of the mobile robot.

The dynamic model of a robot was derived in [YDCV98]. The full dynamic equations of [YDCV98] are given in (3.3), where m is the mass of the robot. The uncertainty vector $\boldsymbol{\delta} = \left[\delta_x, \delta_y, 0, \delta_v, \delta_\omega\right]^T$ includes the slip speed of the wheels, the viscous friction forces and the resistance force of the castor. The input of the model is given by τ_v and τ_ω.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\phi} \\ \dot{v} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} v\cos(\phi) \\ v\sin(\phi) \\ \omega \\ \frac{mbr^2}{\theta_v}\omega^2 \\ -\frac{2mbr^2}{\theta_\omega} v\omega \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{2r}{\theta_v} & 0 \\ 0 & \frac{2rd}{\theta_\omega} \end{bmatrix} \begin{bmatrix} \tau_v \\ \tau_\omega \end{bmatrix} + \begin{bmatrix} \delta_x \\ \delta_y \\ 0 \\ \delta_v \\ \delta_\omega \end{bmatrix} \tag{3.3}$$

$$x(0) = x_0, y(0) = y_0, \phi(0) = \phi_0, v(0) = v_0, \omega(0) = \omega_0$$

with

$$\begin{aligned} \theta_v &= mR_\mathrm{t} + 2I_\mathrm{e}, & \theta_\omega &= I_\mathrm{e}d^2 + 2R_\mathrm{t}^2(I_\mathrm{z} + mb^2) \\ \tau_v &= \frac{1}{2}(k_\mathrm{t}i_\mathrm{L} + k_\mathrm{t}i_\mathrm{R}), & \tau_\omega &= \frac{1}{2}(k_\mathrm{t}i_\mathrm{L} - k_\mathrm{t}i_\mathrm{R}) \end{aligned} \tag{3.4}$$

(3.3) and (3.4) according to [YDCV98], where R_t is the nominal radius of the tire, I_e is the moment of inertia of the combined motor rotor and wheel, I_z is the moment of inertia of the robot about the vertical axis, k_t is the motor torque constant and i_L and i_R are the motor currents of the left and the right motors.

3.2.2 Reduced Model

The input signal of the motor is the duty cycle of the PWM signal, as mentioned in Section 2.3. Since the model in (3.3) is using the current as an input, it has to be changed. For simplicity the voltage of the motor is chosen as an input. The relation between the voltage and the current of a DC motor can be described as:

$$u = R_\mathrm{a}i + L_\mathrm{a}\dot{i} + k_\mathrm{v}\omega \tag{3.5}$$

where R_a is armature resistance, L_a is the armature inductance, k_v is the motor velocity constant and u and i are the armature voltage and the armature current respectively. Since (3.5) is a differential equation, it would add two further states to the dynamic model. This can be avoided if the dynamics of the current in (3.5) are significantly faster than the fastest dynamic in (3.3). This assumption is satisfied and the dynamics of the current can be neglected. This yields to the following equation.

$$u = R_\mathrm{a}i + k_\mathrm{v}\omega \tag{3.6}$$

For further simplification the uncertainty vector $\boldsymbol{\delta}$ in (3.3) will be neglected. This can be done by the assumption of no slipping wheels, no external disturbance and by neglecting the friction of the castor. These simplifications in addition with (3.6) yield the reduced dynamic model of the robot.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\phi} \\ \dot{v} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} v\cos(\phi) \\ v\sin(\phi) \\ \omega \\ \frac{mbr^2}{\theta_v}\omega^2 - \frac{2k_\mathrm{t}k_\mathrm{v}}{\theta_v R_\mathrm{a}}v \\ -\frac{2mbr^2}{\theta_\omega}v\omega - \frac{d^2k_\mathrm{t}k_\mathrm{v}}{\theta_\omega R_\mathrm{a}}\omega \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{rk_\mathrm{t}}{\theta_v R_\mathrm{a}} & \frac{rk_\mathrm{t}}{\theta_v R_\mathrm{a}} \\ -\frac{rdk_\mathrm{t}}{\theta_\omega R_\mathrm{a}} & \frac{rdk_\mathrm{t}}{\theta_\omega R_\mathrm{a}} \end{bmatrix} \begin{bmatrix} u_\mathrm{L} \\ u_\mathrm{R} \end{bmatrix} \tag{3.7}$$
$$x(0) = x_0, y(0) = y_0, \phi(0) = \phi_0, v(0) = v_0, \omega(0) = \omega_0$$

3.2.3 Model Identification

The goal of the model identification is to determine the unknown parameter in (3.7). Since in (3.7) parameters only occur on the right hand side of $[\dot{v}, \dot{\omega}]^T$ and these equations do not depend on $[x, y, \omega]^T$, the dynamic model for the identification can be reduced to the states $[v, \omega]^T$. Furthermore, the parameters in (3.7) are combined to one parameter vector. This yields to the following model description for the identification:

$$\begin{aligned} \begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix} &= \begin{bmatrix} p_1\omega^2 + p_2 v \\ p_4 v\omega + p_5\omega \end{bmatrix} + \begin{bmatrix} p_3 & p_3 \\ -p_6 & p_6 \end{bmatrix} \begin{bmatrix} u_\mathrm{L} \\ u_\mathrm{R} \end{bmatrix} \\ v(0) &= v_0, \omega(0) = \omega_0 \end{aligned} \tag{3.8}$$

with the parameter vector $\boldsymbol{p} = [p_1, p_2, p_3, p_4, p_5, p_6]^T$. In order to identify the model, the velocity and the angular velocity need to be measured. For this purpose the encoder values of each wheel are stored in the low level of the robot. These encoder values are used offline to calculate the velocity and the angular velocity of the robot. Therefore, the calculation for the position in Section 2.2.2 is used again. By dividing the Δpos and $\Delta angle$ by the sample time of the low level, the velocity and angular velocity results. If the outcomes of 2.2.2 are taken into account, the calculation of the velocity and angular velocity can be assumed as accurate enough. For the identification there are three experiments considered. The input signal is chosen as a step function, whereby each experiment has different final values. These are chosen as typical trajectories of a robot. The first experiment is a straight drive, in the second experiment the robot does a turn and in the last experiment the robot drives a curve. In order to obtain the parameters, the model (3.8) is simulated with a parameter vector $\boldsymbol{p}_0$. The velocity and the angular velocity from the measurements are calculated in parallel. The output of the model $\boldsymbol{y}_\mathrm{sim} = [v_\mathrm{sim}, \omega_\mathrm{sim}]^T$ and the output of the measurements $\boldsymbol{y}_\mathrm{meas} = [v_\mathrm{meas}, \omega_\mathrm{meas}]^T$ are considered in a cost function which is given in (3.9). Thereby$N$ is the quantity of measured points.

$$J(\boldsymbol{p}) = \frac{1}{2}\sum_{i=1}^{N}((v_\mathrm{meas}(t_i) - v_\mathrm{sim}(t_i, \boldsymbol{p}))^2 + (\omega_\mathrm{meas}(t_i) - \omega_\mathrm{sim}(t_i, \boldsymbol{p}))^2) \tag{3.9}$$

To minimize this cost function, a nonlinear optimizer is used. The measurement and the simulation with the identified parameters can be seen in Figure 3.2.

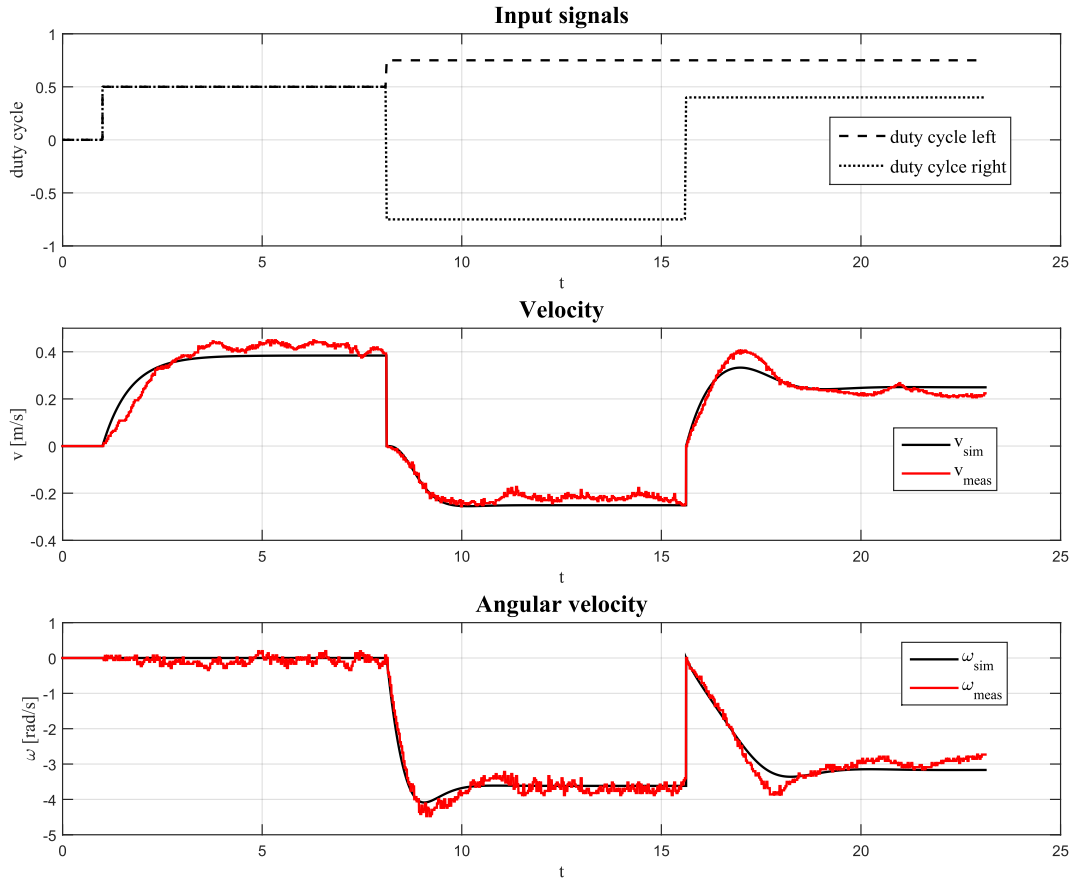

Figure 3.2: Simulation of the model with the identified parameters and the visualization of the measurement.

3.2.4 Speed Controller

The formation controller, which is considered in this work, uses the kinematic model in (3.1) to obtain the control law for each robot. Since the input of our robot is the duty cycle of the PWM signal of each wheel, it is not guaranteed that the current speed of each wheel reaches the desired speed. Therefore, it is necessary to implement a speed controller for each wheel. For this purpose a simple PID controller is considered. The input of the controller is the error between the desired speed and the current speed of each wheel. In order to obtain the desired speed from the velocity and angular velocity of the robot, (3.2) can be rearranged. The transfer function of the continuous-time PID controller is:

$$G_{\mathrm{C}}(s) = K_{\mathrm{P}} + K_{\mathrm{I}}\frac{1}{s} + K_{\mathrm{D}}\frac{Ns}{s+N} \tag{3.10}$$

with the control parameter K_P, K_I and K_D and the filter coefficient N. In order to obtain the parameters for the PID controller, a nonlinear optimization is used. The goal of this optimization is to minimize the integrated squared error e between the desired speed and the current speed. Therefore, the reduced dynamic model (3.7) with the identified parameter is used. Since the input u of the motor is constrained, this constraint has to be considered in the optimization, so a nonlinear constrained optimization algorithm is used. In order to avoid slipping wheels, the acceleration of the robot is constrained as well as the angular acceleration. The optimization problem with the considered constraints is:

$$\begin{aligned} &\min_{K_\mathrm{P},K_\mathrm{I},K_\mathrm{D}} \int_0^T e^2 dt \\ \text{subject to } &|u| - u_\mathrm{max} \leq 0 \\ &|\dot{v}| - a_\mathrm{max} \leq 0 \\ &|\dot{\omega}| - \alpha_\mathrm{max} \leq 0 \end{aligned} \tag{3.11}$$

with the maximum voltage of the motor u_max, the maximum acceleration a_max, the maximum angular acceleration α_max and the upper bound T of the integral. This bound should be chosen large enough until the error becomes zero.

For the implementation of the PID controller, the continuous-time controller has to be transformed into a discrete controller. This is done with the discretization method zero-order hold.

Another characteristic of the given system has to be considered here. This concerns the discrete encoder values. As mentioned in Section 3.2.3, Δpos and $\Delta angle$ from (2.1) are divided by the sample time to achieve the current speed of each wheel (3.2). Since (2.1) uses the encoder values, the difference of the speed from one sample to the next sample could be relatively large, depending on the encoder values. Since the input of the controller is the error e between the desired speed and the current speed, the signal course of the current speed needs to be smooth. In order to reduce the noise of the current speed, a discrete filter is implemented. With this filter a better performance of the PID controller can be achieved.

Chapter 4

Review of Different Formation Controllers

In this section different controllers for achieving a formation are investigated, for the purpose of possible implementation on the given robots. Specifically, several simulations of each formation controller are carried out with different models of the robot. These models are changed in an appropriated manner to achieve a more detailed model of the real robots. According to the results of these simulations, a conclusion is given regarding the possibility of implementing a formation controller on the robots.

4.1 Basic Formation Tracking Controller

The basic formation tracking controller is proposed in [LA06], which is an extension of the tracking controller for one robot in [PLNS98]. This controller takes advantage of the cascade structure. The goal of which is to decompose the tracking control problem into two sub-problems which are simpler and 'independent'. One resulting controller concerns the position and the other the orientation.

4.1.1 Setup

The controller does not use the original coordinates of the robot. Instead it uses so called error coordinates:

$$\begin{bmatrix} x_\mathrm{e} \\ y_\mathrm{e} \\ \phi_\mathrm{e} \end{bmatrix} = \begin{bmatrix} \cos(\phi) & \sin(\phi) & 0 \\ -\sin(\phi) & \cos(\phi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_\mathrm{r} - x \\ y_\mathrm{r} - y \\ \phi_\mathrm{r} - \phi \end{bmatrix} \tag{4.1}$$

which are proposed in [KKMN90], whereby the subscript r indicates the reference and the coordinates without a subscript refer to the current position of the robot. Figure 4.1 illustrates the error coordinates. These error coordinates yield to the error dynamics:

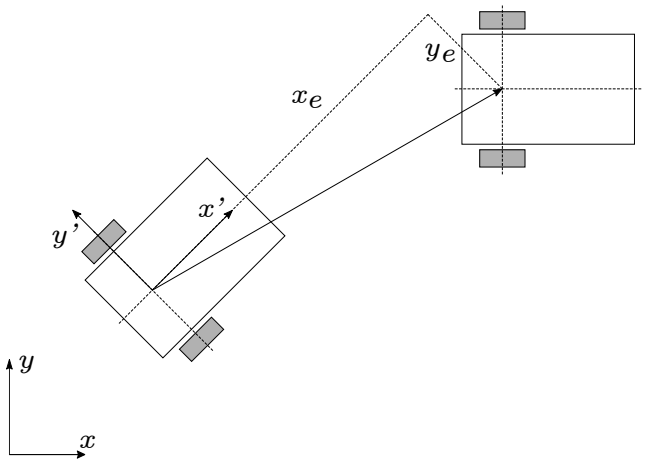

Figure 4.1: Illustration of the error coordinates according to [PLNS98].

$$\begin{aligned} \dot{x}_\mathrm{e} &= \omega y_\mathrm{e} - v + v_\mathrm{r}\cos(\phi_\mathrm{e}) \\ \dot{y}_\mathrm{e} &= -\omega x_\mathrm{e} + v_\mathrm{r}\sin(\phi_\mathrm{e}) \\ \dot{\phi}_\mathrm{e} &= \omega_\mathrm{r} - \omega \\ x_\mathrm{e}(0) = x_{\mathrm{e}0}, y_\mathrm{e}(0) &= y_{\mathrm{e}0}, \phi_\mathrm{e}(0) = \phi_{\mathrm{e}0} \end{aligned} \tag{4.2}$$

according to [KKMN90].

Theorem 1 ([PLNS98]) *Consider the system (4.2) in closed-loop with the controller*

$$\begin{aligned} v &= v_\mathrm{r} + c_2 x_\mathrm{e} \\ \omega &= \omega_\mathrm{r} + c_1 \phi_\mathrm{e} \end{aligned} \tag{4.3}$$

where $c_1 > 0$, $c_2 > 0$. If $\omega_\mathrm{r}(t)$, $\dot{\omega}_\mathrm{r}(t)$, and $v_\mathrm{r}(t)$ are bounded and there exist δ and k such that

$$\int_t^{t+\delta} \omega_\mathrm{r}(\tau)^2 d\tau \geq k, \quad \forall t \geq t_0 \tag{4.4}$$

then the closed-loop system (4.2),(4.3) is globally K-exponentially stable.

In order to extend these results to a formation tracking controller, a formation specification vector $d_{ri} = \left[d_{x_{ri}}, d_{y_{ri}}\right]^T$ is added to the error coordinates [LA06]. This vector describes the distance between the reference trajectory and the follower i in the global coordinate system. The extended error coordinates are given by:

$$\begin{bmatrix} x_{ei} \\ y_{ei} \\ \phi_{ei} \end{bmatrix} = \begin{bmatrix} \cos(\phi_i) & \sin(\phi_i) & 0 \\ -\sin(\phi_i) & \cos(\phi_i) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_\mathrm{r} - x_i - d_{x_{ri}} \\ y_\mathrm{r} - y_i - d_{y_{ri}} \\ \phi_\mathrm{r} - \phi_i \end{bmatrix} \tag{4.5}$$

according to [LA06].

4.1.2 Simulation in Matlab/Simulink

In order to investigate the basic formation tracking controller regarding the ability of the robot, a first simulation with the kinematic model (3.1) is performed. A simple constellation with three robots is considered. The reference trajectory is created by a 'virtual robot' with the inputs $v_\mathrm{r} = 0.1$ and $\omega_\mathrm{r} = 0.05$ and the initial pose $\boldsymbol{q}_{\mathrm{r},0} = [0, -2, 0]^T$. The formation vector for each vehicle is:

$$\boldsymbol{d}_{\mathrm{r}1} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \boldsymbol{d}_{\mathrm{r}2} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} \boldsymbol{d}_{\mathrm{r}3} = \begin{bmatrix} 0.5 \\ -0.5 \end{bmatrix} \tag{4.6}$$

and the initial pose of each robot is chosen as:

$$\boldsymbol{q}_1(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \boldsymbol{q}_2(0) = \begin{bmatrix} -0.5 \\ -0.5 \\ 0 \end{bmatrix} \boldsymbol{q}_3(0) = \begin{bmatrix} -1 \\ -1 \\ 0 \end{bmatrix} \tag{4.7}$$

which is a triangle formation. Considering the controller in (4.3), the information needed for each robot are the error coordinates in (4.5) and the reference inputs v_r and ω_r. In order to calculate the error coordinates, the reference pose is required. This information is necessary for a potential implementation of the basic formation tracking controller on the robots.

A visualization of the results of the simulation can be seen in Figure 4.2. It can be observed that each robot performs the required formation after they converge to the given trajectory. This implies that the error coordinates (4.5) converge to zero.

4.1.3 Applicability

For this controller a simulation with more detailed models of the robot are skipped due to two main reasons. On the one hand the theorem 1 has a condition for the reference angular velocity ω_r as (4.4) suggested. This implies

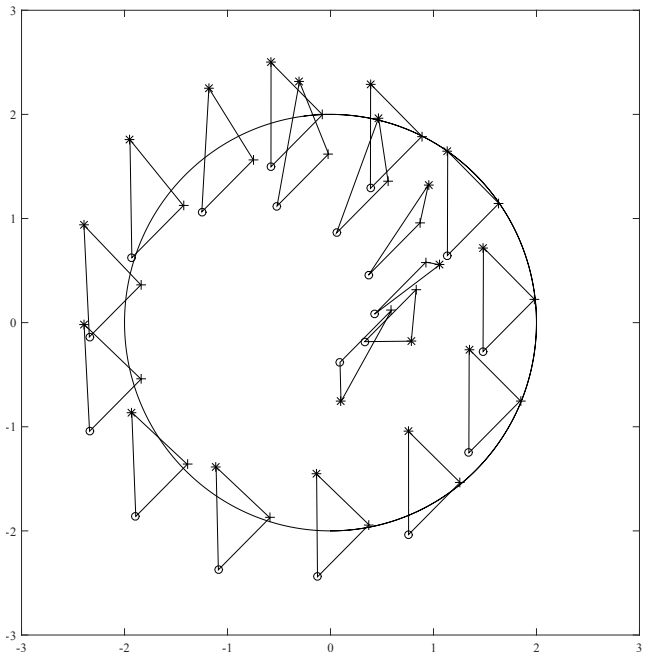

Figure 4.2: Visualization of the basic formation tracking controller. The Robot 1 is denoted as +, Robot 2 as o and Robot 3 is denoted as $*$.

e.g. that a reference trajectory as a straight line with $\omega_r = 0$ is not covered by the theorem. In order to be able to perform as many general trajectories as possible, this controller is not suitable. On the other hand the formation specification vector d_{ri} is defined, as mentioned above, in global coordinates. This can be seen in Figure 4.2 where the desired triangular formation does not change in regards to the global view. For this work a formation needs to be achieved which can be expressed in the coordinates of each robot.

4.2 $l - \psi$ Controller

The $l - \psi$ is proposed in [DOK98]. This controller uses the leader-follower approach, whereby the relative distance and the orientation between the leader and the follower are controlled. This approach uses the techniques of input/output linearization to achieve the formation controller.

4.2.1 Setup

Figure 4.3 illustrates the leader and the follower robot with the controlled values l_{12} and ψ_{12}. The aim of the $l - \psi$ controller is to maintain the desired

length l_{12}^{d} and the desired relative angle ψ_{12}^{d} while the leader is following a given trajectory. Thereby, the state of the follower are given as: $[l_{12}, \psi_{12}, \phi_2]^T$.

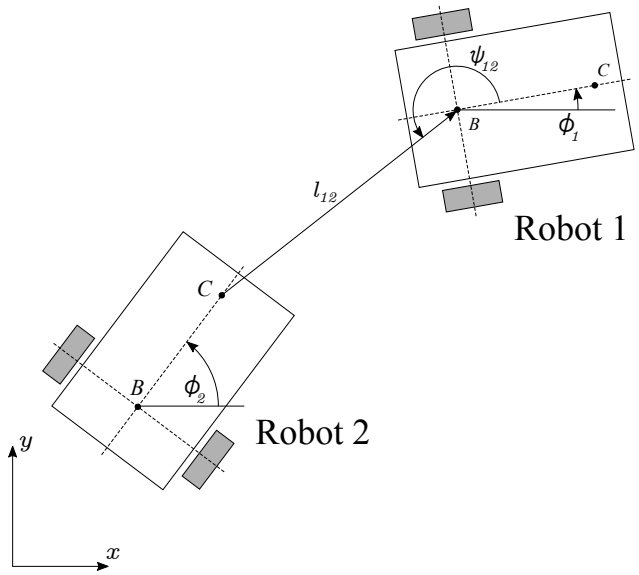

Figure 4.3: Illustration of the leader follower constellation of the $l-\psi$ controller according to [DOK98].

The kinematic equations of Robot 1 shown in Figure 4.3 is given in (3.1). The kinematic equations of Robot 2 are given by:

$$\begin{aligned} \dot{l}_{12} &= v_2 \cos(\gamma_1) - v_1 \cos(\psi_{12}) + c\omega_2 \sin(\gamma_1) \\ \dot{\psi}_{12} &= \frac{1}{l_{12}} \left(v_1 \sin(\psi_{12}) - v_2 \sin(\gamma_1) + c\omega_2 \cos(\gamma_1) - l_{12}\omega_1 \right) \\ \dot{\phi}_2 &= \omega_2 \end{aligned} \tag{4.8}$$

according to [DOK98], with $\gamma_1 = \phi_1 + \psi_{12} - \phi_2$. It is required that $l_{12} > c$ in order to avoid a collision.

The use of input-output linearization yields the following control law for the follower:

$$\begin{aligned} \omega_2 = \frac{\cos(\gamma_1)}{c} \{ & (\alpha_2 l_{12} \left(\psi_{12}^{\mathrm{d}} - \psi_{12} \right) - v_1 \sin(\psi_{12}) + \\ & l_{12}\omega_1 + \rho_{12} \sin(\gamma_1)) \} \\ v_2 = \rho_{12} - c\omega_2 tan(\gamma_1) & \end{aligned} \tag{4.9}$$

where

$$\rho_{12} = \frac{\alpha_1 (l_{12}^{\mathrm{d}} - l_{12}) + v_1 \cos(\psi_{12})}{\cos(\gamma_1)} \tag{4.10}$$

With these inputs the $l - \psi$ variables become:

$$\begin{aligned} \dot{l}_{12} &= \alpha_1(l_{12}^{\mathrm{d}} - l_{12}) \\ \dot{\psi}_{12} &= \alpha_2(\psi_{12}^{\mathrm{d}} - \psi_{12}) \end{aligned} \tag{4.11}$$

whereby (4.9), (4.10) and (4.11) are according to [DOK98]. In order to calculate the control law in (4.9) the length l_{12} and the angle ψ_{12} are required. These can be calculated as follows:

$$\begin{aligned} {}^I\boldsymbol{l}_{12} &= {}^I\boldsymbol{p}_1 - \left({}^I\boldsymbol{p}_2 + {}^I\boldsymbol{R}_{\mathrm{V}_2} \cdot \begin{bmatrix} c \\ 0 \end{bmatrix} \right) \\ l_{12} &= ||{}^I\boldsymbol{l}_{12}|| \\ \psi_{12} &= arccos\left(\frac{\left({}^I\boldsymbol{R}_{\mathrm{V}_1}\boldsymbol{e}\right)^T \cdot (-{}^I\boldsymbol{l}_{12})}{||{}^I\boldsymbol{R}_{\mathrm{V}_1}\boldsymbol{e}|| \cdot ||{}^I\boldsymbol{l}_{12}||} \right) \end{aligned} \tag{4.12}$$

with

$$\begin{aligned} {}^I\boldsymbol{R}_{V_i} &= \begin{bmatrix} \cos(\phi_i) & -\sin(\phi_i) \\ \sin(\phi_i) & \cos(\phi_i) \end{bmatrix}, i = 1, 2 \\ \boldsymbol{e} &= \begin{bmatrix} 1 \\ 0 \end{bmatrix} \end{aligned} \tag{4.13}$$

where ${}^I\boldsymbol{R}_{V_i}$ is a transformation matrix which transforms a vector from the coordinate system V_i into the inertial coordinate system I and ${}^I\boldsymbol{p}_i$ is the position vector of the robot i referred to the inertial coordinate system. This vector contains not the full state of the robot i but only the position (x_i, y_i).

To be able to calculate the control law in the follower robot, the whole state of the leader $\boldsymbol{q}_1$, the whole state of the follower $\boldsymbol{q}_2$ and the input signals of the leader (v_1, ω_1) are needed as well as the angular velocity of the follower ω_2. From this the communication between the leader and the follower can be determined. The information needed to transmit from the follower to the leader are the whole state of the leader and the input signals of the leader.

Theorem 2 ([DOK98]) *Assume the system of two mobile robots shown in Figure 4.3 and the associated control law in (4.9). For the motion of the lead robot following a circular path, $v_1 = K_1$, $\omega_1 = K_2$, ϕ_2 is locally asymptotically convergent to $\phi_1(t) + \psi_{12}^{\mathrm{d}} - \beta_2 - arccos(K_2/\beta_1)$*

where

$$\begin{aligned}\beta_1 &= \sqrt{\left(\frac{K_2 l_{12} + K_1 \sin(\psi_{12}^{\mathrm{d}})}{c}\right)^2 + \left(\frac{K_1 \cos(\psi_{12}^{\mathrm{d}})}{c}\right)^2} \\ \beta_2 &= arctan\left(\frac{K_1 \cos(\psi_{12}^{\mathrm{d}})}{K_2 l_{12} + K_1 \sin(\psi_{12}^{\mathrm{d}})}\right).\end{aligned} \tag{4.14}$$

Theorem 3 ([DOK98]) *For the case that the leader robot follows a straight line ($\omega_1 = K_2 = 0$), ϕ_2 converges exponentially to ϕ_{10}.*

4.2.2 Simulation in Matlab/Simulink

The first simulation of the $l - \psi$ controller in order to investigate the behavior is done with the kinematic model (3.1). Therefore, a constellation with one leader and two followers is considered. The leader drives a curve with the input values $v_{\mathrm{r}} = 0.1$ and $\omega_{\mathrm{r}} = 0.1$. The formation specification for the first follower is defined as follows: $l_{12}^{\mathrm{d}} = 0.3, \psi_{12}^{\mathrm{d}} = 210°$ and for the second follower the desired control values are: $l_{13}^{\mathrm{d}} = 0.3, \psi_{13}^{\mathrm{d}} = 150°$. The initial pose for each robot is:

$$\boldsymbol{q}_1(0) = \begin{bmatrix}0\\0\\0\end{bmatrix} \boldsymbol{q}_2(0) = \begin{bmatrix}-0.5\\-0.6\\0\end{bmatrix} \boldsymbol{q}_3(0) = \begin{bmatrix}-0.5\\-0.3\\0\end{bmatrix}. \tag{4.15}$$

The control parameters are chosen as $\alpha_1 = 0.5$ and $\alpha_2 = 0.5$.

The results of the simulation with the kinematic model can be seen in Figure 4.4. After an initial convergence of the two followers it can be observed that the desired length and the desired angle are maintained. The follower robot follows the leader with the specified parameters. In comparison with the basic formation controller (4.1) the control values of the $l - \psi$ controller are expressed in the local coordinate system of the leader. This fact can be observed in Figure 4.4 where the formation rotates with the leader. This corresponds to the desired behavior of the controller demanded in this work.

Since this controller fulfills all the requirements regarding a kinematic model of the robot, further investigations are carried out. The first further simulation concerns the model of the robot. Instead of using the kinematic model of the robot on which the control design is based, a dynamical model is used. Therefore, the model (3.7) of Section 3.2.2 is used to simulate the behavior of the robot. In order to apply it to the given robots, the identified parameters of Section 3.2.3 are used. Since the velocity v and the angular velocity ω are not controlled in (3.7), the speed controller of Section 3.2.4 is used. Therefore, not merely the identified control parameter for the PID controller are used, but rather other parameters are considered. The expectations

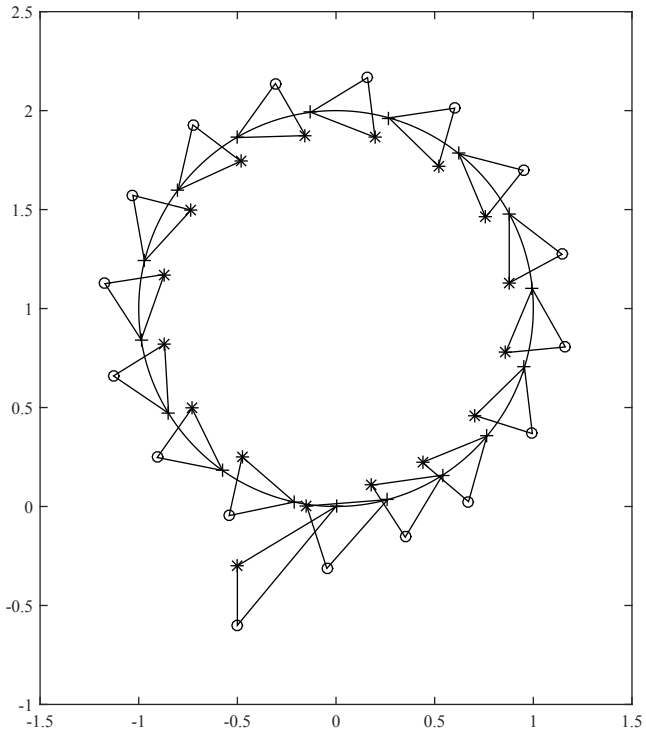

Figure 4.4: Visualization of the $l - \psi$ controller with a kinematic model. The Robot 1 is denoted as $+$, Robot 2 as o and Robot 3 is denoted as $*$.

regarding these changes concern a formulation of requirements on the speed controller. With these predefinitions a more realistic model of the robot is considered.

There are two crucial outcomes of these simulations. On the one hand the speed controller takes an important part in the whole system. If the velocity v and the angular velocity ω of each robot do not converge to the required value, the behavior of the dynamic model is different from the kinematic model. In addition to this the length l_{1i} and the angle ψ_{1i} (where i is the number of the follower) do converge to the desired values if the leader robot converge to the kinematic model and the follower do not converge. In order to achieve this goal the integral part of the PID controller is necessary. Considering the method of identifying the control parameters it is ensured that the velocity and the angular velocity converge. On the other hand the shape of the reference trajectory combined with the parameters of the speed controller have an influence on the convergence of the formation values, the length l_{1i} and the angle ψ_{1i}. If the speed controller is faster, the influence on a change in the reference trajectory regarding the formation values is less than with a slower

speed controller. Therefore, a fast speed controller is required. This demand is covered by the method of identifying the control parameters.

In order to improve the dynamic model, a further characteristic of the robot is included. A crucial aspect from implementing a controller in hardware is the discretization. Due to the characteristic of a microcontroller it is not possible to achieve a continuous output signal of the controller. As opposed to this the control output will be a discrete signal. Hence a few simulations regarding the discrete sampling of the controller are carried out. To be able to gain an insight of the $l - \psi$ applied to a given hardware, simulations with different sampling times are performed. This covers also the characteristic of the high level, as mentioned in 2.1.1, as the sampling time cannot be guaranteed.

For this purpose a constellation with one leader and one follower is considered with the desired length $l_{12}^{\mathrm{d}} = 0.3m$ and the desired angle $\psi_{12}^{\mathrm{d}} = 210°$. The results of two simulation can be seen in Figure 4.5. The upper subplot visualizes the current length and the lower subplot visualizes the current angle, both according to (4.12). In each subplot three simulations are visualized. The first one uses a continuous $l - \psi$ controller whereas the other simulations use a discrete controller, one with a sample frequency of 10 Hz and one with a sample frequency of 0.5 Hz respectively. On the one hand it can be observed, that the controller with the higher sample frequency is close to the continuous one. On the other hand a too low sample frequency leads to a non converging behavior. This characteristic of the discretization has to be considered by the implementation of the control law on the high level.

Due to the fact that this formation controller needs communication, the method of communication is investigated. In particular the impact of the communication is of great interest. For this work two types of error causes are considered. On the one hand the communication via network leads to delays. There are many causes for these delays e.g. the limited network transfer rate or sending packages through several layers. In summary all these delays can be considered as an additional amount of time in the sampling time. Thereby the investigation of the discretization has to be repeated with the new sample time. Nevertheless the result of the previous investigation does not change.

On the other hand, as mentioned in 2.4, the communication protocol UDP is used for transmitting data. This protocol is unreliable and there is no guarantee of delivery. Therefore, the model of the system is extended to cover the occurrence of package loss in the network. The event of a package loss is assumed to be uniformly distributed and if one package is lost, the whole calculation of the controller is skipped in this sample.

There are two outcomes of this investigation regarding the package loss. On the one hand the stability of the $l - \psi$ controller depends on the percentage of the package loss. If the package loss is very high (in this case beyond 80%),

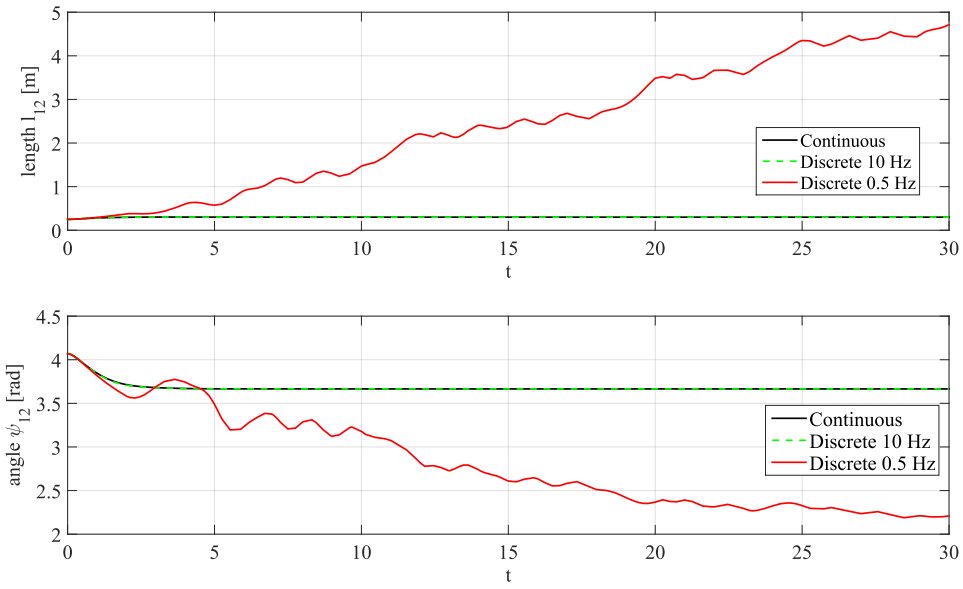

Figure 4.5: Visualization of the $l - \psi$ controller with a dynamic model and a discrete controller.

the stability is not further guaranteed. Nevertheless the system tolerates a huge amount of package loss, which is a positive result regarding the implementation. On the other hand the amount of tolerated package loss depends on the shape of the leader's trajectory. The more changes occur in this trajectory, the less lost packages can be tolerated in order to achieve an adequate performance.

The last improvement of the model for simulating the formation controller concerns the uncertainties in the sensor values. The sensors for calculating the pose of the robot are the encoders, as mentioned in Section 2.2. The critical aspects of the encoders are the discrete values. Considering the position calculation in Section 2.2.2, there is an update of the global position in each sample time of the low level. Combining these things, it is not guaranteed if the robot goes straight, that in each sample time the same amount of encoder ticks are received. Also the difference in the encoder ticks on the right side could differ from the difference in the encoder ticks on the left side. As a first step the difference in the encoder values is considered. Therefore, the simulation model is extended with a subsystem which models the encoders and another subsystem for calculating the position. These subsystems are triggered with the sample time of the low level of the robots. In order to be able to model the differences of the encoder values in each sample time, a uniformly, zero-mean distributed random number generator is added. This takes numbers in a defined range and adds them to the current encoder value.

The results of a simulation concerning the uncertainties in the encoders can be seen in Figure 4.6. The setup for this simulation is the same as in the simulations before, except for the leader's trajectory. For simplicity a straight line is considered. It can be observed, that the current length l_{12} and the current angle ψ_{12} are tending to the desired values and maintaining these values with a certain distance. This characteristic of the encoders can be treated as noise.

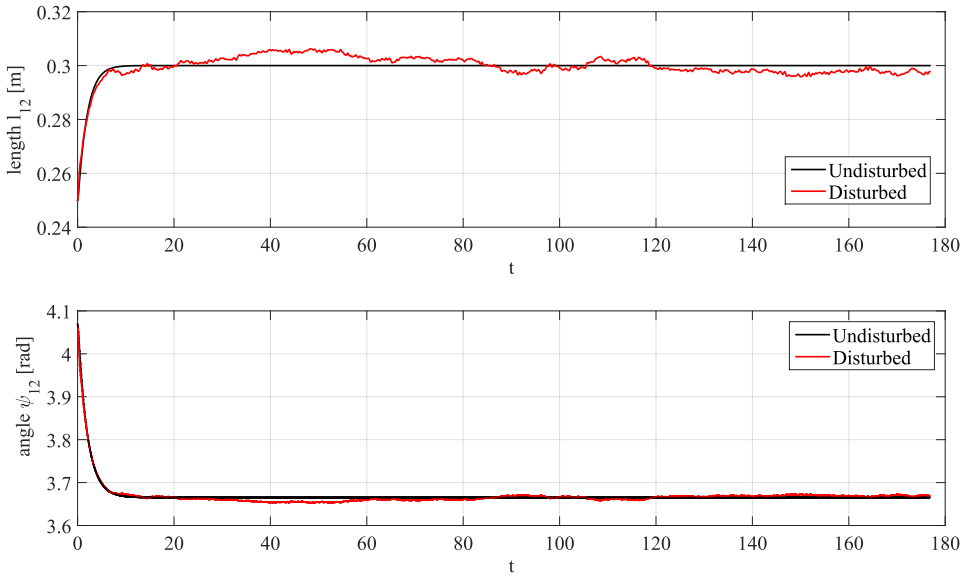

Figure 4.6: Visualization of the $l - \psi$ controller with a dynamic model, a discrete controller and the uncertainties in the encoder values.

As a second step the behavior of the system mentioned in Section 2.2.2 is discussed. Specifically the impact of the missing encoder values per sample time are investigated. If there is an encoder tick missing, the calculation of the global pose is wrong. These have an impact on the formation controller, since the incorrect pose of the robot is considered when calculating the control law. To emphasize this, the model was extended to be able to cover these behaviors. The setup is the same as in the simulation before. In this simulation the uncertainties of the encoder values are neglected and the focus is on the missing encoder values. The results of this simulation can be seen in Figure 4.7. This simulation is based on the following assumptions: The occurrence of a missing encoder value is uniformly distributed with an occurrence rate of 10%, the amount of a missing encoder value is one for the left encoder and two for the right encoder. These results illustrate explicitly the necessity of collecting each encoder value. It is basically not possible to achieve a formation with this setting.

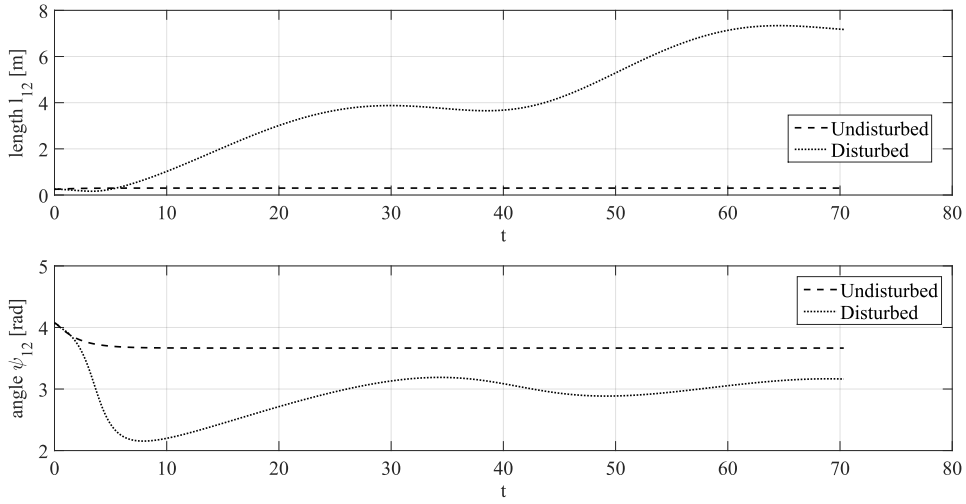

Figure 4.7: Visualization of the $l - \psi$ controller with a dynamic model, a discrete controller and missing encoder values.

4.2.3 Simulation in EyeSim

EyeSim is used as a second environment for simulating the formation controller. This program is a simulation environment for multiple mobile robots which provides the possibility to simulate different kind of robots [Brä08], [KB04]. One of those robots is similar to the robots this thesis takes as a basis. An advantage of EyeSim is the application programmer interface (API) which is the same as the API of the real robot. This means the whole functionality of the high level is provided in EyeSim. The relevant functionality for this purpose is limited to manipulating the speed of the robot, the communication between robots and the retrieving of the sensor data. The use of EyeSim is a good opportunity to check the control law (4.9) in combination with the high level functionality. As a further advantage of testing the $l - \psi$ controller on the EyeSim is the realistic behavior of the PSD sensors. They measure the distance with respect to the mounting position of the PSD sensors on the robot. Furthermore, it is possible to build up walls which are recognized by the PSD sensors. Beyond that the EyeSim simulator detects a collision between two robots or a robot and a wall [Brä08], [KB04].

Since the controller was tested in Matlab/Simulink, the control algorithm (4.9) including (4.10),(4.12),(4.13) has to be translated into C++. In order to guarantee consistency in the control law, the automatic code generation of Matlab/Simulink is used. This code is implemented in EyeSim with the functionality of the high level to supply the control law with all the necessary

data. The generated code is used as a black box to obtain the control values v and ω.

The results of the simulation show a similar behavior of the $l-\psi$ controller to that obtained from the Matlab/Simulink simulations. A convergence of the controlled values l_{12} and ψ_{12} to the desired values can be observed.

4.2.4 Applicability

In this section a summary of all the investigated situations concerning the implementation of the $l-\psi$ controller on the robots is given. The aspects covered from the results before are the speed controller and the uncertainties of the encoder values. If the speed controller is implemented as suggested in Section 3.2.4, the formation can be achieved. Another point mentioned recurrently is the shape of the reference trajectory. This has to be chosen in an appropriate manner.

Critical opened points are the time demand on the robot for one execution of the control algorithm and the package loss in the network. If the suggestions mentioned in the section before are considered, it is possible to implement the $l-\psi$ controller on the given robots. These points will be reconsidered in the section regarding the implementation.

4.3 $l-l$ Controller

The $l-l$ controller is proposed in [DOK98]. This controller also uses the leader-follower approach with the difference of two leaders and one follower. The aim of the $l-l$ controller is to control the distance between the follower and the two leaders.

4.3.1 Setup

Figure 4.8 illustrates the two leaders (Robot 1 and Robot 2) and the follower (Robot 3). The controlled values are the distances between the leaders and the follower l_{13} and l_{23}. The aim of this controller is to maintain the desired lengths l_{13}^{d} and l_{23}^{d} while the leaders perform a given trajectory. Thus, the state of the follower is given as: $[l_{13}, l_{23}, \phi_3]^T$. The requirements for the $l-l$ controller concern the lengths l_{13} and l_{23}. It is required that those lengths are greater than the distance c and the follower must not lie on a line connecting the two leaders [DOK98].

The kinematic equations for the leader robots shown in Figure 4.8 are given in (3.1). The kinematic equations of the follower are given by:

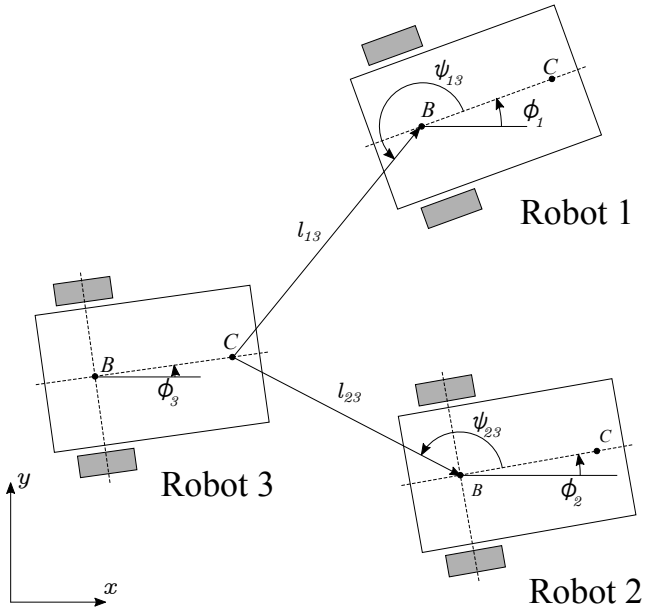

Figure 4.8: Illustration of the leader follower constellation of the $l-l$ controller according to [DOK98].

$$\begin{aligned} \dot{l}_{13} &= v_3 \cos(\gamma_1) - v_1 \cos(\psi_{13}) + c\omega_3 \sin(\gamma_1) \\ \dot{l}_{23} &= v_3 \cos(\gamma_2) - v_2 \cos(\psi_{23}) + c\omega_3 \sin(\gamma_2) \\ \dot{\phi}_3 &= \omega_3 \end{aligned} \tag{4.16}$$

according to [DOK98], with $\gamma_i = \phi_i + \psi_{i3} - \phi_3, i = 1, 2$.

The use of input-output linearization yields the following control law for the follower:

$$\begin{aligned} \omega_3 = \frac{1}{c\sin(\gamma_1 - \gamma_2)} \{ &\alpha_1 (l_{13}^{\mathrm{d}} - l_{13}) \cos(\gamma_2) + v_1 \cos(\psi_{13}) \cos(\gamma_2) \\ &- \alpha_2 (l_{23}^{\mathrm{d}} - l_{23}) \cos(\gamma_1) - v_2 \cos(\psi_{23}) \cos(\gamma_1) \} \end{aligned} \tag{4.17}$$

$$v_3 = \frac{\alpha_1 (l_{13}^{\mathrm{d}} - l_{13}) + v_1 \cos(\psi_{13}) - c\omega_3 \sin(\gamma_1)}{\cos(\gamma_1)}$$

With these inputs the $l - l$ variables become:

$$\begin{aligned} \dot{l}_{13} &= \alpha_1 (l_{13}^{\mathrm{d}} - l_{13}) \\ \dot{l}_{23} &= \alpha_2 (l_{23}^{\mathrm{d}} - l_{23}) \end{aligned} \tag{4.18}$$

whereby (4.17) and (4.18) are according to [DOK98]. The calculation of the current length l_{13}^{d} and l_{23}^{d} follows (4.12).

In order to be able to calculate the control law in the follower robot, the whole state of the two leaders $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$, the whole state of the follower $\boldsymbol{q}_3$ and the velocity of the leaders v_1 and v_2 are required as well as the angular velocity of the follower ω_3. Considering this information, the communication can be determined. The information needed to communicate from both leaders to the follower are the whole state of the leaders and the velocity of the leaders.

Theorem 4 ([DOK98]) *Assume a system of three mobile robots as in Figure 4.8 and the associated control law given by (4.17). For a straight line parallel motion of the first two robots (constant velocity $v_1 = v_2 = K$, $\omega_1 = \omega_2 = 0$ and $\phi_{10} = \phi_{20} = \phi_0$), ϕ_3 locally converges exponentially to $\phi_3^e = \phi_0$ and $\psi_{13}(t) = \psi_{13}(0)$ and $\psi_{23}(t) = \psi_{23}(0)$.*

This theorem only considers a straight line. In comparison with the $l - \psi$ controller, a circular motion is not considered. This shrinks the variety of possible trajectories performed by the leaders.

4.3.2 Simulation in Matlab/Simulink

In order to be able to observe the behavior of the $l - l$ controller, a first simulation with a kinematic model is carried out. Therefore, a simple constellation with two leaders and one follower is considered. Since the Theorem 4 only covers a straight line, the input values of both leaders are chosen as $v_i = 0.1$ and $\omega_i = 0$ $(i = 1, 2)$. The formation specification for the follower is defined as $l_{13}^{\mathrm{d}} = 0.3$ and $l_{23}^{\mathrm{d}} = 0.4$. The initial pose of each robot is:

$$\boldsymbol{q}_1(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \boldsymbol{q}_2(0) = \begin{bmatrix} 0 \\ -0.4 \\ 0 \end{bmatrix} \boldsymbol{q}_3(0) = \begin{bmatrix} -0.5 \\ 0.3 \\ 0 \end{bmatrix} \tag{4.19}$$

and the control parameters are chosen as $\alpha_1 = 1$ and $\alpha_2 = 1$.

The visualization of the results of the simulation can be seen in 4.9. After an initial phase of convergence the formation is maintained with the defined formation specification. This kind of behavior could not be achieved with the $l - \psi$ controller, since the $l - \psi$ controller only allows one leader for each follower. Combining the $l-\psi$ controller and the $l-l$ controller, new formations are achievable. This is the reason for further investigations of this controller regarding the implementation on the robots.

The following procedure refers to the investigation of the $l - \psi$ controller in Section 4.2, since the same considerations take place. The difference is less

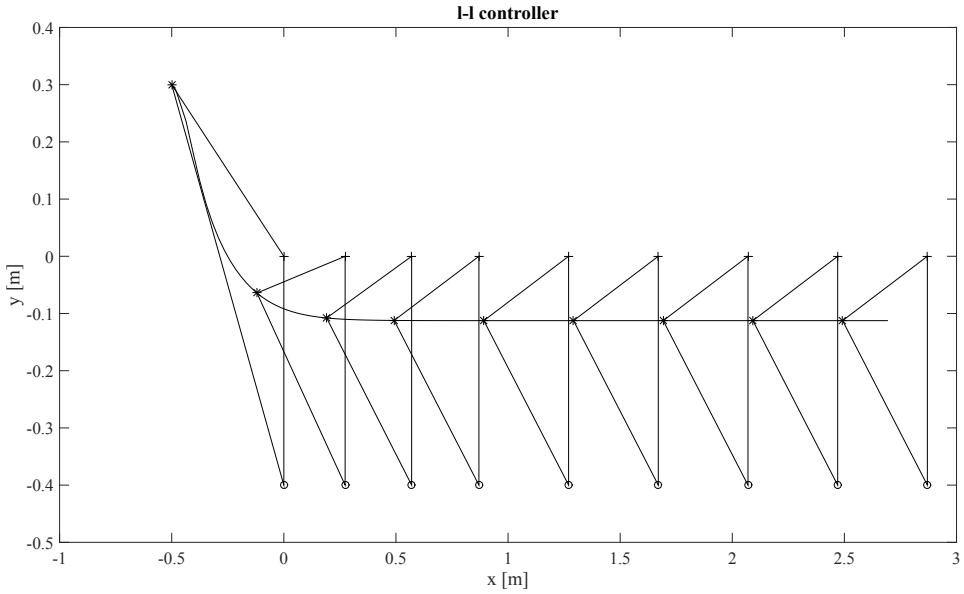

Figure 4.9: Visualization of the $l - l$ controller with a kinematic model. The Robot 1 is denoted as $+$, Robot 2 as o and Robot 3 is denoted as $*$.

emphasis on the necessity of improving the model for the simulation. Thus, only the results of the simulations are considered. In the further simulations the setup considering the initial pose, the desired values and the leader's trajectory will be the same as in the simulation of the kinematic model.

The first simulation beyond the kinematic model is to simulate the $l - l$ controller with a dynamic model. The setup considering the speed controller will be the same as in the investigation of the $l - \psi$ controller. The crucial outcome of these simulations concerns the speed controller. The requirements for the speed controller correspond with the demands in the Section 4.2.2. It must have a converging behavior considering the velocity v and the angular velocity ω. Since there is no change in the leaders' reference trajectory, the demands regarding the transient time are not that strict.

The second further investigation concerns the discrete sampling of the controller. Therefore, simulations with different sample times are performed. In Figure 4.10 the results of a simulation with a sample frequency of 10 Hz and 1 Hz can be seen. For comparison reasons a simulation with a continuous $l - l$ controller is also visualized. Considering the higher sampled controller, it can be observed that the current length is close to the current length of the continuous controller. In fact both are converging to the desired length. As opposed to this, the $l - l$ controller with a sample frequency of 1 Hz shows an unstable behavior. Comparing the behavior of a discrete $l - \psi$ controller and a discrete $l - l$ controller, it can be observed, that the $l - l$ controller tending earlier to

an unstable behavior as the $l - \psi$ controller, when the sampling frequency is decreased. These properties of the $l - l$ controller needs to be reconsidered in the implementation of the controller.

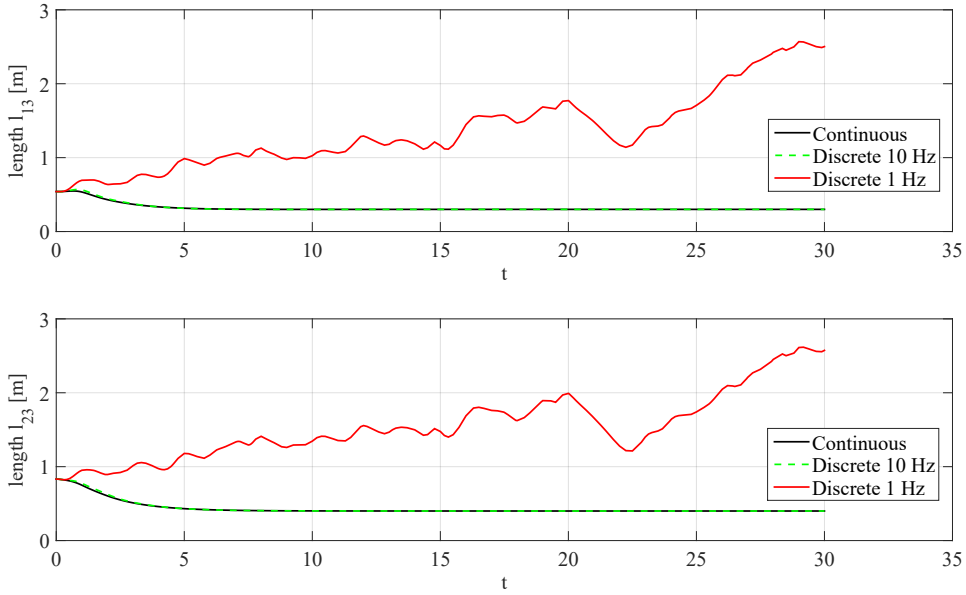

Figure 4.10: Visualization of the $l - l$ controller with a dynamic model and a discrete controller.

The delay in the network communication is considered as an additional discrete delay as in Section 4.2.2. Therefore, the next investigation considers the package loss in the network. Several simulations with different percentage of package loss are performed. The crucial outcome concerns the tolerated amount of package loss. The more packages are lost, the longer it takes to converge to the desired formation. This is valid until a certain amount of package loss. Beyond this point the $l - l$ controller is unstable.

The last consideration regarding the $l - l$ controller concerns the encoder values. As mentioned in the Section 4.2.2, the encoder values can differ in each sample of the low level. The results of this investigation are similar to those in the investigation of the $l - \psi$ controller. The current length l_{13} and l_{23} are tending to the desired values and maintaining these values with a certain range.

4.3.3 Simulation in EyeSim

As discussed in Section 4.2.3 during the investigation of the $l - \psi$ controller, a second simulation environment is applied. The $l - l$ controller is treated similarly to the approach with the $l - \psi$ controller. In order to observe the behavior

of the formation controller in EyeSim, the code generator in Matlab/Simulink is used. Provided with all the functionality in the high level regarding the necessary inputs of the control law in (4.17), the simulation in EyeSim can be started. Analysing the results of these simulations, a converging behavior of the control values l_{13} and l_{23} can be observed.

4.3.4 Applicability

The results discussed above are reconsidered in order to summarize the investigation of the $l - l$ controller. In comparison with the $l - \psi$ controller, the demands on the speed controller are less restrictive. However the requirements on the sample time are higher. Even so this requirement is not a disqualifier, since the sample frequency is very low when the instabilities occur. Nevertheless the execution time of the control algorithm including all other necessary executions has to be considered during the implementation. Furthermore, the occurrence of a package loss has to be reconsidered for the implementation. This is covered with the investigation of the $l - \psi$ controller.

4.4 Collision Avoidance Controller

In this section an extension of the $l-\psi$ controller from Section 4.2 is proposed. The goal of this extension concerns the collision avoidance based on local sensors. On the one hand static obstacles should be recognized by the local PSD sensors. According to this measurement, a defined safety distance to this static obstacle should be maintained. On the other hand a collision with other robots in the formation needs to be avoided.

4.4.1 Setup

For the collision avoidance controller a formation with one leader and N follower is considered. This formation is controlled by a $l - \psi$ controller. In order to be able to avoid a collision, the desired control values l_{1i}^{d} and ψ_{1i}^{d} $(i = 1, 2, ..., N)$ are changed in an appropriated manner. This implies that the desired values are time dependent. Therefore, the following algorithm for changing the desired values and hence the formation is proposed:

$$\begin{aligned} \dot{l}_{1i}^{\mathrm{d}} &= k_1 u_1 \\ \dot{\psi}_{1i}^{\mathrm{d}} &= k_2 u_2 \\ l_{1i}^{\mathrm{d}}(0) = l_{1i0}^{\mathrm{d}}, \psi_{1i}^{\mathrm{d}}(0) &= \psi_{1i0}^{\mathrm{d}} \end{aligned} \tag{4.20}$$

with

$$
u_1 = \begin{cases} 1, & \text{if}(PSD_{\mathrm{L}} < \epsilon_{\mathrm{L}} \quad \text{and} \quad PSD_{\mathrm{R}} < \epsilon_{\mathrm{R}}) \quad \text{or} \\ & (PSD_{\mathrm{F}} < \epsilon_{\mathrm{F}}) \quad \text{or} \quad (PSD_{\mathrm{L}} > \epsilon_{\mathrm{L}} \quad \text{and} \\ & PSD_{\mathrm{R}} > \epsilon_{\mathrm{R}} \quad \text{and} \quad PSD_{\mathrm{F}} > \epsilon_{\mathrm{F}} \quad \text{and} \quad l_{1i}^{\mathrm{d}} < l_{1i0}^{\mathrm{d}}) \\ -1, & \text{if}(PSD_{\mathrm{L}} > \epsilon_{\mathrm{L}} \quad \text{or} \quad PSD_{\mathrm{R}} > \epsilon_{\mathrm{R}}) \quad \text{and} \\ & PSD_{\mathrm{F}} > \epsilon_{\mathrm{F}} \quad \text{and} \quad l_{1i}^{\mathrm{d}} > l_{1i0}^{\mathrm{d}} \\ 0, & \text{else} \end{cases}
$$

$$
u_2 = \begin{cases} 1, & \text{if}(PSD_{\mathrm{L}} < \epsilon_{\mathrm{L}} \quad \text{and} \quad PSD_{\mathrm{R}} > \epsilon_{\mathrm{R}}) \quad \text{or} \\ & (PSD_{\mathrm{L}} > \epsilon_{\mathrm{L}} \quad \text{and} \quad PSD_{\mathrm{R}} > \epsilon_{\mathrm{R}} \quad \text{and} \\ & \psi_{1i}^{\mathrm{d}} < \psi_{1i0}^{\mathrm{d}}) \\ -1, & \text{if}(PSD_{\mathrm{L}} > \epsilon_{\mathrm{L}} \quad \text{and} \quad PSD_{\mathrm{R}} < \epsilon_{\mathrm{R}}) \quad \text{or} \\ & (PSD_{\mathrm{L}} > \epsilon_{\mathrm{L}} \quad \text{and} \quad PSD_{\mathrm{R}} > \epsilon_{\mathrm{R}} \quad \text{and} \\ & \psi_{1i}^{\mathrm{d}} > l_{1i0}^{\mathrm{d}}) \\ 0, & \text{else} \end{cases}
\tag{4.21}
$$

where subscript L, R and F refer to the left side, the right side and the front respectively. The symbol PSD denotes the value of the PSD sensor and ϵ specifies the threshold. The definition regarding the values of the PSD sensors is as follows: The value of the measurement is related to the distance, i.e. if the value is beyond a threshold, no threat concerning a collision exists.

This algorithm works in the following way: Four different situations are considered. The fist two situations concern a change in the desired angle ψ_{1i}^{d}. If the distance to an obstacle on the left is below a threshold, the desired angle is increased. Where as if the PSD_{R} is below a threshold, ψ_{1i}^{d} is decreased. Both situations assume that all other distances are beyond the specified threshold. The third situation occurs if both values of the PSD sensors of the left side and on the right side are below a threshold and PSD_{F} is beyond a threshold. If so, the desired length l_{1i}^{d} is increased. The last case concerns the distance to an obstacle in front of the robot, measured by PSD_{F}. If this value is below the defined threshold, the desired length is increased. All these cases have one thing in common. If the hazard from an obstacle is gone, the desired values l_{1i}^{d} and ψ_{1i}^{d} are forced to reach the initial values l_{1i0}^{d} and ψ_{1i0}^{d}.

According to (4.21) the initial values l_{1i0}^{d} and ψ_{1i0}^{d} are only reached, if the the current length and the current angle is exactly the same as the initial ones. For practical purposes a hysteresis is applied to this controller in order to avoid consistently changing in the desired values.

4.4.2 Simulation in EyeSim

The simulation in Matlab/Simulink is skipped for the collision avoidance controller for two reasons. On the one hand this controller has the $l - \psi$ controller as a basis and has already investigated with an extended model in Matlab/Simulink (see Section 4.2.2). On the other hand the collision avoidance controller uses PSD sensors. EyeSim provides a realistic behavior of the PSD sensors. This also includes the measurement of the distance to another robot considering the appearance of it. For these reasons only a simulation in EyeSim is performed.

The first simulation concerns the static obstacle. Therefore, a constellation with one leader and two followers is considered. The formation specification is defined as: $l_{12}^{\mathrm{d}} = 0.4, \psi_{12}^{\mathrm{d}} = 150°$ for the first follower and $l_{13}^{\mathrm{d}} = 0.4, \psi_{13}^{\mathrm{d}} = 210°$ for the second follower. The initial pose is given as:

$$\boldsymbol{q}_1(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \boldsymbol{q}_2(0) = \begin{bmatrix} -0.5 \\ 0.4 \\ 0 \end{bmatrix} \boldsymbol{q}_3(0) = \begin{bmatrix} -0.5 \\ 0 \\ 0 \end{bmatrix}. \tag{4.22}$$

The simulation setup is chosen as follows: The leader performs a straight line with two followers. After a defined distance, an obstacle occurs for a certain length. This obstacle forces the second follower to change the position. The results of this simulation can be seen in Figure 4.11. It visualizes the position of each robot to certain time steps. It can be observed, that the second follower is changing the position in order to maintain a desired minimum distance to the obstacle. The change appears firstly in the angle until the distance to the first follower is under a threshold. After that the desired length is increased. If the formation passed the obstacle, it returns to the initial formation. This simulation does not show an avoidance of a collision, since the obstacle is not a potential collision. But it demonstrates the possibility to maintain a specified distance around each robot based on the PSD sensors.

In Figure 4.12 a screen shot from EyeSim can be seen. It shows the formation after passing the obstacle. The path visualization shows the trajectory of each robot and thus the changing of the position of the second follower. This simulation has a different setup as a basis as the simulations in Figure 4.11 and 4.13.

The corresponding current length l_{13} and current angle ψ_{13} of the second follower are visualized in Figure 4.13. Additional to this, the desired formation parameters l_{13}^{d} and ψ_{13}^{d} are plotted. The situation according to the change in the desired formation parameters can be observed as described above. Furthermore, a converging of the current length l_{13} and current angle ψ_{13} to the desired ones can be seen.

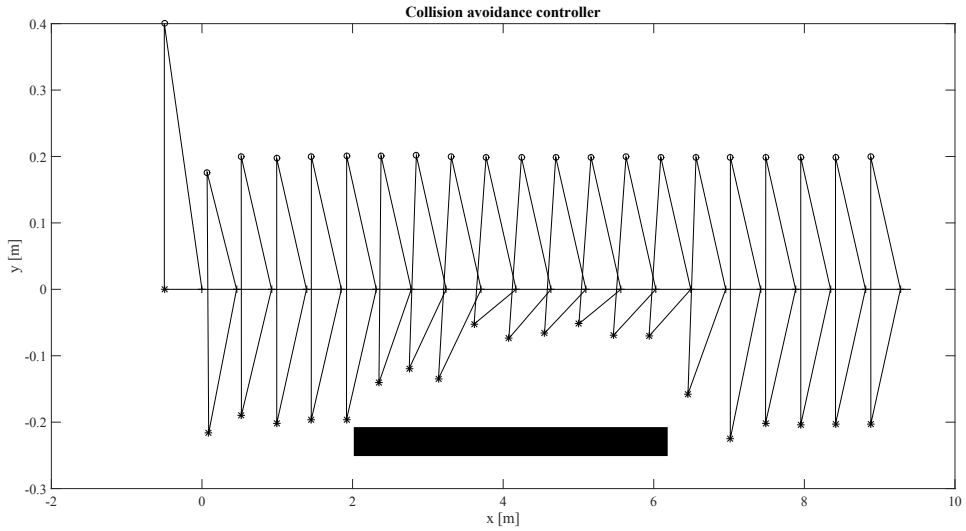

Figure 4.11: Visualization of the collision avoidance controller with an obstacle on the right. The Robot 1 is denoted as +, Robot 2 as o and Robot 3 is denoted as ∗.

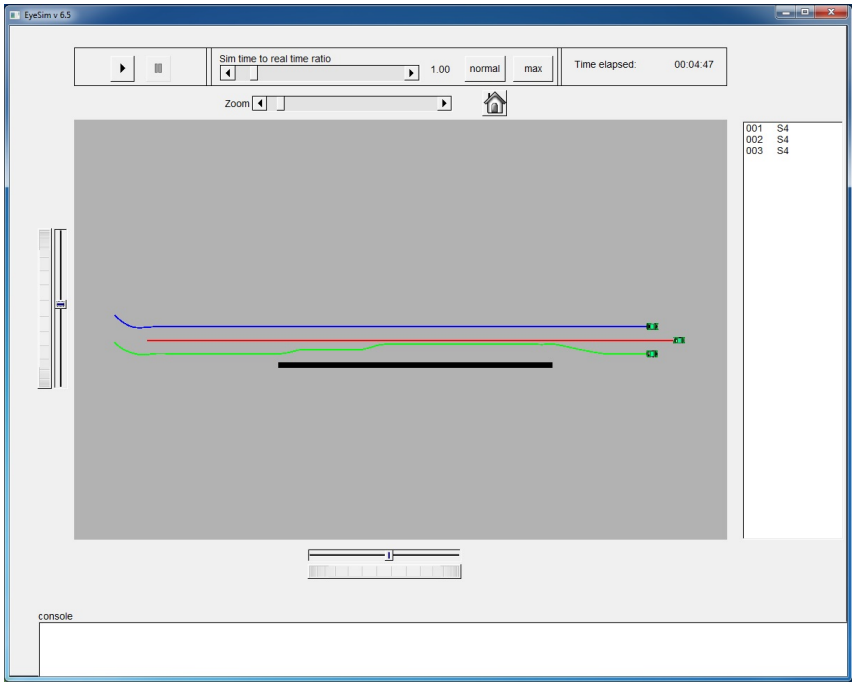

Figure 4.12: Screen shot from EyeSim after simulating the collision avoidance controller with an obstacle on the right.

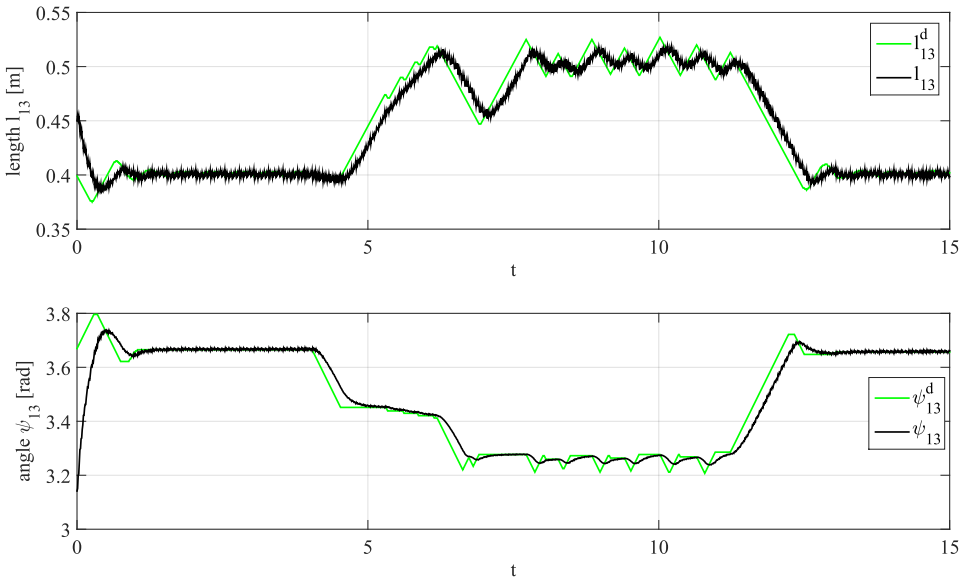

Figure 4.13: Visualization of the formation specification of the collision avoidance controller with an obstacle on the right.

A further simulation considers the collision avoidance with two robots. Therefore, a formation constellation with one leader and two followers is considered. In difference to the simulation before, this simulation uses the same formation parameters for each robot. They are defined as $l^{\mathrm{d}}_{1i} = 0.5$ and $\psi^{\mathrm{d}}_{1i} = 180°$ $(i = 1, 2)$ with the initial posture:

$$\boldsymbol{q}_1(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \boldsymbol{q}_2(0) = \begin{bmatrix} -0.5 \\ 0 \\ 0 \end{bmatrix} \boldsymbol{q}_3(0) = \begin{bmatrix} -0.8 \\ 0 \\ 0 \end{bmatrix}. \tag{4.23}$$

The leader is performing a straight line and the follower robots trying to follow this line at the same position. Since this is physically not possible, the collision avoidance controller is taken into account. The results of this simulation can be seen in Figure 4.14. Once the second follower gets closer to the first follower, the desired length is increased. This is done in a way, that no collision occurs. This new formation is maintained while the second follower has a new desired length.

4.4.3 Applicability

The simulations show the possibility of maintaining a defined distance on each side of the robot and the ability of avoiding a collision. If the crucial parts

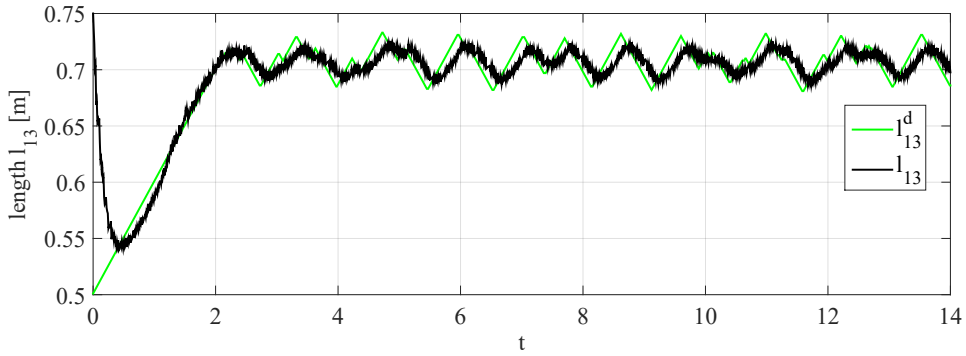

Figure 4.14: Visualization of the current length and the desired length of the follower two with the collision avoidance controller.

in Section 4.2 are considered for the $l - \psi$ controller, an implementation with these specifications is possible.

Chapter 5

Implementation and Results

5.1 Implementation on Real Mobile Robot

In this section the aspects that have to be considered during the implementation of the formation controller are emphasized. In particular, the open points from Section 4 are reconsidered. Furthermore, practical points regarding the operationality are discussed. Following, the implementation of the $l - \psi$ and the $l - l$ controller are considered. If the results concern a specific controller, it will be emphasized. Otherwise it is a general result and thus valid for both formation controllers.

The first step is to implement the control algorithm with the high level function calls to provide all necessary information for the controller. Therefore, the intermediate stage with EyeSim is an advantage. Since the API for the high level function calls is the same and the control algorithm has not to be changed, the code can be directly implemented on the robots.

Once the control algorithm is implemented, the open points can be reconsidered. The first concern is the execution time of the control algorithm. This will be distinguished between two setups. On the one hand only the execution time of the control algorithm itself is measured. Since this control algorithm requires more data from the low level e.g. the position and information from the leader, the sample time will be greater than the execution time of the control algorithm. Therefore, the time from one execution of the control algorithm to the next execution is measured in order to obtain the sample time.

As mentioned in 2.1.1, the high level is operated by an operating system, so it cannot be guaranteed, that the execution time is the same in each sample. Therefore, the measured execution time of the control algorithm is averaged. The results for the $l - \psi$ and the $l - l$ controller can be seen in Table 5.1. Both sample times are below the time of 100 ms which correspond to the sample frequency of 10 Hz. A simulation with this sample frequency has been carried

Table 5.1: Execution Time of Control Algorithm and Sample Time

	Execution Time Control Algorithm	Sample Time
$l-\psi$ controller	48 μs	48 ms
$l-l$ controller	49 μs	56 ms

out for both formation controllers in Section 4. Thus, the sample frequency is considered fast enough and the implementation concerning this point is possible.

The second open point concerns the package loss in the communication network. Several experiments are performed with different amounts of robots in the network. In this constellation no lost package could be observed. Furthermore, the formation controller is very robust against package loss as discussed in Section 4.2.2. Thus, the network communication is considered as robust enough.

Another point that has to be considered is the buffer of the receiver. This buffer is working in a first-in first-out principle, which leads to a constraint in the sender-receiver constellation. In order to guarantee that the receiver processes the currently sent packages from the sender, the sender must not send the packages faster than the receiver can process it. Therefore, the sender has to limit the frequency of sending packages to the receiver.

For practical purposes a new specification file is introduced. In this file all the parameters regarding the formation controller are specified. This includes the formation parameters of the $l-\psi$ and $l-l$ controller as well as the initial position. Furthermore, it is specified which role the robot occupies in the formation, either a leader or a follower. And if it is a follower, it is specified which formation controller the robot uses. With this definition, only one source file is needed for the three different formation controllers. Therefore, a change in each of these parameters is easy to execute.

5.2 Results of the Implementation

In this section the results of the implementation are discussed. This is basically done for each controller itself. Since the $l-\psi$ controller was the first implemented controller, these studies are more extended and in some point more general than the others.

5.2.1 $l - \psi$ Controller

After a successful implementation of the $l - \psi$ controller in Section 5.1, a first experiment is performed. In order to gain a first impression of the connected real system, a simple experiment with one leader and one follower is considered, whereby the leader performs a straight line. The formation specification is defined as $l_{12}^{\mathrm{d}} = 0.3$ and $\psi_{12}^{\mathrm{d}} = 180°$ with the initial posture:

$$\boldsymbol{q}_1(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \boldsymbol{q}_2(0) = \begin{bmatrix} -0.45 \\ 0 \\ 0 \end{bmatrix}. \tag{5.1}$$

The results can be seen in Figure 5.1. It can be observed that the current length is converging to the desired length after a settling time. The fact of an increasing length at the beginning of the experiment is caused by the static friction of the robot. Since the leader has a step function in the velocity v, the robot can overcome the static friction faster. On the other hand the desired velocity of the follower, resulting from the formation controller, is relatively low. Thus, it takes longer to overcome the static friction. For the angle ψ_{12} this behavior cannot be observed, since the initial pose of the leader and the follower already fulfill the desired angle. Nevertheless it can be seen, that the desired angle ψ_{12}^{d} is maintained with a certain range during this experiment.

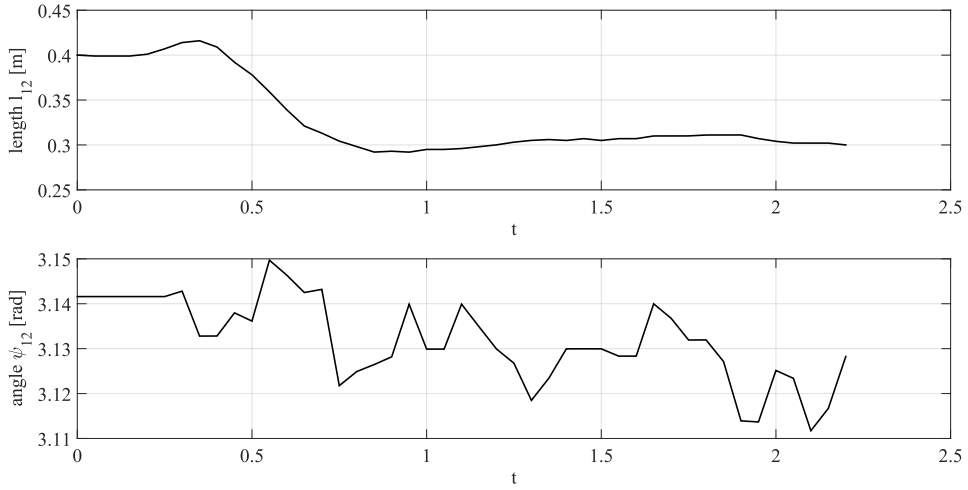

Figure 5.1: Visualization of the results of the first experiment with the $l - \psi$ controller.

The second experiment considers one leader and two followers with the following formation specification: For the first follower the desired length is $l_{12}^{\mathrm{d}} = 0.3$ and the desired angle is defined as $\psi_{12}^{\mathrm{d}} = 150°$ whereas the formation

specification for the second follower is chosen as $l_{13}^{\mathrm{d}} = 0.3, \psi_{13}^{\mathrm{d}} = 210°$. The initial pose of each robot is defined as:

$$\boldsymbol{q}_1(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \boldsymbol{q}_2(0) = \begin{bmatrix} -0.45 \\ 0.35 \\ 0 \end{bmatrix} \boldsymbol{q}_3(0) = \begin{bmatrix} -0.45 \\ 0 \\ 0 \end{bmatrix}. \tag{5.2}$$

The leader performs a given trajectory. This trajectory is defined as a straight line for one meter with a constant velocity. After the straight line, the leader performs a circle with a radius of 0.25 m and an angle of 180°. The current length l_{12} and l_{13} of both follower as well as the current angle ψ_{12} and ψ_{13} are visualized in Figure 5.2. It can be observed, that the current length and the current angle are converging to the desired ones. At the time t = 2.3 seconds the current length of both follower increases until 2.5 seconds. This is the point when the leader changes from the straight line to the circle. Thus, the followers need to converge to the desired length again. The difference in the current angle is not as significant as the difference in the current length.

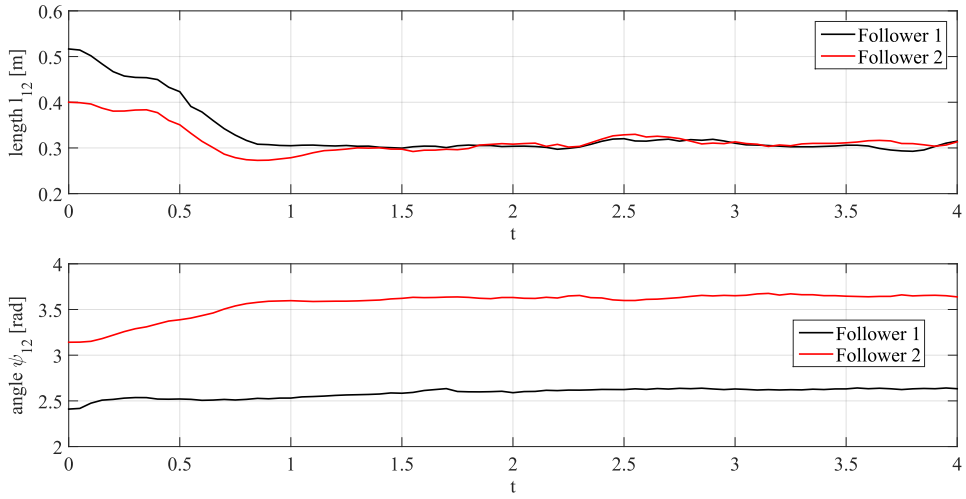

Figure 5.2: Visualization of the results of the second experiment with the $l-\psi$ controller.

Figure 5.3 visualizes the formation of the three robots. For reasons of clarity and comprehensibility this figure only shows the straight line. The convergence behavior as described above is clearly visible. Furthermore, the maintaining of the formation while performing a straight line is recognizable.

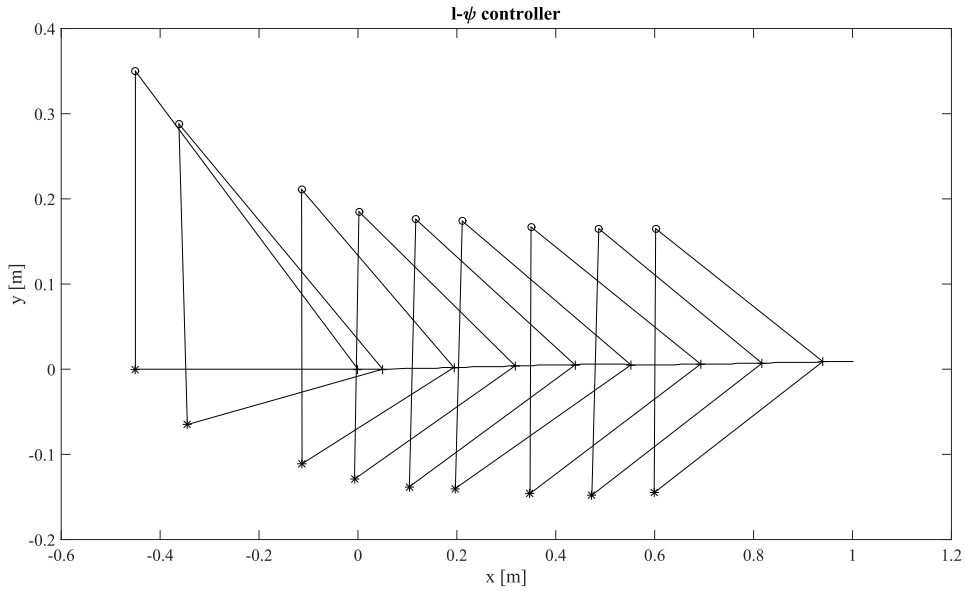

Figure 5.3: Visualization of the results of the second experiment with the $l-\psi$ controller.

5.2.2 $l-l$ Controller

Since the experiments with the $l-\psi$ controller are successfully carried out, a first experiment is performed with the $l-l$ controller. Therefore, a constellation with two leaders and one follower is considered. As mentioned in Section 4.3, the Theorem 4 only covers a straight line. Therefore, both leaders perform a straight line with the input values $v_i = 0.1$ and $\omega_i = 0$ $(i = 1, 2)$. The formation specification for the follower is chosen as $l_{13}^{\mathrm{d}} = 0.4$ and $l_{23}^{\mathrm{d}} = 0.4$ with the initial pose of each robot:

$$\boldsymbol{q}_1(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \boldsymbol{q}_2(0) = \begin{bmatrix} 0 \\ 0.4 \\ 0 \end{bmatrix} \boldsymbol{q}_3(0) = \begin{bmatrix} -0.45 \\ 0.15 \\ 0 \end{bmatrix}. \tag{5.3}$$

The visualization of the results can be seen in Figure 5.4. It can be observed that the current length l_{13} and l_{23} are converging to the desired length l_{13}^{d} and l_{23}^{d} after a settling time. These desired lengths are maintained during the whole experiment, as required.

In Figure 5.5 a visualization of the formation can be seen. It can be observed, that both leaders perform a straight line while the follower maintains the desired lengths and thus the formation. The initial convergence to the specified formation as described above, can be seen as well.

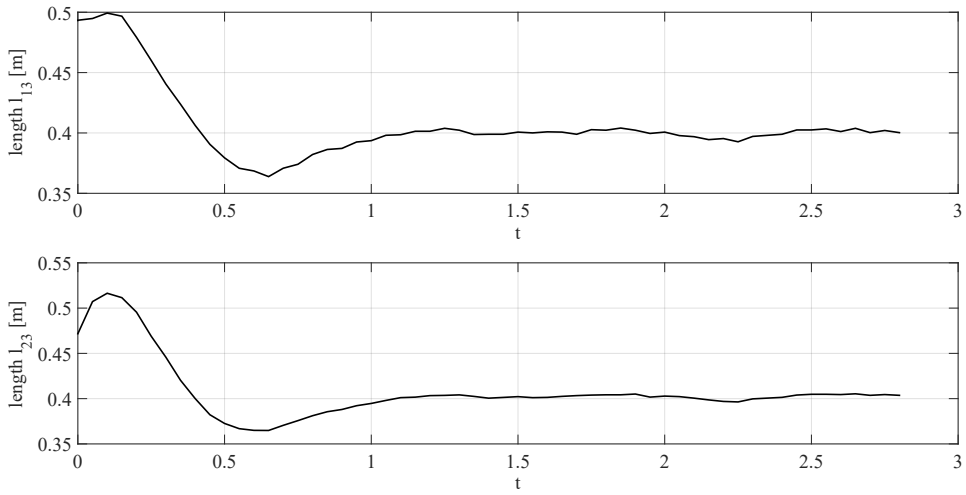

Figure 5.4: Visualization of the results of the experiments on the real robots with the $l - l$ controller.

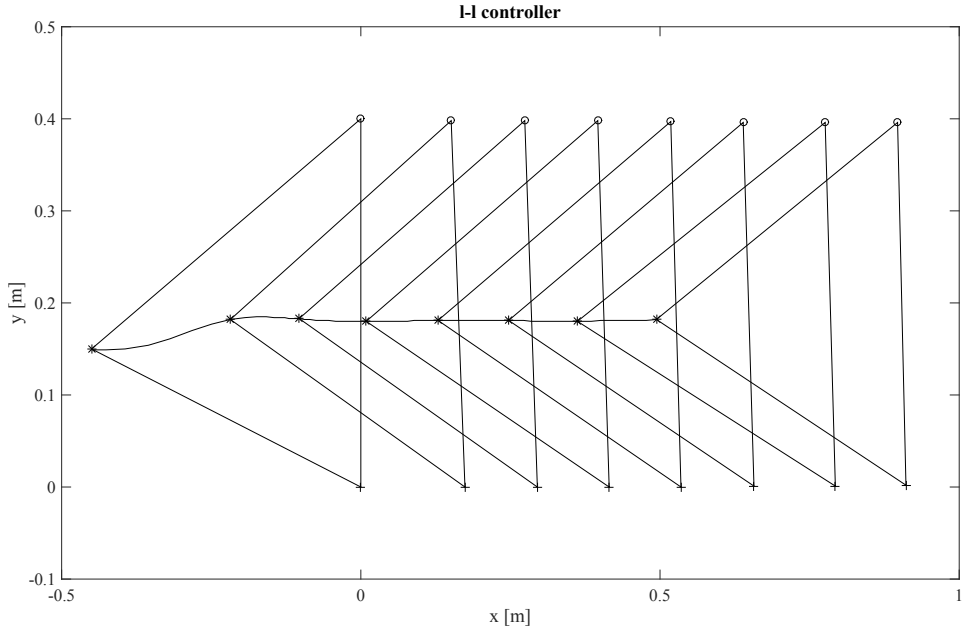

Figure 5.5: Visualization of the $l - l$ controller on the real robots. The Robot 1 is denoted as $+$, Robot 2 as o and Robot 3 is denoted as $*$.

5.2.3 Collision Avoidance Controller

The setup of the experiments which are performed for the collision avoidance controller are similar to those in Section 4.4.2. The first experiment considers a static wall on the right side of the robots. Therefore, a constellation with one

leader and one follower is considered. The formation specification is defined as $l_{12}^{\mathrm{d}} = 0.3$ and $\psi_{13}^{\mathrm{d}} = 180°$ with the initial pose:

$$\boldsymbol{q}_1(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \boldsymbol{q}_2(0) = \begin{bmatrix} -0.43 \\ 0 \\ 0 \end{bmatrix}. \tag{5.4}$$

The visualization of the current length and the desired length as well as the current angle and the desired angle can be seen in Figure 5.6. Typical for the collision avoidance controller is the change in the desired length and desired angle. It can be observed, that the current length and the current angle are close to the desired values with a bit of a time delay. These results are expected from the simulation with EyeSim in Section 4.4.2. The occurrence of the static obstacle on the right side can be seen at time t = 0.5 seconds. At that time a significant change in the desired angle can be observed. The angle is decreasing until t = 0.7 seconds. Subsequently, the robot overcomes the obstacle and the desired angle increases until the current angle reaches the initial angle ψ_{120}^{d}.

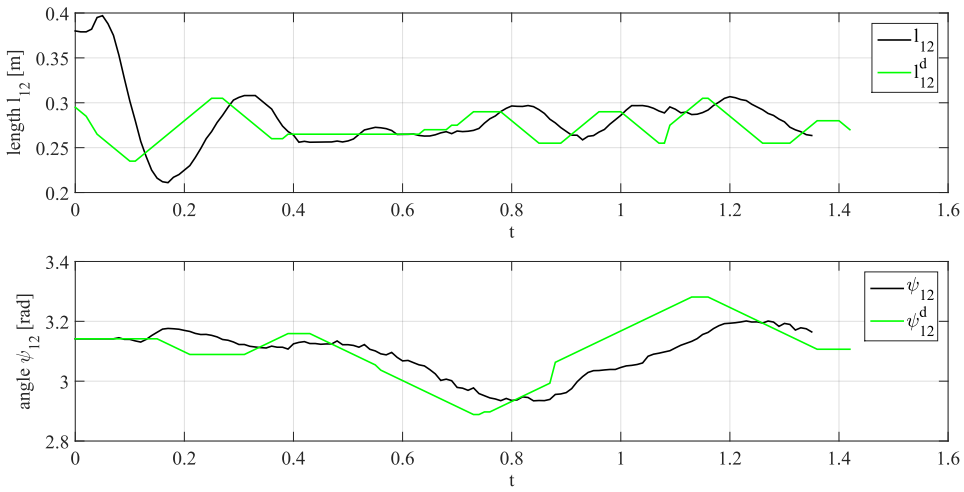

Figure 5.6: Visualization of the collision avoidance controller on the real robots with an obstacle on the right.

In Figure 5.7 a visualization of the formation can be seen. It can be observed, that the follower is changing its position when the obstacles occur. Thus, a specified minimum distance to the obstacle is maintained. After the follower passes the obstacle, the desired angle changes back to the initial one. In Figure 5.8 a photo of the real experiment can be seen. It shows the formation at a point, where the follower increases the distance to the obstacle.

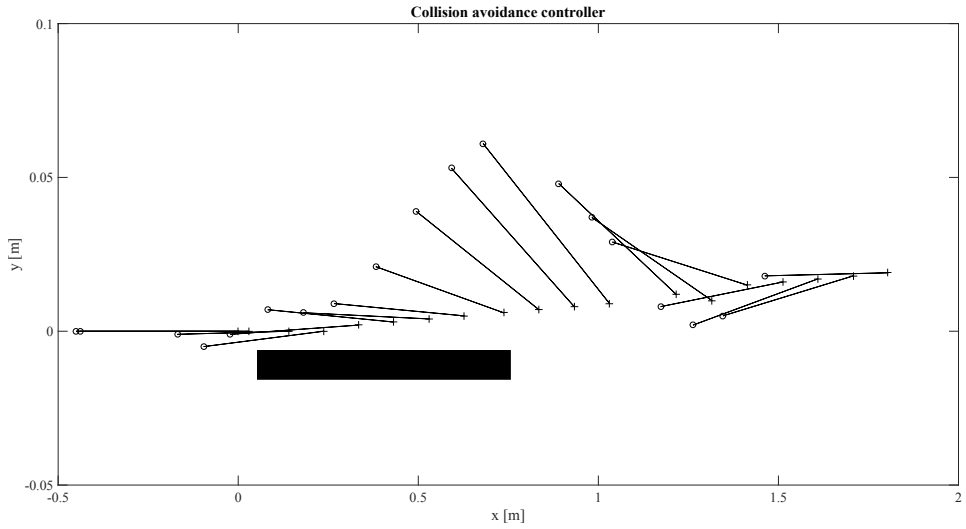

Figure 5.7: Visualization of the collision avoidance controller on the real robots with an obstacle on the right. The Robot 1 is denoted as + and Robot 2 is denoted as o.

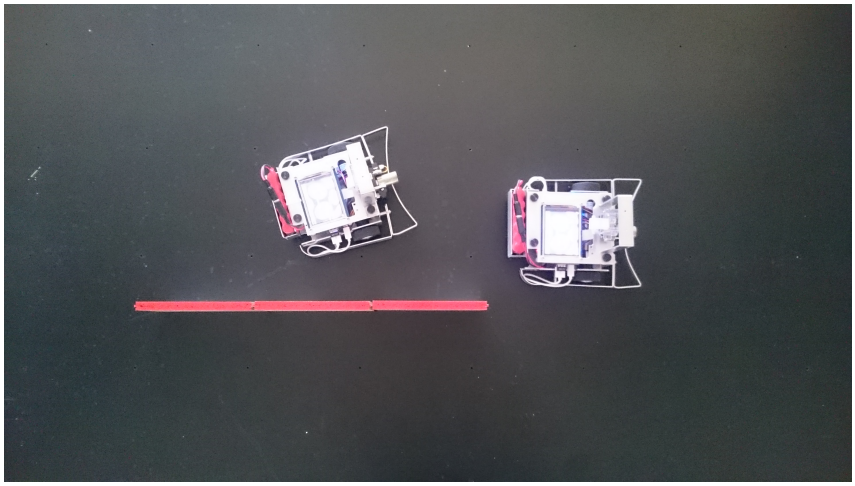

Figure 5.8: Photo of the experiment with the collision avoidance controller with an obstacle on the right

The second experiment concerns the avoidance of a collision with a robot in front. This one is similar to the experiment in Section 4.4.2, with one leader and two followers. These two followers are trying to achieve the same formation, which is defined as: $l_{1i}^{\mathrm{d}} = 0.3$ and $\psi_{1i}^{\mathrm{d}} = 180°$ $(i = 1, 2)$ with the initial pose:

$$\boldsymbol{q}_1(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \boldsymbol{q}_2(0) = \begin{bmatrix} -0.45 \\ 0 \\ 0 \end{bmatrix} \boldsymbol{q}_3(0) = \begin{bmatrix} -0.9 \\ 0 \\ 0 \end{bmatrix}. \tag{5.5}$$

The collision avoidance controller considers the distance to another object, measured by the PSD sensors. For practical purposes another safety feature is added. If the measured distance is below a critical distance, which is lower than the thresholds defined in Section 4.4.1, the robot is forced to stop. This method is necessary at the beginning of the experiment due to the initial position of the second follower. Therefore, the speed at the beginning is too high to stop the robot with the collision avoidance controller. The results of this experiment can be seen in Figure 5.9. Due to the fact of another robot in front of the follower two, the desired length l_{13}^{d} is increased. After the safety distance, which is given by the threshold TH_{F}, is maintained, the robot achives a new desired length. With this desired length, no collision occurs. The stopping of the robot can be observed at the beginning of the experiment as a decreasing and increasing of the current length.

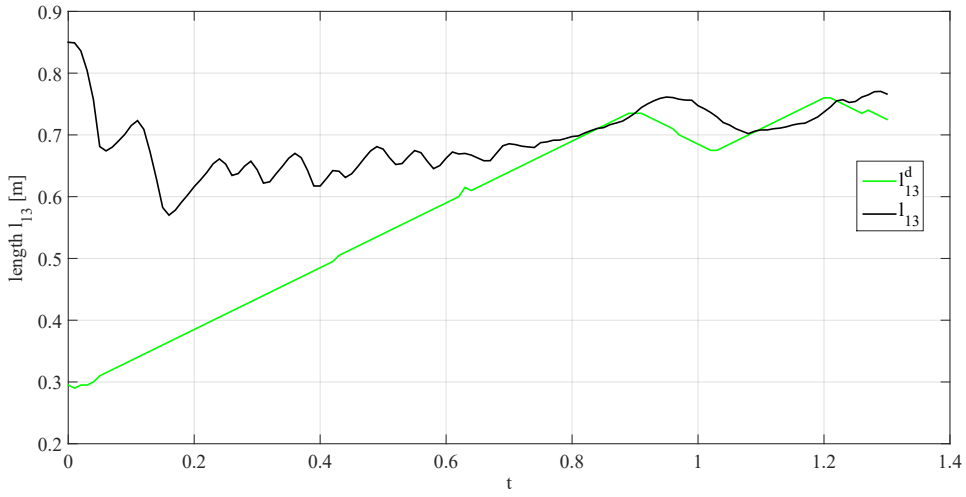

Figure 5.9: Visualization of the current length and the desired length of the follower two with the collision avoidance controller on the real robots.

Chapter 6

Conclusion

This thesis reviews formation controllers for the practical implementation on given robots. An existing parametric dynamic model is used as a basic model for the investigation of different formation controllers. A model identification is performed to obtain the parameters of the dynamic model. Based on this model the parameters of the PID controller for the speed of each wheel are determined. The review of the formation controllers show that the $l-\psi$ controller and the $l-l$ controller are the most suitable approaches for implementing based on the requirements. Studying the results of the simulations, certain requirements concerning the implementation can be imposed. Considering all these demands the implementation is performed successfully. The results of the experiment show a similar behavior to the simulations and thus validates the simulations. The successful implementation and experiments show the ability of applying a formation controller on real robots which was developed for a kinematic model of a mobile robot. In addition to this, it shows that the given mobile robots are able to perform a formation with these formation controllers. Furthermore, an obstacle avoidance controller based on the $l-\psi$ controller is proposed. Therefore, simulations with two different situations are carried out and similar experiments are performed. These results show the ability of the collision avoidance controller in maintaining a safe distance to an obstacle and avoiding collision.

The simulation models developed in this work can be used for further investigations of formation controllers for the application on real robots. With these models it is possible to verify the ability of implementing a formation controller on real robots. Critical parts of the implementation can be determined and appropriate improvements can be carried out. A further improvement involves the dynamic model of the mobile robot. This dynamic model can be extended to consider the friction of the castor. Therefore, a suitable friction model should be chosen. In addition to that, new formation control

approaches can be considered, such as a model predictive control approach. Since this controller requires high computational power, investigations should occur to assure sufficient amounts of computational power prior usage. The last improvement includes the collision avoidance controller. The ability of maintaining a desired distance to a static obstacle laterally can be extended to avoid obstacles in front of a robot. Therefore, the robots ability to sense obstacles in front has to be considered.

List of Figures

List of Tables

Bibliography

[AAC09] G. Antonelli, F. Arrichiello, and S. Chiaverini. *Experiments of Formation Control With Multirobot Systems Using the Null-Space-Based Behavioral Control*. Control Systems Technology, IEEE Transactions on, 17(5):1173–1182, 2009.

[BA98] T. Balch and R. C. Arkin. *Behavior-based formation control for multirobot teams*. Robotics and Automation, IEEE Transactions on, 14(6):926–939, 1998.

[Ben91] J. G. Bender. *An overview of systems studies of automated highway systems*. Vehicular Technology, IEEE Transactions on, 40(1):82–99, 1991.

[Brä08] Thomas Bräunl. *Embedded Robotics: Mobile Robot Design and Applications with Embedded Systems*. Springer Publishing Company, Incorporated, 3rd ed. edition, 2008.

[CC12] In-Sung Choi and Jong-Suk Choi. *Leader-Follower Formation Control Using PID Controller*. In Proceedings of the 5th International Conference on Intelligent Robotics and Applications - Volume Part II, ICIRA'12, pages 625–634, Berlin, Heidelberg, 2012. Springer-Verlag.

[CSVS03] N. Cowan, O. Shakerina, R. Vidal, and S. Sastry. *Vision-based follow-the-leader*. In Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on, volume 2, pages 1796–1801, 2003.

[CY04] Lei Cheng and Wang Yongji. *Communication-based multiple mobile robots rigid formation control*. In Control, Automation, Robotics and Vision Conference, 2004. ICARCV 2004 8th, volume 1, pages 729–734, 2004.

[De 93] De Luca, Alessandro, Di Benedetto, Maria Domenica. *Control of nonholonomic systems via dynamic compensation.* Kybernetika, 29(6):593–608, 1993.

[dLOS98] A. de Luca, G. Oriolo, and C. Samson. *Feedback control of a nonholonomic car-like robot.* In J.-P Laumond, editor, Robot Motion Planning and Control, volume 229 of *Lecture Notes in Control and Information Sciences*, pages 171–253. Springer Berlin Heidelberg, 1998.

[dNCB95] B. d'Andréa Novel, G. Campion, and G. Bastin. *Control of Nonholonomic Wheeled Mobile Robots by State Feedback Linearization.* Int. J. Rob. Res., 14(6):543–559, 1995.

[DOK98] J. P. Desai, J. Ostrowski, and V. Kumar. *Controlling formations of multiple mobile robots.* In Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on, volume 4, pages 2864–2869, 1998.

[FDKO01] R. Fierro, A. K. Das, V. Kumar, and J. P. Ostrowski. *Hybrid control of formations of robots.* In Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on, volume 1, pages 157–162, 2001.

[GSM08] J. Ghommam, M. Saad, and F. Mnif. *Formation path following control of unicycle-type mobile robots.* In Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on, pages 1966–1972, 2008.

[HF10] Yang Huizhen and Zhang Fumin. *Geometric formation control for autonomous underwater vehicles.* In Robotics and Automation (ICRA), 2010 IEEE International Conference on, pages 4288–4293, 2010.

[HYC+08] Lim Heonyoung, Kang Yeonsik, Kim Changwhan, Kim Jongwon, and Jae You Bum. *Nonlinear Model Predictive Controller Design with Obstacle Avoidance for a Mobile Robot.* In Mechtronic and Embedded Systems and Applications, 2008. MESA 2008. IEEE/ASME International Conference on, pages 494–499, 2008.

[KB04] Andreas Koestler and Thomas Bräunl. *Mobile Robot Simulation with Realistic Error Models.* In International Conference on Autonomous Robots and Agents, ICARA 2004, pages 46–51 (6), 2004.

[KKMN90] Y. Kanayama, Y. Kimura, F. Miyazaki, and T. Noguchi. *A stable tracking control method for an autonomous mobile robot.* In Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on, pages 384–389, 1990.

[KY15] M. A. Kamel and Zhang Youmin. *Decentralized leader-follower formation control with obstacle avoidance of multiple unicycle mobile robots.* In Electrical and Computer Engineering (CCECE), 2015 IEEE 28th Canadian Conference on, pages 406–411, 2015.

[LA06] Fang Lei and P. J. Antsaklis. *Decentralized Formation Tracking of Multi-vehicle Systems with Nonlinear Dynamics.* In Control and Automation, 2006. MED '06. 14th Mediterranean Conference on, pages 1–6, 2006.

[LBY03] J.R.T. Lawton, R. W. Beard, and B. J. Young. *A decentralized approach to formation maneuvers.* Robotics and Automation, IEEE Transactions on, 19(6):933–941, 2003.

[LJ12] Yang Liu and Yingmin Jia. *An iterative learning approach to formation control of multi-agent systems.* Systems & Control Letters, 61(1):148–154, 2012.

[LT97] M.Anthony Lewis and Kar-Han Tan. *High Precision Formation Control of Mobile Robots Using Virtual Structures: Autonomous Robots.* Autonomous Robots, 4(4):387–403, 1997.

[OdLV02] G. Oriolo, A. de Luca, and M. Vendittelli. *WMR control via dynamic feedback linearization: design, implementation, and experimental validation.* Control Systems Technology, IEEE Transactions on, 10(6):835–852, 2002.

[PLNS98] Elena Panteley, Erjen Lefeber, Henk Nijmeijer, and Petersbourg St. Russia. *Exponential tracking control of a mobile car using a cascaded approach.* In In Proceedings of the IFAC Workshop on Motion Control, pages 221–226, 1998.

[RI96] H. Raza and P. Ioannou. *Vehicle following control design for automated highway systems.* Control Systems, IEEE, 16(6):43–60, 1996.

[SA91] C. Samson and K. Ait Abderrahim. *Feedback control of a nonholonomic wheeled cart in Cartesian space.* In Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on, pages 1136–1141, 1991.

[SB00] D. J. Stilwell and B. E. Bishop. *Platoons of underwater vehicles.* Control Systems, IEEE, 20(6):45–52, 2000.

[Ser03] A. Serrani. *Robust coordinated control of satellite formations subject to gravity perturbations.* In American Control Conference, 2003. Proceedings of the 2003, volume 1, pages 302–307, 2003.

[SHP04] D. P. Scharf, F. Y. Hadaegh, and S. R. Ploen. *A survey of spacecraft formation flying guidance and control. Part II: control.* In American Control Conference, 2004. Proceedings of the 2004, volume 4, pages 2976–2985, 2004.

[TKCC01] Chung Lee Ti, Tai Song Kai, Hung Lee Ching, and Cheng Teng Ching. *Tracking control of unicycle-modeled mobile robots using a saturation feedback controller.* Control Systems Technology, IEEE Transactions on, 9(2):305–318, 2001.

[TNO04] H. Takahashi, H. Nishi, and K. Ohnishi. *Autonomous decentralized control for formation of multiple mobile robots considering ability of robot.* Industrial Electronics, IEEE Transactions on, 51(6):1272–1279, 2004.

[TO03] H. Takahashi and K. Ohnishi. *Autonomous decentralized control for formation of multiple mobile-robots considering ability of robot.* In Industrial Electronics Society, 2003. IECON '03. The 29th Annual Conference of the IEEE, volume 3, pages 2041–2046, 2003.

[YB96] Hiroaki Yamaguchi and Gerardo Beni. *Distributed Autonomous Formation Control of Mobile Robot Groups by Swarm-Based Pattern Generation.* In Hajime Asama, Toshio Fukuda, Tamio Arai, and Isao Endo, editors, Distributed Autonomous Robotic Systems 2, pages 141–155. Springer Japan, 1996.

[YB98] H. Yamaguchi and J. W. Burdick. *Time-varying feedback control for nonholonomic mobile robots forming group formations.* In Decision and Control, 1998. Proceedings of the 37th IEEE Conference on, volume 4, pages 4156–4163, 1998.

[YDCV98] Zhang Yulin, Hong Daehie, J. H. Chung, and S. A. Velinsky. *Dynamic model based robust tracking control of a differentially steered wheeled mobile robot.* In American Control Conference, 1998. Proceedings of the 1998, volume 2, pages 850–855, 1998.

[YZ05] Quan Chen Yang and Wang Zhongmin. *Formation control: a review and a new consideration*. In Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on, pages 3181–3186, 2005.

[ZFM05] Lin Zhiyun, B. Francis, and M. Maggiore. *Necessary and sufficient graphical conditions for formation control of unicycles*. Automatic Control, IEEE Transactions on, 50(1):121–127, 2005.

[ZN99] Ping Jiang Zhong and H. Nijmeijer. *A recursive technique for tracking control of nonholonomic systems in chained form*. Automatic Control, IEEE Transactions on, 44(2):265–279, 1999.