Unstable behavior of a unicycle mobile robot with tracking control

Unstable behavior of a unicycle mobile robot with tracking control

E.Wijn s0508988

Report No. DCT 2004.63

Bachelors Thesis

Coach: Dr. Ir. P.Lambrechts
Supervisor: Prof. dr. H. Nijmeijer

EINDHOVEN UNIVERSITY OF TECHNOLOGY
DEPARTMENT OF MECHANICAL ENGINEERING
DYNAMICS AND CONTROL TECHNOLOGY GROUP

Abstract

This report considers the unstable behavior which occurs when an orientation-error observer tracking controller combined with a real-time operator-controlled reference trajectory is implemented on an experimental unicycle-type mobile robot. With help of simulations and experiments it is attempted to determine the cause for this behavior. Furthermore, an alternative combined tracking/stabilizing controller, which resets the unicycle system in case of instability, is implemented. In this combined controller the stabilizing mode will function as a back-up when the tracking mode becomes unstable. Effects of certain system configurations on instability and results of the combined controller are presented.

Contents

Chapter 1

Introduction

1.1 Introduction

In recent years a lot of research has been carried out on the control problem for autonomic mobile robotic systems. This is mainly due to an ever-growing application area for these systems, both in industrial and service environments. Some typical application areas for these kinds of systems are for instance order-pick robots in automated warehouses, post delivery robots in office buildings or deep-sea exploration robots. A lot of different types of robotic systems are used. In rough terrain, walking robots are usually preferred. On smoother surfaces wheeled vehicles have the advantage that they are faster and more agile. And systems like unmanned aerial vehicles or unmanned submarines need to manoeuver in 3 dimensions. In this report wheeled mobile robots are considered. The non-holonomic constraint arising from the no-slip condition provides an extra challenge to the control problem. The simplest mobile robot that incorporates non-holonomic constraints is the unicycle-type robot.

The global tracking problem for a unicycle-type mobile robot is already studied by a number of people. Panteley et al. [1] proposes a state feedback controller. This controller is adapted to an output-feedback trajectory tracking controller by Jakubiak et al.(see [4]), under the assumption that one of the tracking error coordinates, a position coordinate or the orientation, is unknown. The unknown error coordinate is reconstructed in an error-observer. For details on stability of this controller the reader is referred to [4]. To further examine and implement these control laws, an experimental setup with a unicycle-type robot is created in the *Dynamics and Control Technology laboratory* at the *Eindhoven University of Technology*. The small two-wheel differential drive mobile robot, called *BellyBot*, is setup on a table and position measuring equipment is installed.

An alternative observer that solves certain implementation issues and is based on the combination of an orientation-error based observer and a non-linear state feedback-tracking controller, was derived and implemented on the experimental BellyBot setup by Noijen (see [6]). Further several control strategies for trajectory tracking and posture stabilization were reviewed and compared on simulation and experimental level (see [5]). These simulation and experiments brought to light that the current orientation-error observer tracking controller shows unstable behavior when large reference signals are fed into the unicycle system. For predefined trajectories this need not be a problem, since the trajectory can be constructed in such a way that the system remains in its stable region. When the reference trajectory is real-time and operator-defined however, the possibility exists that the user-input leads to an unstable system, which of course is not desirable.

1.2 Problem Statement

Tracking control for a unicycle mobile robot generates unstable behavior when a real-time operator-defined reference trajectory is used. In order to prevent such behavior an analysis has to show what causes instability in the system. This leads to the following problem statement:

Analyse the unstable behavior occurring while implementing the tracking-control problem for a unicycle mobile robotic system, and improve its behavior.

The analysis will consist of the modeling of a unicycle-type mobile robot, an overview of the current tracking controller and a systematic limitation of the possible causes for unstable behavior. A solution for the instability problem is found in a combined tracking/stabilizing controller.

1.3 Overview

Chapter 2 will give a short description of the kinematic model of a unicycle-robot. The fundamentals of the current orientation-error observer tracking controller wil be discussed together with the problems that occur when a real-time operator-defined reference trajectory is used. In chapter 3, a study is carried out in order to find the source for the unstable behavior that occurs when large inputs are fed into the system. An alternative solution to the stability problem in the form of a combined tracking and stabilizing controller, will be presented in chapter 4. All this will be validated using simulations and experiments on the BellyBot setup, of which a short description will also be given. Finally some conclusions and recommendations are given in the chapter 5.

Chapter 2

Tracking Control for a Unicycle Mobile Robot

In many practical applications, mobile robots need to follow a trajectory to accomplish their tasks. To follow a trajectory in the best way possible, tracking controllers that keep the robot on a reference trajectory are designed. Such a controller continuously monitors the position of the robot and gives input corrections if needed. In this report, a trajectory is set out in a two dimensional cartesian space and the robot in question is a unicycle mobile robot. Several tracking control laws that apply to this configuration have already been derived. Here we look at the special case that only two of the three coordinates of the robot are known. The two position coordinates are measured, but the orientation is unknown and will be reconstructed by an orientation-observer.

A solution to this problem is given in Noijen (see [6]). This solution, based on an error observer-controller combination, will be briefly discussed in section 2.1. This controller, together with some improvements, is already implemented in MatLab/Simulink (see [5]). Some of the improvements and some practical points about the implementation are discussed in section 2.2. The controller as presented in the report [5] shows good behavior when relatively slow and simple reference-trajectories are used, for instance a circle and eight-figure. However, when a more complicated real-time user-defined reference-trajectory is generated, the controller becomes unstable. In section 2.3 a more detailed description of this problem is given.

2.1 Orientation-Error Observer Tracking Controller

2.1.1 Kinematics

A representation of a model for a unicycle mobile robot is given in figure 2.1. Where x and y denote the position coordinates of the center of the axis connecting the rear wheels. And θ denotes the orientation of the robot with respect to the x-axis. The kinematic model is as follows

$$\begin{cases} \dot{x} &= v\cos(\theta) \\ \dot{y} &= v\sin(\theta) \\ \dot{\theta} &= \omega. \end{cases} \tag{2.1}$$

Where the linear velocity v and the angular velocity ω are the available control inputs.

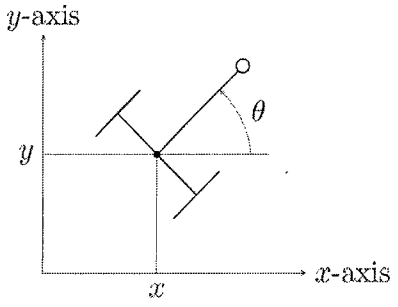

Figure 2.1: The definition of coordinates x, y and θ (figure taken from [5])

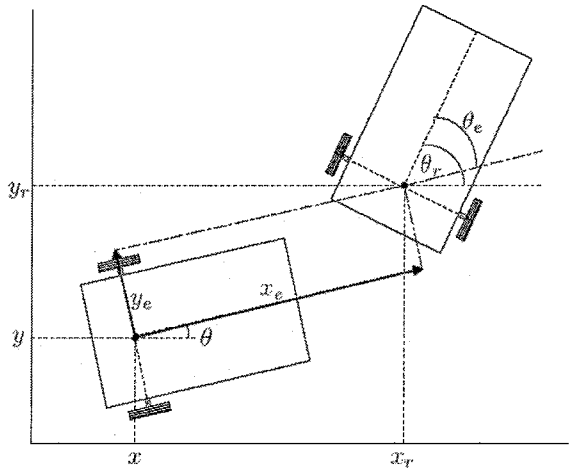

Figure 2.2: Definitions of the coordinates $\underline{x}$, reference-coordinates $\underline{x}_r$ and the error coordinates $\underline{x}_e$ (figure taken from [5])

2.1.2 Controller

The trajectory that the robot has to follow is generated by a reference system

$$\begin{cases} \dot{x_r} &= v_r \cos\theta_r \\ \dot{y_r} &= v_r \sin\theta_r \\ \dot{\theta_r} &= \omega_r. \end{cases} \tag{2.2}$$

To determine the error dynamics of the unicycle mobile robot, error coordinates x_e, y_e and θ_e are defined. As shown in figure 2.2, coordinate x_e is in the direction in which the robot is heading and y_e is the coordinate perpendicular to x_e. The combination of the kinematics of (2.1) and (2.2) gives an expression for the error coordinates of the system

$$\begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{bmatrix}. \tag{2.3}$$

Differentiating (2.3) with respect to time and substituting (2.1) and (2.2) leads to the error dynamics of this system

$$\begin{bmatrix} \dot{x_e} \\ \dot{y_e} \\ \dot{\theta_e} \end{bmatrix} = \begin{bmatrix} \omega y_e - v + v_r \cos\theta_e \\ -\omega x_e + v_r \sin\theta_e \\ \omega_r - \omega \end{bmatrix}. \tag{2.4}$$

The error dynamics are subsequently used in the Jakubiak state tracking controller [4]

$$\begin{aligned} \omega &= \omega_r + c_1 \sin\theta_e \\ v &= v_r + c_4 x_e - c_5 \omega_r y_e, \end{aligned} \tag{2.5}$$

with $c_4 > 0$ and $c_5 > -1$. Furthermore, it has been proven that if $v_r(t)$ and $\omega_r(t)$ are bounded and persistently exciting, the error dynamics is local-exponentially stable. The error coordinates x_e and y_e that appear in (2.5) are not directly known from measurements. They will be reconstructed using the orientation-error estimate ($\widehat{\theta_e}$ or, more precisely: $\widehat{\sin\theta_e}$), which in turn is calculated in the orientation-error-observer. The orientation-error-observer makes use of the reference coordinates and the measured position of the robot. All this is put together by using the cascaded form.
for a more detailed description of the orientation-error observer tracking controller see [6] and [4].

2.2 Implementation of the Controller

2.2.1 The Experimental Setup: BellyBot

To implement the theoretically derived orientation-error observer tracking controller, an experimental setup is available in the form of BellyBot. BellyBot is a mobile two-wheel drive unicycle robot (see figure 2.3). It is equipped with position and orientation measuring devices, in the form of the eBeam position measurement system and a Honeywell digital compass. To make the robot function properly, some other equipment is needed, like data-acquisition equipment, power electronics and a personal computer. In [5] the experimental setup is described in more detail.

2.2.2 Extra Features

The derived orientation-error observer tracking controller is implemented in MatLab/Simulink together with some extra features. To smoothen the measurements made by the coordinate measuring system, a reduced order state filter was added in the MatLab/Simulink-model. This filter is a manually tuned low-pass filter.(See [5] for a more detailed description)

The existing reference generater is able to generate two numerically calculated references, namely a circle and an eight-figure. The circle reference requires constant velocities and the eight-figure has variable velocities $v_r(t)$ and $\omega_r(t)$. Furthermore, the reference generator has an extra

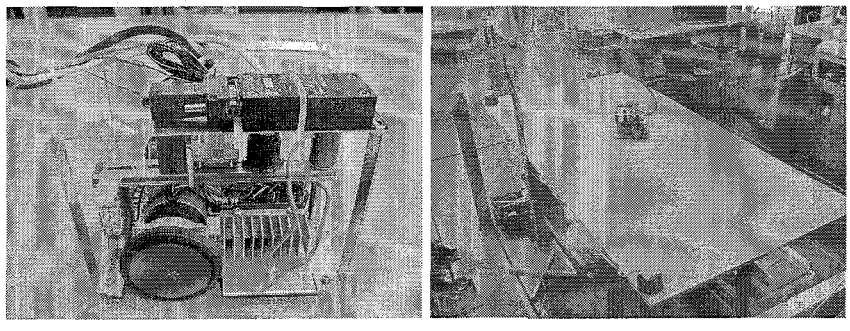

Figure 2.3: The available unicycle mobile robot, BellyBot, and its experimental environment (pictures taken from [5])

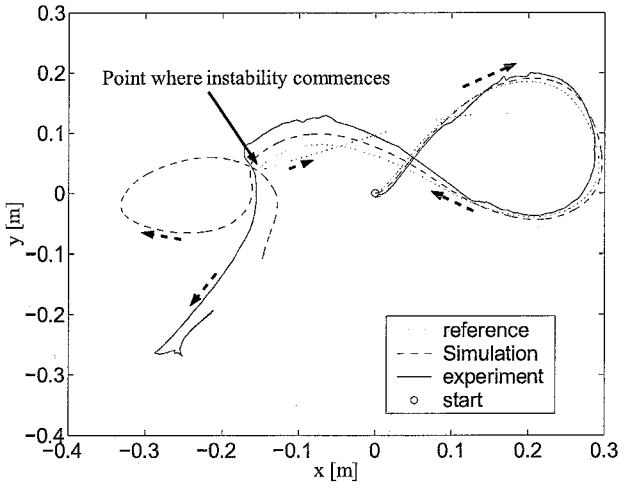

Figure 2.4: xy-plot of a operator defined reference trajectory, and instabilities showing in both simulation and experiment

mode in which the reference trajectory is specified manually by the operator. This is done by using the movements of the operators mouse. After being filtered, these signals are converted into the input needed to control BellyBot. It is of course impossible to draw an identical mouse trajectory twice or even more times in succession. However sometimes this is desirable, for instance when the effect of a changed parameter is examined. In order to reproduce a previously generated mouse trajectory, a new reference-mode was added to the existing set. This mode loads a previously saved mouse-reference trajectory (in the form of a .mat-file) and sends it to BellyBot again, as a new reference-trajectory.

Two more reference-modes were added, namely one where the angular velocity remains constant and the forward velocity increases linearly and one where the forward velocity remains constant and the angular velocity increases linearly. The purpose of the added reference-modes will be discussed further in chapter 3.

2.3 Previous Results and Encountered Problems

In figure 2.4 an xy-plot created with the current setup is shown. The reference is generated by a .mat-file, which contains data from a previously recorded mouse trajectory. It is visible that in the first part of the trajectory, both the simulation and BellyBot track the trajectory well. Only a slight offset in the y_e-direction, the direction perpendicular to the robot's heading, is present. This behavior is already noted in the report [5] and it is determined to be caused by the system delay time. The reason that the offset is not corrected by the controller is caused by the fact that the first line of (2.5) contains only $\sin\theta_e$ as an error-term. This means that when the orientation-error is zero, the angular velocity input to the robot will be equal to the reference signal ω_r. However, to reduce the error coordinate y_e, the angular velocity ω of the robot must differ from the reference angular velocity ω_r. And since this is not the case, an offset in the y_e-direction will continue to exist.(see [5])

In figure 2.4 it is also visible that from a certain point unstable behavior commences, and the unicycle loses track. This happens both in simulation and with the experimental setup. In the report [5] this unstable behavior is already mentioned. Instabilities occurred while using the circle reference. It was noticed that there is a maximum velocity at which the circle reference can be driven. When a higher velocity is imposed, the system becomes unstable and will no longer follow the given trajectory. With the operator controlled reference one can easily generate a reference-

trajectory that produces unstable behavior of the system. The cause of this unstable behavior will be examined in chapter 3.

Chapter 3

Unstable Behavior of the Tracking Controller

In this chapter it is attempted to find the cause for the unstable behavior which occurs when large input signals are fed into the unicycle system. Since the existing orientation-error observer tracking controller is quite a complicated system, a cause for the problem cannot immediately be pointed out. Not to mention the possibility that the problem has a number of connected causes. So, in order to say more about the real cause for the unstable behavior it is necessary to systematically cancel out possible causes.

In the section 3.1 a description of the methods used for deliberately creating unstable behavior is given. The sections 3.2 to 3.4 present a systematical approach for narrowing the search-area for the main cause of unstable behavior. And finally in section 3.5 these results are put together and a conclusion on the probable main cause is given.

3.1 Creating Unstable Behavior

To be able to study instability in a system it is necessary to have a reliable and consistent method of creating unstable behavior. In the previous chapter it is pointed out that the instability problem arises when an operator generates a reference-trajectory using the mouse-reference. When the reference-trajectory contains large and sudden changes in heading and/or position (i.e. large values for forward velocity v and angular velocity ω, the available input) the system becomes unstable. This leads to the idea to create a reference-generator in which values v and ω steadily increase, hoping to find the values of the input-signals at which instability first occurs. To this purpose, two new generators are designed. The first one keeps the angular velocity ω equal to zero and has a linear increase in the forward velocity $v_r(t)$. In the second generator the angular velocity increases linearly and the forward velocity v_r is kept at a constant value. Plots of the resulting trajectories are shown in figures A.1 and A.2 in appendix A. The two new reference generators are added to the already existing reference generators and implemented in MatLab/Simulink in both the simulation model and in the real-time controller connected to the experimental BellyBot setup.

When a simulation with the linearly increasing angular velocity generator is run, results as in figure 3.1 are obtained. It is clearly visible that at a certain angular velocity the system becomes unstable and no longer follows the reference trajectory. The generator with linearly increasing forward velocity also produces unstable behavior, starting at a certain forward velocity. After this critical speed has been reached the unicycle cannot keep up with the reference trajectory but stays tracking the straight line. This is due to the fact that the input signal for the angular velocity ω_r is kept at zero. In the rest of the report, use will be made of the linearly increasing angular velocity reference generator. This is because this generator produces better graphical information than the linearly increasing forward velocity reference generator.

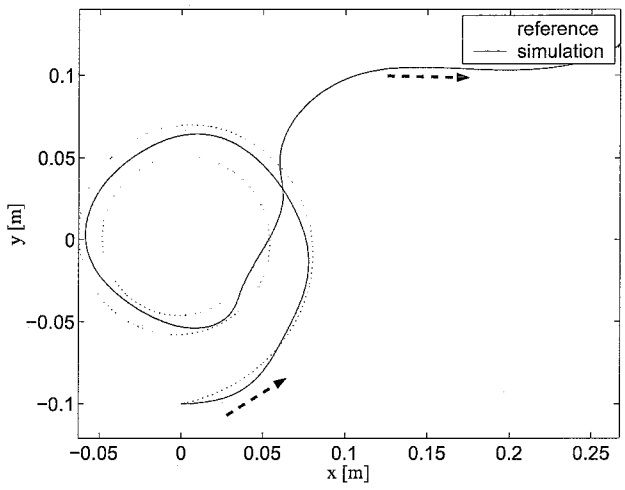

Figure 3.1: xy-plot of a simulation with the linearly increasing angular velocity reference generator applied

3.2 Effect of System Configurations

In this section the effect of actuator saturation, system delay and measurement quantization on stability of the system is examined.

Actuator Saturation The fact that instability occurs when reference signals increase beyond a certain value gives rise to the idea that actuator saturation might be a cause for the stability problem. Actuator saturation often causes problems when theoretically derived control laws are put into practise. The controller affects a system by manipulating the input. This input is often directly coupled to an actuator, in our problem the the inputs v en ω are coupled to BellyBot's electric drives. In the case of large errors a controller will give large input-signals to eliminate the existing error. However, BellyBot's electric drives have a maximum speed at which they can operate. Actuator saturation occurs when the controller produces input-signals that exceed the mechanical capabilities of BellyBot's drives. Actuator saturation can have a negative effect on the stability of the system because the input desired by the controller cannot be handled by the unicycle-system and the error will not be eliminated sufficiently.

Actuator saturation is realistically modeled in the existing MatLab/Simulink model. (See figure B.1) It appears that actuator saturation does not affect the point at which instability occurs. In figure 3.2 the saturation limits for both $v(t)$ and $\omega(t)$ are plotted together with the velocities $v(t)$ and $\omega(t)$. These saturation limits are determined in the report [5] at 0.4 m/s for $v(t)$ and 5.1 rad/sec for $\omega(t)$. The instability is visible in figure 3.2 as violent changes in the velocities. It is also visible that this instability occurs well before the actuator saturation limits are reached. This implies that other causes for the unstable behavior have to be present.

System Delay The system delay, which is determined at 0.1 sec in the report [5], is caused by the actuator-system. In chapter 2.1 it was already mentioned that the system delay is the cause for the permanent offset in the y_e direction. The effect on unstable behavior is less drastic. But when its turned of in a simulation with the linearly increasing angular velocity reference generator, some improvement in stability is visible (see figure 3.3). It takes longer before the system to become unstable. In other words: instability occurs at a higher angular velocity which leads to a more stable system. However, since instability does still occur, the system delay cannot be the main cause for it.

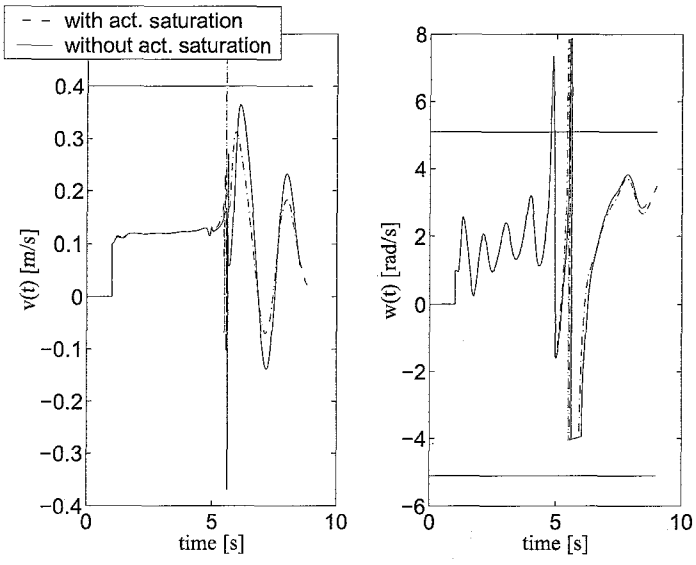

Figure 3.2: $v(t)$ and $\omega(t)$ together with saturation limits

Measurement Quantization The position and orientation measurement systems used in the BellyBot setup have got a certain resolution. These resolutions have also been implemented in the simulation by a quantization of the position and orientation signals. To test what effect the quantized measurements have on the occurrence of unstable behavior, a comparison has been made on simulation-level. In figure 3.4 a xy-plot of the simulation with and without measurement quantization is shown. Clearly, measurement quantization has virtually no effect on the instability problem.

3.3 Numerical Errors

Because the computer plays a vital role in the control and driving of the system, this section covers the effects that limitations in numerical computations can have on unstable behavior of the unicycle-system. The effects of various methods and parameters are examined in simulation and on the experimental setup.

3.3.1 Solver

Matlab is equipped with several different types of fixed and variable step ordinary differential equation (ODE) solvers. The solver used in the available model (see figure B.1) is ODE1. This is a fixed step solver for continuous time systems with integration based on Eulers method. ODE1 is the fastest, but least accurate solver available. To determine if the lack of accuracy of this method is a cause for instability in the system, some different solvers are put to the test and results are compared to the ODE1 solver. The alternative solvers tested are:

- ODE45; A variable step solver for continuous time systems based on the Runge-Kutta (4,5) formula
- Fixed step solver for discrete time systems available in MatLab/Simulink
- Variable step solver for discrete time systems available in MatLab/Simulink

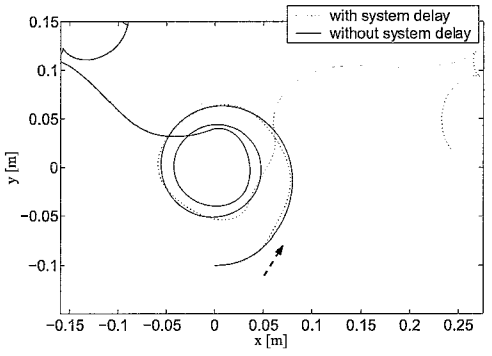

Figure 3.3: xy-plot of simulation with and without system delay

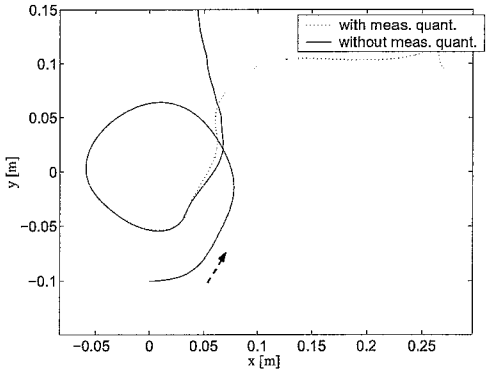

Figure 3.4: xy-plot of simulation with and without measurement quantization

The different solvers are compared with the linearly increasing angular velocity reference generator and with actuator saturations in v and ω and the system delay switched on. Results of this test are shown in appendix C. It is clear to see that the different solvers have no real effect on the speed at which the system becomes unstable. At most they affect the way in which the unstable behavior manifests itself. For more detailed information about the used ODE solvers, the reader is referred to [3].

3.3.2 Time-step

Within fixed time-step solvers, the size of the time-step dt is of great effect on the accuracy of the calculation. Smaller step-sizes lead to more accurate results, but are more expensive in terms of computing-time. A test is carried out with the fixed time-step solver ODE1, a small time-step has been compared to the standard time-step. (dt=1e-3 instead of dt=5e-3) But the results show that the differences between the two time-steps are negligible (see figure C.4).

3.4 Noijen Observer and Reduced Order State Filter

From the previous sections it is concluded that neither actuator saturation, system delay nor numerical calculation errors cause the instability problem. It appears that the cause has to sought in either the reduced order state filter, the Noijen observer or the Jakubiak state tracking controller or a combination of these parts. In order to find out what is (are) the weakest link(s) in the system, the default simulation file is adapted. The following system configurations are compared:

- default system with reduced order state filter and Noijen observer (see figure B.1)
- system with reduced order state filter but Noijen observer removed (see figure B.2)
- system with Noijen observer but reduced order state filter removed (see figure B.3)
- Ideal system with both Noijen observer and reduced order state filter removed (see figure B.4)

In all the configurations mentioned above, actuator saturation, system delay and measurement quantization are switched off.

The comparison is made by looking at a plot of the input signals v and ω as functions of time and the xy-plot. In the plots (see appendix D)it is easy to determine at which values for the input the instability commences by looking for the point where violent fluctuations in the velocities start. The following paragraphs specify the differences between the configurations mentioned above. The simulations are carried out with the linearly increasing angular velocity reference generator. The configuration that incroporates actuator saturation, system delay, measurement quantization, Noijen observer and reduced order state filter is called the default model. The system with only the Jakubiak controller and no other elements is called the ideal system.

Default Simulation The plot of ω shows that the first signs of unstable behavior appear at an angular velocity of approximately ω=3.5 [rad/sec]. Also visible in both plots is the fact that after some time a new 'stable' position is obtained. This has however another position and input-signals. The cause and meaning of this new stable position is not further examined.

Simulation with Noijen observer removed The plot of ω shows that the first signs of unstable behavior appear at an angular velocity of approximately ω=11.0 [rad/sec]. In the plot of v however shows unstable behavior from the 12th second and further. If we look at the plot of ω, a small change is also visible in the slope of $\omega(t)$. The xy-plot of this simulation shows very different, but equally unacceptable behavior, as the default simulation.

Simulation with reduced order state filter removed This configuration shows a considerable improvement over the previous two. The plot of $\omega(t)$ shows no signs of unstable behavior until it reaches an angular velocity of ω=13.0 [rad/sec]. In this case the forward velocity v remains at an expected (nearly)constant value. The xy-plot of figure confirms this improved stability behavior. The unicycle-system follows the reference for a much longer period, before eventually becoming unstable.

Ideal system simulation The ideal system unexpectedly appears not as ideal as it should be. In the plot of $\omega(t)$ it can be seen that the system turns unstable at about the same angular velocity as the system with the reduced order state filter removed. Further, in the plot of $v(t)$ it is visible that the forward velocity is not a constant as it is supposed to be. $v(t)$ decreases before the unstable behavior in the $\omega(t)$-plot appears. This effect remains unexplained. But might play a role in the instability problem.

3.5 Conclusion on Unstable Behavior

Research has been carried out on several parameter and system configurations in order to find out what causes the unicycle-system to become unstable when large inputs are given. It appeared that instability even occurs in the ideal system. Probably because the the stability requirements of the Jakubiak tracking controller are no longer satisfied. These stability requirements are mentioned in section 2.1.2. Especially the requirement that v_r is bounded and ω_r is persistently exciting might cause problems. The rest of the elements have got variable effects on the stability of the system.

From section 3.2 we can conclude that, actuator saturation, system delay and measurement quantization have got little effect on the destabilization of the system. Even less effect can be attributed to numerical errors. Their role in deteriorating unstable behavior is minimal. The reduced order state filter worsens the instability of the system to large extent. From section 3.4 it becomes clear that the Noijen observer has got little effect on the unstable behavior of the system. The element which has the largest effect on the instability is the reduced order state filter.

It must be said that the linearly increasing angular velocity reference generator used produces a worst-case scenario. Because the angular velocity is ever increasing, the unicycle system and controller have no chance to recover. If the operator controlled mouse reference generator is used, recovery is more likely and as a result of that it is almost impossible to create an unstable system,

especially when the configuration without the reduced order state filter is used. However, this is true for simulation only. In the experimental setup the reduced order state filter is of great importance and when it is removed, disturbances and noise from the position-measurements vastly reduce the accuracy with which BellyBot follows its reference trajectory. Finally table 3.1 gives a short survey of the effect that various tested elements have on the stability of the system.

Configuration/parameter	**Importance**
ODE solver	$--$
Time step	$--$
Actuator saturation	$+/-$
Measurement quantization	$+/-$
System delay	$+$
Noijen observer	$+$
Reduced order state filter	$++$

Table 3.1: Survey of different model-configurations and the importance in deteriorating unstable behavior

Chapter 4

A Combination of Tracking and Stabilizing

In the previous chapter, no concrete cause for the instability of the tracking control problem for the unicycle-system has been determined. And subsequently no satisfying solution has been found, removing the reduced order state filter drastically improves performance in the simulation, but removing it in the experimental model is not an option. In this chapter a suggestion will be made to get round the instability problem by resetting the system as soon as instability occurs. This will be done by switching from the orientation-error observer tracking controller to a posture stabilization controller when the system becomes unstable. In section 4.1 the basic principles of the combined controller will be presented and in section 4.2 the implementation together with the encountered problems and results are discussed, finally some possible improvements for the combined tracking/stabilizing controller are given in section 4.3.

4.1 Basic Principles of the Combined Controller

When the orientation-error observer tracking controller is used, problems arise when large input is provided to the unicycle-system. This happens when the error-coordinates $\underline{x}_e$ become large. The resulting unstable behavior is undesired and causes the unicycle-system to loose track of its given reference-trajectory. A posture stabilizing controller on the other hand works best when relatively large error coordinates are present. But it can become unstable when the error coordinates converge to zero. Due to the opposite conditions in which these controllers work best, the idea is to combine them to tackle both their weaknesses.

In this combined tracking and stabilizing controller the default control mode will be tracking, since this mode produces more accurate results than the stabilizing mode when following a given trajectory. While the tracking-controller is active, the x and y error (measured with respect to a fixed absolute coordinate system) are continuously monitored. When the absolute value of either the x or y error exceeds a given constant value, the tracking controller has lost track and will be turn off. At the same time the stabilizing controller is activated, which has the task to eliminate the error up to the point where the absolute value of both the x and y error converge below a certain (small) constant value. From this point the control is switched back from the stabilizing controller to the tracking controller, which can now eliminate the remaining error and continu tracking the reference trajectory. In this form the stabilizing controller provides a sort of reset when the tracking controller becomes unstable. It must be noted however, that to do its work properly, the stabilizing controller needs some time to converge. When a reference with no rests or slow sequences is generated, the stabilizing controller will not have enough time to stabilize. This is for example the case with the linearly increasing angular velocity reference generator.

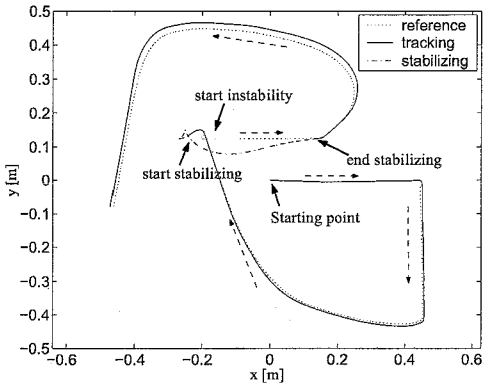

Figure 4.1: xy-plot of a trajectory followed with the combined tracking/stabilizing controller (simulation)

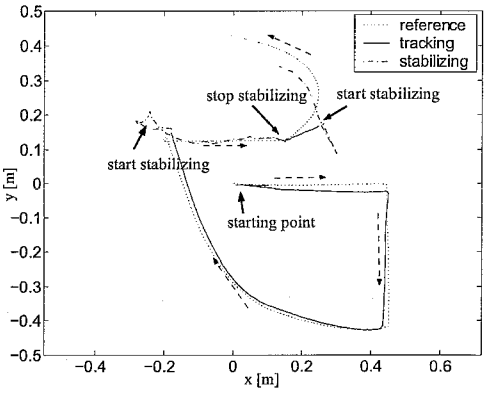

Figure 4.2: xy-plot of a trajectory followed with the combined tracking/stabilizing controller (experiment)

4.2 Implementation

The stabilizing controller used in this combined controller is the piecewise continuous state feedback stabilization controller as presented by Pourboghrat (see [2]). This and other types of posture stabilizing controllers have already been implemented and compared on BellyBot (see [5]). The main advantages of this controller over the other ones tested appeared to be the ease of tuning for the control-parameters. Furthermore this controller is proved to be globally exponentially stable. For more information about the derivation of the control law or its exact working principles the reader is referred to [2].

In figure B.5 an illustration of the MatLab/Simulink model is presented. The basis of this model is the existing tracking controller but some extra features have been added. Of course the Pourboghrat controller has been implemented in the model. This controller was directly taken from the report [5]. Further a switch has been added. This switch performs the task of switching between tracking controller and stabilizing controller as described in the previous section. The Pourboghrat controller in combination with BellyBot has previously only been used with a constant target configuration. This is the target location and orientation of the robot. In this application however the target configuration will be generated by the reference-generator and will therefore be a function of time. This proved not to be a problem if the target orientation is set to $\theta = 0$. For other values of θ the Pourboghrat stabilizing controller did not function as expected. This is probably due to an error in the implementation of the controller. The reason that this error did not show up earlier is probably because in [5] the Pourboghrat controller was only tested with a target configuration at the origin with $\theta = 0$.

In order to switch back from the stabilizing-controller to the tracking-controller properly, it is necessary to provide the tracking-controller with the correct initial conditions. The required conditions for x_0 and y_0 are sampled from the position of the reference-signal at the time of a switch-back. The condition θ_0 is set to zero, for reasons explained above. Furthermore the initial conditions for the Noijen observer are set to 0 and 1. Due to the fact that the initial condition for θ has to be set to zero, the tracking-controller will only pick up the the unicycle-system correctly if it starts from a horizontal position ($\theta = 0$). Otherwise the initial condition and the real orientation of the unicycle do not match and the system will fall back into stabilizing mode. This is a severe limitation to the systems functionality. In figure 4.2 a plot is shown of a trajectory which causes the unicycle to destabilize at a certain point and then gives it time to recover. From the figure it is clearly visible that some time after the instability occurs, the Pourboghrat controller is switched on and converges the unicycle to the (stationary) reference signal. It is also visible that the

reference signal is horizontal in order to keep $\theta = 0$. when the error has converged below the given criteria, the tracking-controller is switched back on. From this point on the unicycle is tracking the reference again.

If the same reference-signal is provided to the experimental setup, the results are not as good as in simulation. In figure 4.2 a xy-plot of the experimental BellyBot is given. It is visible that the robot becomes unstable at the same position as predicted in the simulation. The stabilizing-controller converges the position-error efficiently, but the convergence of the orientation-error is not good. In the stabilizing sequence the orientation of the robot is constantly altered in order to converge the position-error. This causes the robot to reach its target-configuration while its orientation angle $\theta \neq 0$. And as reported earlier, in order to switch back to the tracking-controller properly, the orientation angle has to equal zero. This is not the case and the result is visible in figure 4.2. The robot cannot track the reference properly and jumps back into stabilizing-mode.

The last minute orientation-angle changes that BellyBot performs might be caused by a combination of the following points:

- The compass used for orientation measurement has a time delay of 0.44 seconds, which is approximately 5 times larger than the system delay caused by the actuator-system. This makes the compass unsuited for control-applications. For further information about the compass, the reader is referred to [5].
- BellyBot is a tethered robot. A large amount of cabling connects the sensors and actuator-system with the fixed hardware and the personal computer. This cabling, in combination with the fact that BellyBot is very light, negatively influences the behavior of BellyBot to large extend.

4.3 Possible Improvements

As mentioned in the previous section, the combined tracking/stabilizing controller is not as effective as it was intended to be. In order to improve its behavior some changes could be made to the current design. Some possible changes are listed below:

- Debugging the implementation of the Pourboghrat controller. Possibly there is an error, because it only works well if target-orientation $\theta = 0$ is used. While theoretically no difference should exist between $\theta = 0$ and values with $\theta \neq 0$.
- The Pourboghrat-controller, in contrast to the Jakubiak state tracking controller does not make distinction between a positive or negative forward velocity, when converging to the target configuration. This can lead to the problem that the unicycle-robot arrives at its target-configuration back to front, which in turn leads to problems when switching back to the tracking-controller. Because an orientation-error of 180 degrees is present, the system instantaneously becomes unstable. A solution to this problem, in the form of an extra open-loop controller which turns the robot around its axis until the orientation-error is small enough for the tracking-controller to function properly, could be implemented in the existing combined controller.
- In the current combined tracking/stabilizing model (see figures B.5 and B.6) the target configuration is taken directly from the reference-generator. This produces a target-configuration which is a function of time. The Pourboghrat controller can handle this if the reference-trajectory is close to stationary. It would be better to sample the reference-trajectory at the moment the stabilizing-controller is activated and use this sample as the target-configuration. When arriving at the target the controller has to determine whether the error is small enough to switch back to the tracking-controller. When the error is to large, another sample of the reference-trajectory has to be taken and a new target-configuration is created. This proces will continue until the operator gives the robot time to catch up with the trajectory.

Chapter 5

Conclusions and Recommendations

Conclusions

This report analyses a unicycle mobile robot in order to find a solution for the unstable behavior that occurs when an operator controlled reference generator is used in combination with an orientation-error observer tracking controller. First a short overview of the existing controller and its problems was given. This was followed by a description of what happens when the system becomes unstable. Three new generators, designed to consistently produce unstable behavior, were implemented in the existing MatLab/Simulink model. With these generators the effect that several parameters and system configurations have on the stability of the system has been compared. This comparison did not provide a clear cause for the occurrence of unstable behavior. It does however narrow the search-area for further research on this subject.

Conclusion:

The research after the cause for the unstable behavior leads to the following conclusion:

No real cause for the stability problem has been found, but the effects of several components on the stability of the system were determined. Also an alternative solution for the instability problem, in the form of a combined controller, has been presented.

The effect of configurations/parameters on the stability of the system are determined.

- Numerical errors have negligible effect on the stability of the system
- Actuator saturation, measurement quantization and the use of a Noijen observer have little effect on the stability of the system
- The system delay and the reduced order state filter decrease the system's stability considerably

In order to reset the system in cases of instability a combination of a tracking controller and a posture stabilizing controller was designed. In its current form, this combined tracking/stabilizing controller does not provide satisfying results. Stability and reliability are not sufficiently high enough to function as a backup for the tracking controller. Some possible improvements to this combined controller are suggested in the following section.

Recommendations

The research carried out narrows the search-area for further research on the instability problem, but it does not provide a conclusive result on the stated problem. In order to find the real cause for the instability problem the following recommendations are stated:

- Further research may reveal whether the instability is indeed related to the stability requirements of the tracking controller.
- It appeared that in some unstable situations, the unicycle-system converged to a new 'stable' position. Be it with different position and velocities than dictated by the reference-signal. The cause and effect of this behavior needs further research to be explained.

To solve the problems encountered while implementing the combined tracking/stabilizing controller some improvements were proposed at the end of chapter 4. These improvements are discussed in the following recommendations:

- Debugging of the Pourboghrat stabilizing controller.
- Implementing an intermediate orientation control mode to reduce the orientation-error after stabilizing mode.
- Implementing reference-trajectory-sampler to create constant target-configurations for the Pourboghrat stabilizing controller.
- Replace compass on experimental BellyBot setup with an orientation measuring device with better control-applicability. (i.e.smaller time delay)
- Replace tether on experimental BellyBot setup with lighter version or design a better connection between cabling and robot, in order to reduce negative effect on behavior of BellyBot.

Appendix A

Reference Generators: Linearly Increasing Velocities

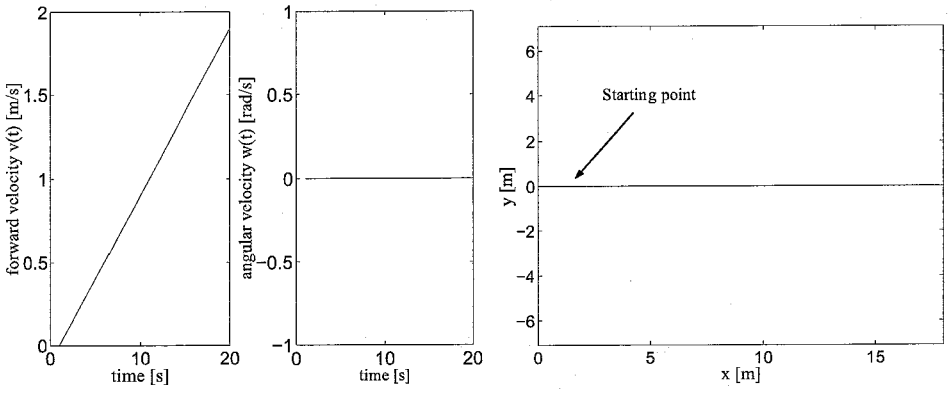

Figure A.1: Reference-trajectory with linearly increasing forward velocity $v(t)$

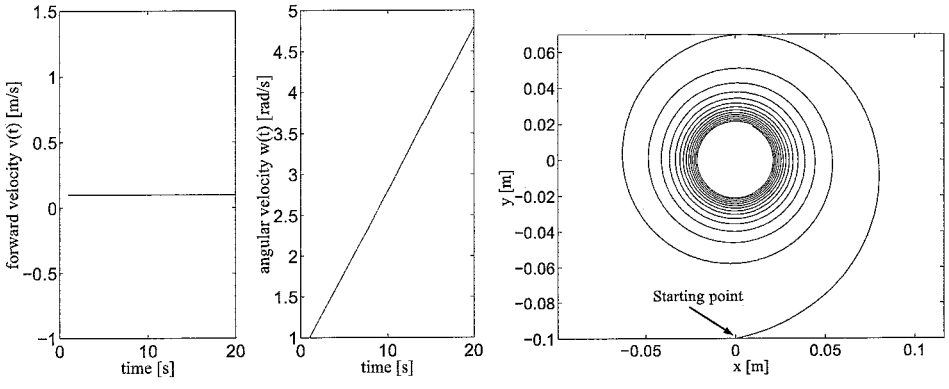

Figure A.2: Reference-trajectory with linearly increasing angular velocity $\omega(t)$

Appendix B

Models

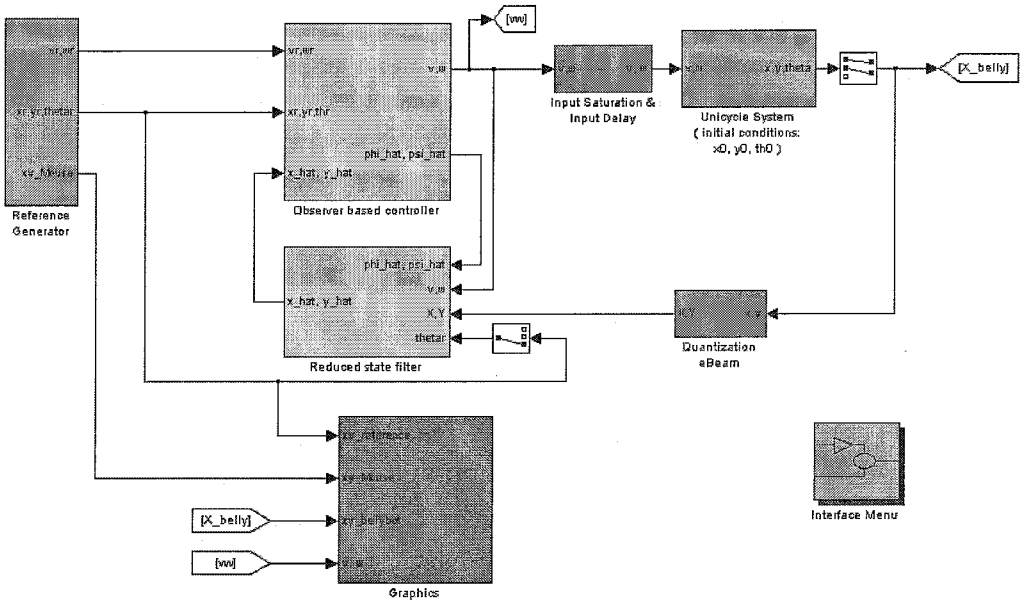

Figure B.1: The default simulation model. *SIM_DEFAULT.mdl* (Only upper layer shown)

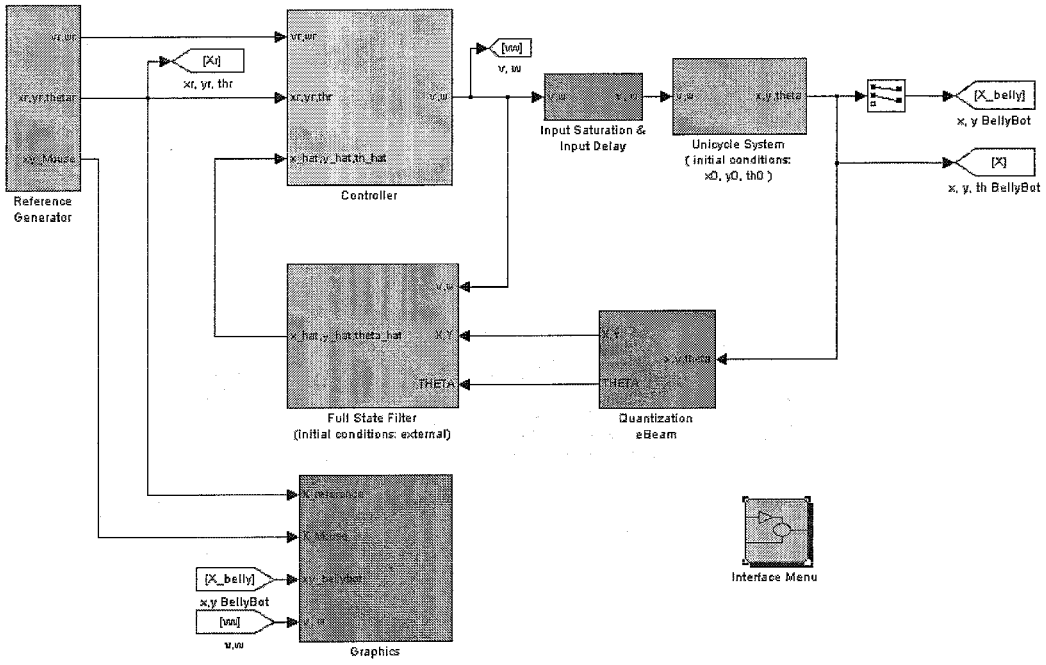

Figure B.2: The simulation model with Noijen observer removed. *SIM_NO_OBSERVER.mdl* (Only upper layer shown)

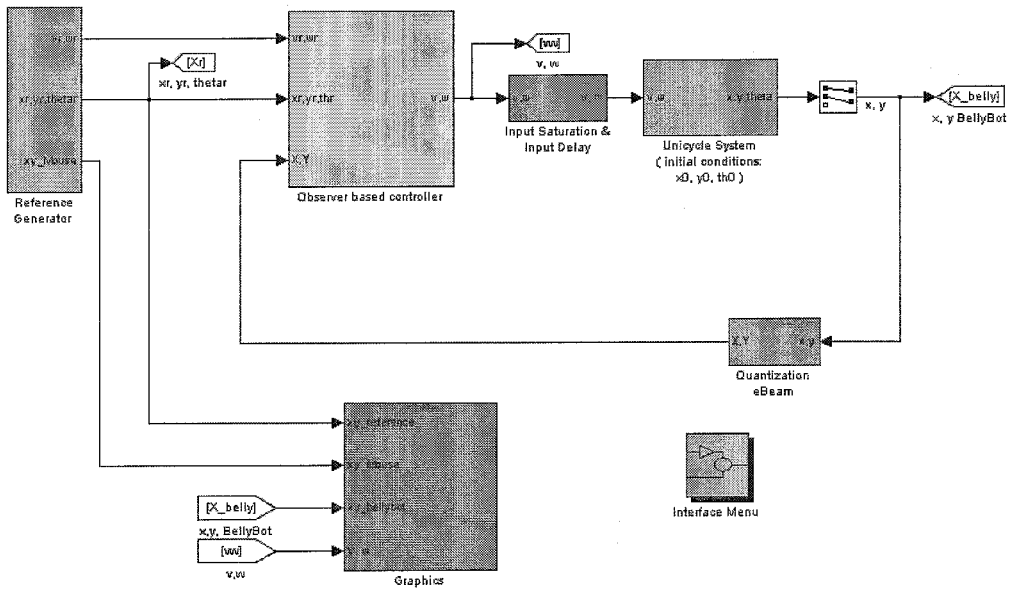

Figure B.3: The simulation model with reduced order state filter removed. *SIM_NO_FILTER.mdl* (Only upper layer shown)

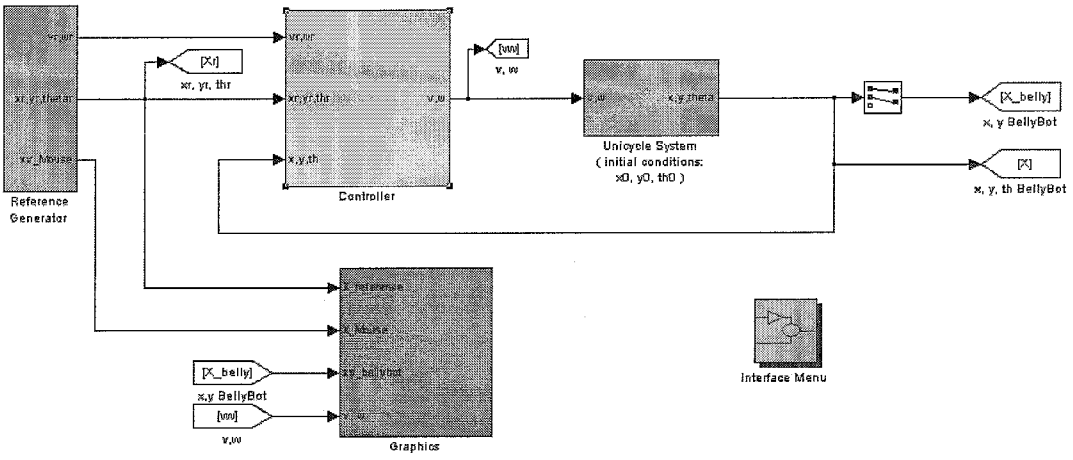

Figure B.4: The ideal simulation model. *SIM_IDEAL.mdl* (Only upper layer shown)

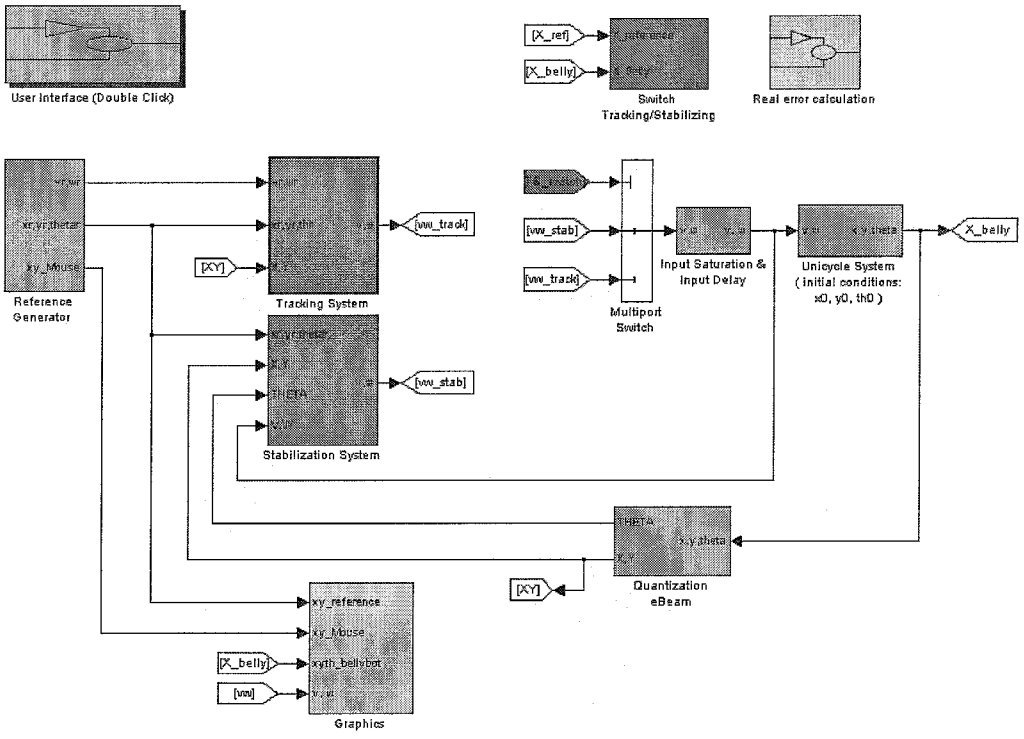

Figure B.5: The simulation combined tracking and stabilizing model. *SIM_TRACKING_STABILIZING.mdl* (Only upper layer shown)

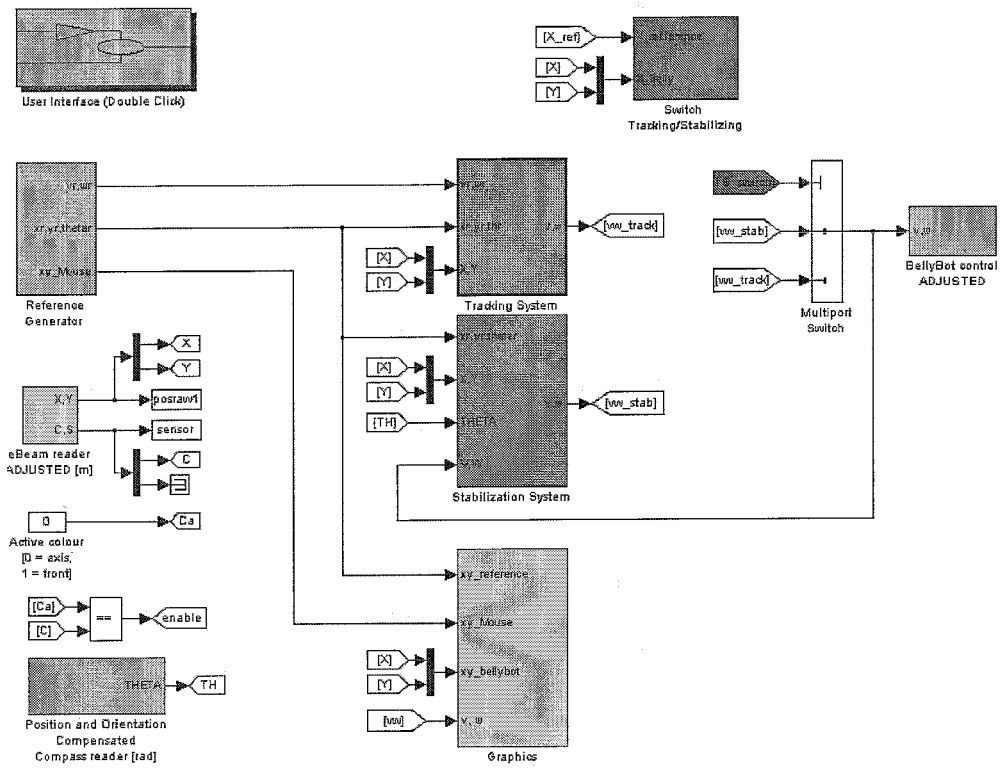

Figure B.6: The combined tracking and stabilizing model for use with experimental BellyBot setup. *BB_TRACKING_STABILIZING.mdl* (Only upper layer shown)

Appendix C

Numerical Errors

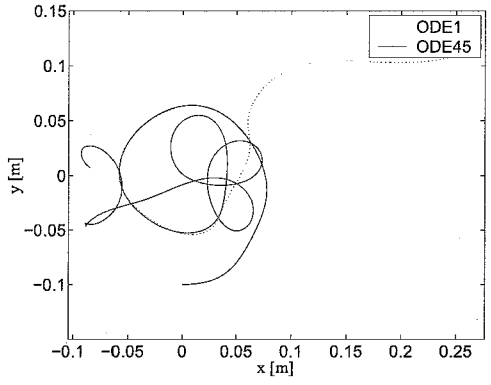

Figure C.1: xy-plot with ODE1 and ODE45

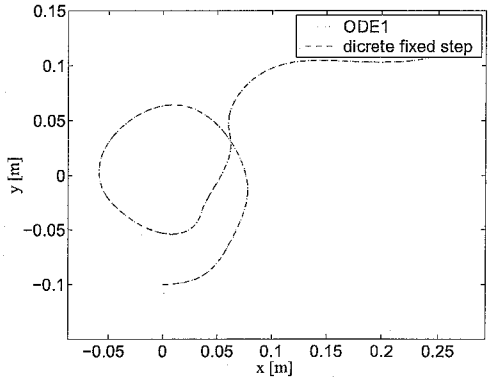

Figure C.2: xy-plot with ODE1 and fixed step discrete solver

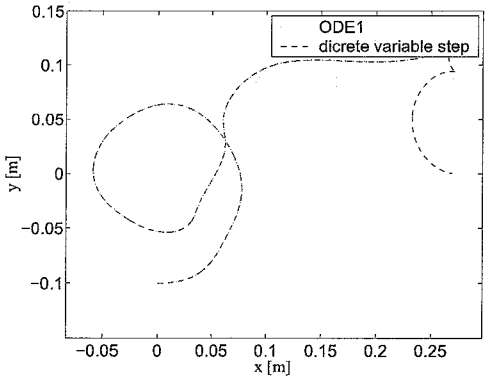

Figure C.3: xy-plot with ODE1 and variable step discrete solver

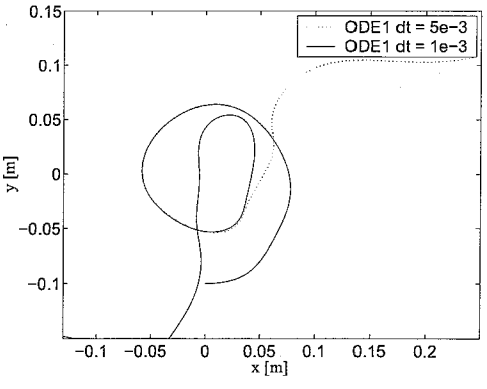

Figure C.4: xy-plot with ODE1 dt = 5e-3 and ODE1 dt = 1e-3

Appendix D

Results on Unstable Behavior

Results for the following files are shown for simulation with the linearly increasing angular velocity reference generator.

- default system with reduced order state filter and Noijen observer (model shown in figure B.1)
- system with reduced order state filter but Noijen observer removed (model shown in figure B.2)
- system with Noijen observer but reduced order state filter removed (model shown in figure B.3)
- ideal system with both Noijen observer and reduced order state filter removed (model Shown in figure B.4)

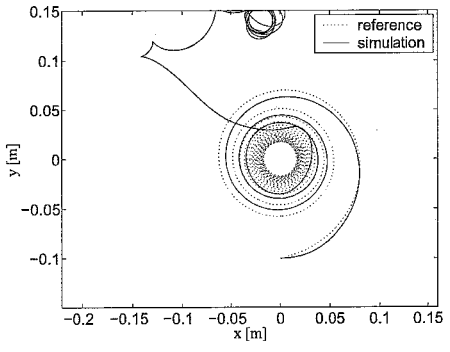

Figure D.1: xy-plot for default system

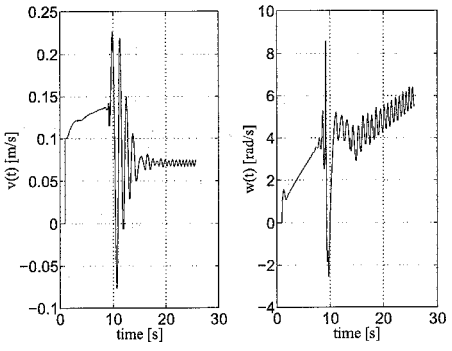

Figure D.2: vw-plot for default system

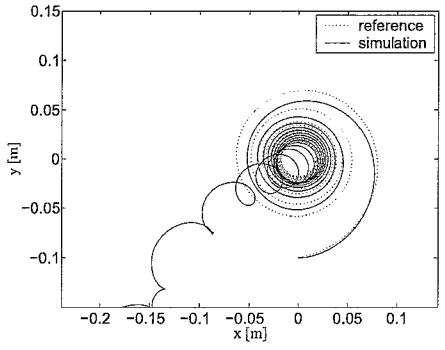

Figure D.3: xy-plot for system with Noijen observer removed

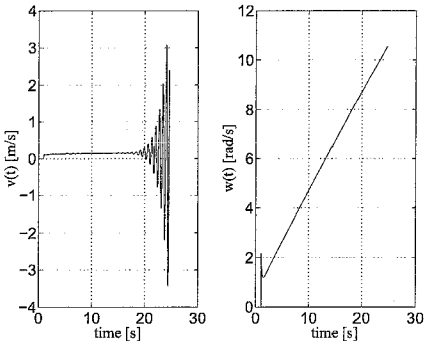

Figure D.4: vw-plot for system with Noijen observer removed

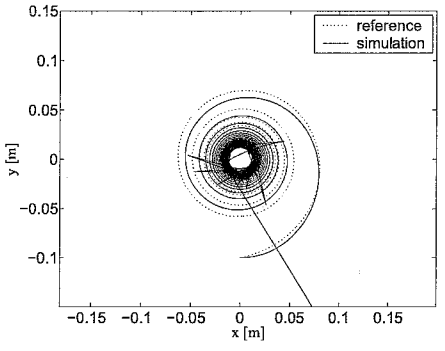

Figure D.5: xy-plot for system with reduced order state filter removed

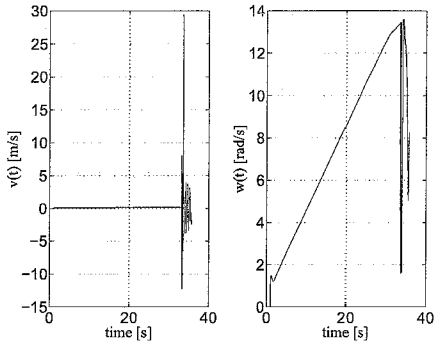

Figure D.6: vw-plot for system with reduced order state filter removed

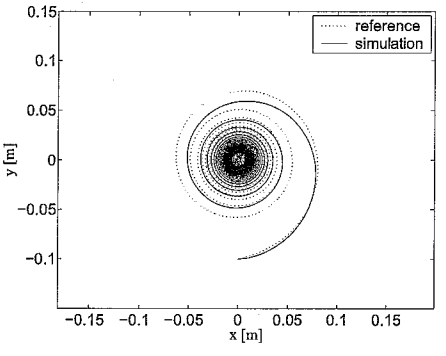

Figure D.7: xy-plot for ideal system

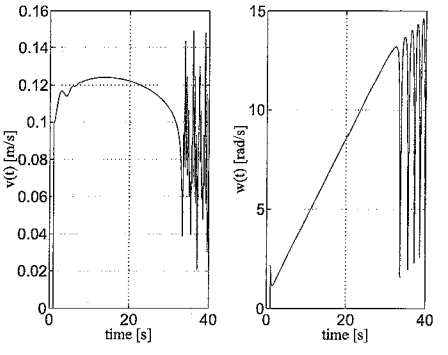

Figure D.8: xy-plot for ideal system

Bibliography

[1] E.Panteley, E.Lefeber, A.Loría, and H.Nijmeijer. Exponential tracking control of a mobile car using a cascaded approach. 221-226, Proc. IFAC Workshop Motion Control, Grenoble, France, 1998.

[2] F.Pourboghrat. exponential stabilization of non-holonomic mobile robots. *Computers and Electric Engineering*, (no.28):340–359, 2002.

[3] The Mathworks inc. Matlab documentation release 12.1. software documentation, The Mathworks inc., Natick Massachusetts USA, 2001.

[4] J.Jakubiak, E.Lefeber, K.Tchoń, and H.Nijmeijer. Two observer-based tracking algorithms for a unicycle mobile robot. *Int. J. Appl. Math. Comput. Sci.*, vol.12(no.4):513–522, 2002.

[5] N.J.M.Bosch and S.H. van der Meulen. Experimental tracking an stabilization of a unicycle mobile robot. DCT Report No. 2003.110, Eindhoven University of Technology, Department of Mechanical Engineering, Eindhoven, 2003.

[6] S.P.M.Noijen. An orientation error observer-controller combination for a unicycle mobile robot. DCT Report No. 2003.65, Eindhoven University of Technology, Department of Mechanical Engineering, Eindhoven, 2003.