ISSN 0280-5316
ISRN LUTFD2/TFRT--5751--SE

Stabilization of Vehicle Formations - A Case Study

Johannes Berglund

Department of Automatic Control
Lund Institute of Technology
June 2005

Department of Automatic Control
Lund Institute of Technology
Box 118
SE-221 00 Lund Sweden

Document name
MASTER THESIS

Date of issue
June 2005

Document Number
ISRNLUTFD2/TFRT--5751--SE

Author(s)
Johannes Berglund

Supervisor
Ather Gattami and Anders Rantzer at Automatic Control in Lund

Sponsoring organization

Title and subtitle
Stabilization of Vehicle Formations – A Case Study (Stabilisering av fordonsformationer, en fallstudie)

Abstract
This work treats a specific multi-vehicle control case, namely six vehicles on the Multi-Vehicle Wireless Testbed (MVWT), which is an arena at Caltech, California, built for doing experiments on small vehicles. A model is set up and a controller is found through simulations that stabilizes the system. Stability is proven through a Nyquist-like criterion and by looking at the poles for the closed system.

Keywords

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280-5316

ISBN

Language
English

Number of pages
42

Recipient's notes

Security classification

CONTENTS

ACKNOWLEDGEMENTS

I enjoyed doing this thesis a lot. It was very satisfying watching the simulation and animation the first time it worked. Also most things went very smooth, the Simulink model worked more or less on the first try and the same goes for the animation.
The only part that was problematic was showing stability, which consumed some energy. However my supervisor Ather Gattami was very reachable and helpful throughout the work, and specifically helped me a lot to sort the stability problem. A very big thank you to Ather, I can really recommend you as a supervisor.
My family is my biggest support. They always listen to my ramblings even if they don't understand half of it.
A thank you also goes to Andreas Wernrud, who has listened to all the problems that have arisen, and has come up with some ideas.

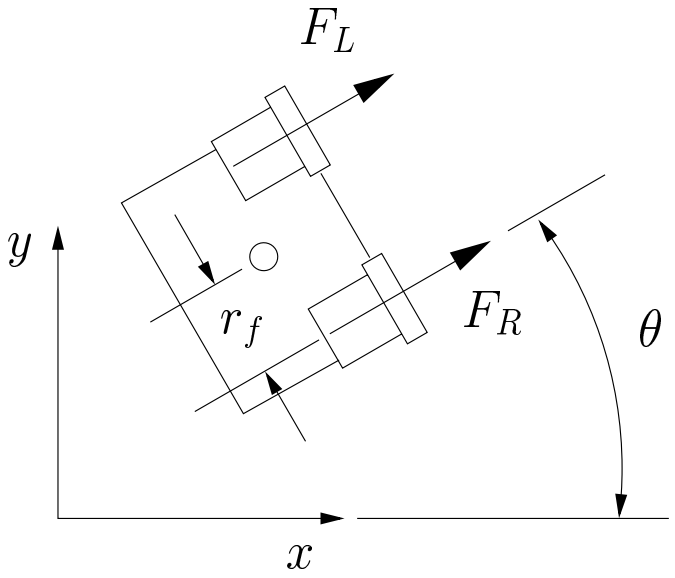

Figure 1: Kelly vehicle.

1 INTRODUCTION

The object of this work was first to find a controller that stabilizes a specific interconnected system, and then to do simulations to adjust the parameters and thus further optimize the performance.
The specific case setup consists of six vehicles on The Caltech Multi-Vehicle Wireless Testbed (MVWT). Each vehicle (named Kelly vehicle) has a rectangular shaped form seen from above with two fans used to control its motion, see figure 1. The MVWT arena is a restricted flat rectangular surface of approximate dimension 6.7 m × 7.3 m.
The task was to obtain and maintain the vehicles on a formation modelled in two dimensions, with the only information available to each vehicle *j*, being the distance to vehicle *j - 1* and the distance to vehicle *j + 1*.

2 PRELIMINARIES

2.1 MATRIX ALGEBRA

The Kronecker product for two matrices **A** and **B** is defined as

$$\mathbf{A} \otimes \mathbf{B} = \begin{pmatrix} \mathbf{A}_{11}\mathbf{B} & \mathbf{A}_{12}\mathbf{B} & \dots & \mathbf{A}_{1l}\mathbf{B} \\ \mathbf{A}_{21}\mathbf{B} & \mathbf{A}_{22}\mathbf{B} & \dots & \mathbf{A}_{2l}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{k1}\mathbf{B} & \mathbf{A}_{k2}\mathbf{B} & \dots & \mathbf{A}_{kl}\mathbf{B} \end{pmatrix}.$$

For a set of **N** matrices $\{\mathbf{M}_1, ..., \mathbf{M}_\mathbf{N}\}$ of size $r \times s$, we define the *direct sum* as the $\boldsymbol{Nr} \times \boldsymbol{Ns}$ block diagonal matrix $\hat{\mathbf{M}}$ whose $r \times s$ diagonal blocks are the matrices $\mathbf{M}_1, ..., \mathbf{M}_\mathbf{N}$ (in this order), and the other entries are zero, which we write as

$$\hat{\mathbf{M}} = \oplus \sum_{i=1}^{N} \mathbf{M}_i$$

For a given $\boldsymbol{N} \times \boldsymbol{N}$ matrix **Q**, define a $\boldsymbol{Nk} \times \boldsymbol{Nk}$ matrix $\mathbf{Q}_{(k)}$ by the equation

$$\mathbf{Q}_{(k)} = \mathbf{Q} \otimes \mathbf{I}_k,$$

where $\mathbf{I}_k$ is the $k \times k$ identity matrix. So, we have simply replaced each element of **Q** by the same element times the $k \times k$ identity matrix.

2.2 GRAPH THEORETICAL APPROACH TO OUR SYSTEM

This chapter contains an explanation of graph theory and how it is connected to the rest of this thesis. Most of the information comes from Gattami *et al* [1] and Fax *et al* [2].

Our system constitutes of six identical vehicles, all undergoing circular motion around the same point. Consider each vehicle as a dot in this section. Now if vehicle *i* can sense information from vehicle *j*, an arc directed towards vehicle *i* connects the vehicles.

As already stated, for our case each vehicle can sense information from its two index-wise closest neighbours, i.e. vehicle 3 for instance can sense vehicle 2 and vehicle 4.

The graph for our system can look like in figure 2.

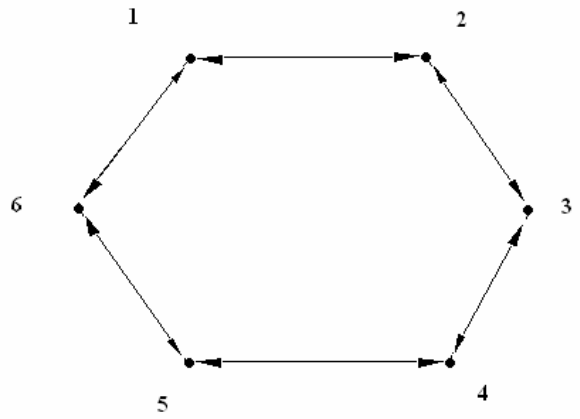

Figure 2: System graph

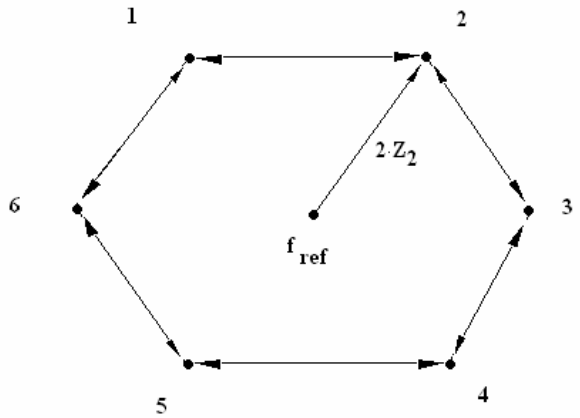

Figure 3: The weighted signal z_i

Each vehicle senses two neighbour vehicles. Now to make it easier to control the formation, we want these two sensed signals for each vehicle to be merged into one signal. We choose an equally weighted sum of the two signals to be our merged signal. Expressed in vectors we let the merged signal for vehicle *i* be represented by the mean value of the vector connecting vehicle *i-1* with vehicle *i*, and the vector connecting vehicle *i+1* with vehicle *i*. This merged signal z_i for vehicle *i* equals half the vector connecting vehicle *i* with a formation reference point, see figure 3.

We now have a graphical representation of our interconnected system. To be able to control the system we need a mathematical representation of our interconnected system:

Let the dynamics for each vehicle *i* be described on state space form by

$$\dot{x}_i = Ax_i + Bu$$

$$y_i = Cx_i$$

where y_i represents the 2D-spatial coordinates for vehicle *i*. From this, we want to extract our weighted measurement z_i, which as already mentioned is the mean value of the vector connecting vehicle *i-1* with vehicle *i*, and the vector connecting vehicle *i+1* with vehicle *i*. Or mathematically:

$$z_i = \frac{1}{2}(y_i - y_{i-1}) + \frac{1}{2}(y_i - y_{i+1})$$

From (2) we can now build our interconnection matrix H, satisfying

$$z = Hy:$$

$$\mathbf{H} = \begin{bmatrix} 1 & -1/2 & 0 & 0 & 0 & -1/2 \\ -1/2 & 1 & -1/2 & 0 & 0 & 0 \\ 0 & -1/2 & 1 & -1/2 & 0 & 0 \\ 0 & 0 & -1/2 & 1 & -1/2 & 0 \\ 0 & 0 & 0 & -1/2 & 1 & -1/2 \\ -1/2 & 0 & 0 & 0 & -1/2 & 1 \end{bmatrix}$$

This matrix is essential for this work, and is really the one thing that makes the work deviate from classical control.

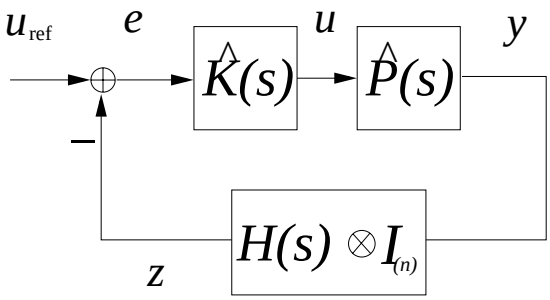

Figure 4: The interconnected system.

3 THE UNDERLYING MATHEMATICS

3.1 PROBLEM SETUP

This chapter contains a problem setup for the interconnected system used in this thesis. A lot of the stuff is taken from [1], which I recommend reading for a better understanding of interconnected systems.
Consider our set of six identical plants, **P(s)**, and its controllers, **K(s)**. Since each plant possesses two fans and two spatial coordinates, **P(s)** has two inputs and two outputs. Now to get the correct dimension of the controller, also **K(s)** has two inputs and two outputs. Let

$$\hat{\mathbf{P}}(\mathbf{s}) = \oplus\sum_{i=1}^{6}\mathbf{P}_i(\mathbf{s})$$

and

$$\hat{\mathbf{K}}(\mathbf{s}) = \oplus\sum_{i=1}^{6}\mathbf{K}_i(\mathbf{s})$$

Now consider the *interconnected* MIMO system given as in figure 4, where **H(s)** is the interconnection matrix function with proper dimensions.

To obtain stability we use a theorem from [1]:

Theorem 1. $\mathbf{U(s)} = \hat{\mathbf{K}}\mathbf{(s)(U_{ref} - Z(s))}$ *stabilizes the system*

$$\begin{cases} \mathbf{Y(s)} = \hat{\mathbf{P}}\mathbf{(s)U(s)} \\ \mathbf{Z(s)} = \mathbf{H_{(n)}(s)Y(s)} \end{cases} \tag{1}$$

iff the Nyquist plot of

$$\det[\mathbf{I_n} + \lambda_i(s)\mathbf{P(s)K(s)}]$$

for each λ_i *makes p anti-clockwise encirclements of the origin, where* λ_i**,** *i=1..N, are the eigenvalues of* **H(s)** *and p is the number of unstable poles of*

$$\lambda_i \det[\mathbf{P(s)K(s)}].$$

3.2 THE INTERCONNECTION MATRIX

To find a $\hat{\mathbf{K}}$ that stabilizes the system, we first need expressions for $\hat{\mathbf{P}}$ and **H**. We first look at **H:**

The interconnection matrix was computed in 2.2 is repeated here:

$$\mathbf{H} = \begin{bmatrix} 1 & -1/2 & 0 & 0 & 0 & -1/2 \\ -1/2 & 1 & -1/2 & 0 & 0 & 0 \\ 0 & -1/2 & 1 & -1/2 & 0 & 0 \\ 0 & 0 & -1/2 & 1 & -1/2 & 0 \\ 0 & 0 & 0 & -1/2 & 1 & -1/2 \\ -1/2 & 0 & 0 & 0 & -1/2 & 1 \end{bmatrix}$$

The eigenvalues of this matrix are:

$$\lambda_1 = 0,\ \lambda_2 = 0.5,\ \lambda_3 = 0.5, \lambda_4 = 1.5, \lambda_5 = 1.5,\ \lambda_6 = 2.$$

For our specific case:

$$\mathbf{H_{(n)}} = \mathbf{H_{(2)}} = \mathbf{H} \otimes \mathbf{I_2},$$

because each vehicle has two spatial coordinates.

$\mathbf{Z_i}$ now equals an equally weighted sum of the distances from vehicle *i* to its index wise two closest neighbours. (This does not necessarily mean that those two vehicles are the spatially closest ones).

This particular **H** is exactly the Laplacian matrix **L**, which originates in graph-theoretical aspects of the interconnection. For more information on the Laplacian see [2].

3.3 THE SYSTEM MODEL

To achieve our **P**, in (1), we need to start examining physical equations that describe the dynamical behaviour of our vehicles.
Basically, all the derivations and computations used for the actual dynamical behaviour of a Kelly vehicle, (in other words section 3.3 excluding 3.3.2) were already known, and were taken right off from Cremean *et al* [3] and Gogh [4]. Here follows a brief summation of the system model equations and the thoughts behind.

The simplified equations of motion (assuming perfect sensing and actuation, no delays, no disturbances, and linear friction) in cartesian coordinates for the vehicle are listed in equation (2). See figure 5 for reference.

$$\begin{aligned} m\ddot{x} &= -\mu\dot{x} + (F_R + F_L)\cos\theta \\ m\ddot{y} &= -\mu\dot{y} + (F_R + F_L)\sin\theta \\ J\ddot{\theta} &= -\varphi\dot{\theta} + (F_R - F_L)r_f \end{aligned} \tag{2}$$

These equations include four physical parameters: the mass *m*, the mass moment of inertia *J*, and the linear and rotational viscous friction coefficients μ and φ. The geometric parameter r_f is the distance between the centre of mass of the vehicle and each fan axis. *x* and *y* are the positional coordinates, θ is the orientation and F_R and F_L are the fan forces, see figure 5.

3.3.1 Linearization.

A linearization of equation (2) is not controllable around any of its equilibria. To achieve controllability, we can consider, for example, the error dynamics around a constant velocity and heading. That would give us a way to track a straight line path. However, because of the spatially constrained testbed, a more suitable vehicle motion would be tracing a circular path with a constant radius and a constant angular velocity.

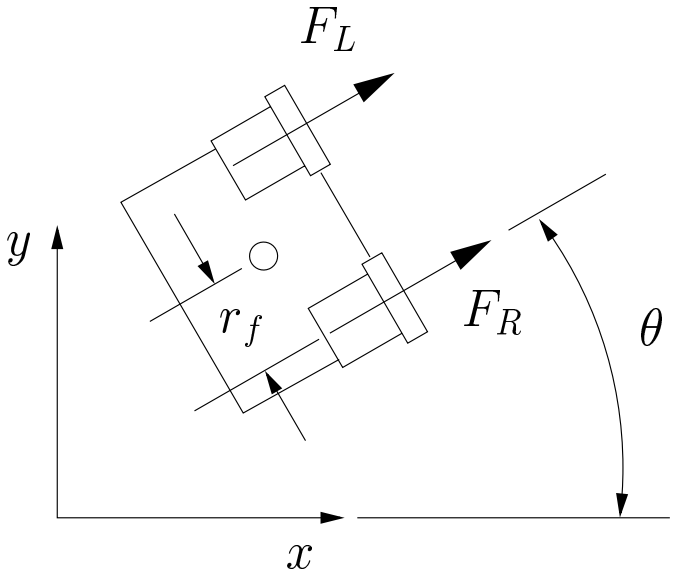

Figure 5: Kelly vehicle in a cartesian coordinate system.

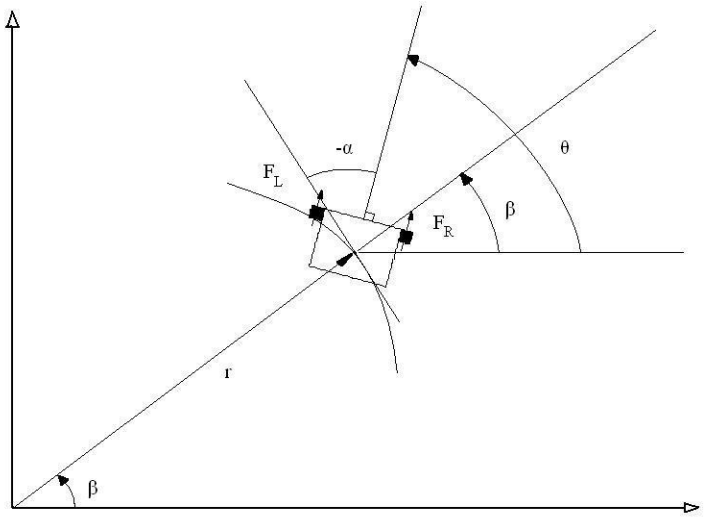

Figure 6: Kelly vehicle in a polar coordinate system.

Such motion is most easily analyzed when the model in equation (2) is written in polar coordinates (see figure 6):

$$\begin{aligned} m(\ddot{r} - r\dot{\beta}^2) &= -\mu\dot{r} + (F_R + F_L)\cos(\theta - \beta) \\ m(r\ddot{\beta} + 2\dot{r}\dot{\beta}) &= -\mu r\dot{\beta} + (F_R + F_L)\sin(\theta - \beta) \\ J\ddot{\theta} &= -\mu r_c^2\dot{\theta} + (F_R - F_L)r_f \end{aligned} \qquad (3)$$

A state vector suitable for this motion would be **x**, see below. A reference equilibrium state, $\mathbf{x_r}$ representing circular motion of constant radius ρ and angular velocity $\dot{\xi}$ is also described below:

$$\mathbf{x} = \begin{bmatrix} r \\ \beta \\ \theta \\ \dot{r} \\ \dot{\beta} \\ \dot{\theta} \end{bmatrix},$$

$$\mathbf{x_r} = \begin{bmatrix} r_r \\ \beta_r \\ \theta_r \\ \dot{r}_r \\ \dot{\beta}_r \\ \dot{\theta}_r \end{bmatrix} = \begin{bmatrix} \rho \\ \dot{\xi}t + \beta_{r0} \\ \dot{\xi}t + \theta_{r0} \\ 0 \\ \dot{\xi} \\ \dot{\xi} \end{bmatrix}.$$

Linearization of equation (3) around the reference state $\mathbf{x_r}$ results in the following system on state space form:

$$\begin{aligned} \dot{\mathbf{e}} &= \mathbf{Ae} + \mathbf{Bu} \\ \mathbf{Y_e} &= \mathbf{Ce}, \end{aligned} \tag{4}$$

where

$$\mathbf{e} = \begin{bmatrix} r_e \\ \beta_e \\ \theta_e \\ \dot{r}_e \\ \dot{\beta}_e \\ \dot{\theta}_e \end{bmatrix} = \mathbf{x} - \mathbf{x_r},$$

$$\mathbf{u} = \begin{bmatrix} F_{Re} \\ F_{Le} \end{bmatrix} = \begin{bmatrix} F_R - F_{R,nom} \\ F_L - F_{L,nom} \end{bmatrix},$$

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \dot{\xi}^2 & \frac{\mu\rho\dot{\xi}}{m} & -\frac{\mu\rho\dot{\xi}}{m} & -\frac{\mu}{m} & 2\rho\dot{\xi} & 0 \\ 0 & \dot{\xi}^2 & -\dot{\xi}^2 & -\frac{2\dot{\xi}}{\rho} & -\frac{\mu}{m} & 0 \\ 0 & 0 & 0 & 0 & 0 & -\frac{r_c^2\mu}{J} \end{bmatrix},$$

and

$$\mathbf{B} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{sign(\dot{\xi})}{\rho}\begin{bmatrix} -\frac{\rho\sin\alpha_0}{m} & -\frac{\rho\sin\alpha_0}{m} \\ -\frac{\cos\alpha_0}{m} & -\frac{\cos\alpha_0}{m} \end{bmatrix} \\ \frac{r_f}{J} & -\frac{r_f}{J} \end{bmatrix}.$$

The actual linearization is not listed here, but is carried out in [4].

3.3.2 LQR Control.

Using the notation in equation (1) we have for our particular system, for each vehicle:

$$\mathbf{Y} = \begin{bmatrix} r \\ \beta \end{bmatrix},$$

and

$$\mathbf{Z} = \begin{bmatrix} z_r \\ z_\beta \end{bmatrix}.$$

In order to make the formation controller **K**'s "job easier", more intuitive and also its structure more simple, we want each component of **Z** to affect

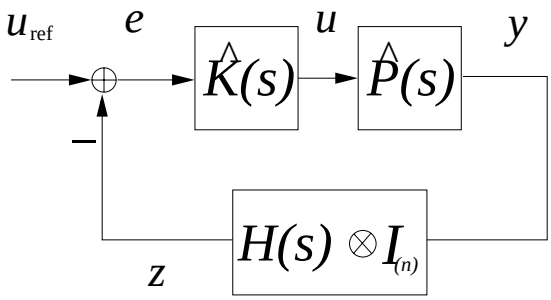

Figure 7: A general interconnected system.

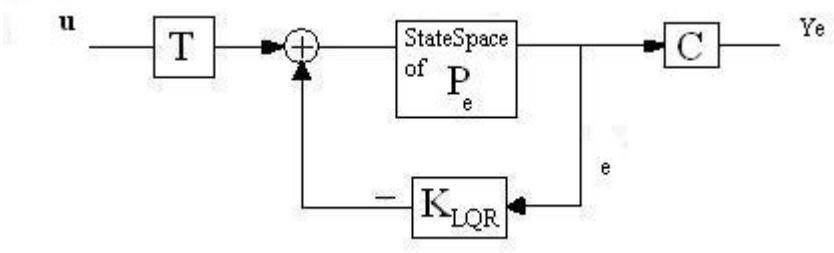

Figure 8: LQR stabilized plant.

only its corresponding component of $\mathbf{Y}$. Now if we look at figure 7, how do we accomplish this goal?
The way chosen here was to set $\mathbf{K}(s)$ on the form

$$\mathbf{K}(s) = \begin{bmatrix} f_1(s) & 0 \\ 0 & f_2(s) \end{bmatrix},$$

where f_1 and f_2 are any arbitrary transfer function, and $\mathbf{P}(0)$ on the form

$$\mathbf{P}(0) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \qquad (5)$$

This should guarantee that each component of $\mathbf{Z}$, at least in stationarity (after long time) only affects its corresponding component of $\mathbf{Y}$.
As a first step to achieve this, each plant described by equation (4) was stabilized with a LQR controller, $\mathbf{K_{LQR}}$, according to figure 8.

T is here calculated to have

$$\mathbf{Y_e} = \begin{bmatrix} r_e \\ \beta_e \end{bmatrix}$$

follow u statically and to have each component of $\mathbf{Y_e}$ only be affected by its corresponding component of u in stationarity (i.e. to satisfy (5)):

$$\mathbf{T} = \mathbf{I_2}(\mathbf{C}((-\mathbf{A} + \mathbf{BK_{LQR}})^{-1}\mathbf{B}))^{-1}$$

u is the output of the formation controller **K**.

The weight matrices used for the LQR were optimized for a radius of 2.0 m and an angular velocity of - 0.2 rad/s, see appendix B computeKlqr.

To conclude, the closed system from u to Ye is our **P** in section 3.1, see figure 2. It is straight forward calculations to calculate **P** because all the parts in figure 8 are known. The result is a 2 by 2 transfer matrix, (calculations carried out by Matlab):

P(s) =

```
Transfer function from input 1 to output...
             0.1414 s^4 + 1.253 s^3 + 6.065 s^2 + 6.665 s + 1.79
 #1:  ------------------------------------------------------------------
      s^6 + 8.891 s^5 + 26.06 s^4 + 37.26 s^3 + 28.84 s^2 + 11.49 s + 1.79

                  0.308 s^4 + 2.164 s^3 + 3.879 s^2 + 1.57 s
 #2:  ------------------------------------------------------------------
      s^6 + 8.891 s^5 + 26.06 s^4 + 37.26 s^3 + 28.84 s^2 + 11.49 s + 1.79

Transfer function from input 2 to output...
               0.1636 s^4 + 1.543 s^3 + 1.742 s^2 + 0.5532 s
 #1:  ------------------------------------------------------------------
      s^6 + 8.891 s^5 + 26.06 s^4 + 37.26 s^3 + 28.84 s^2 + 11.49 s + 1.79

             0.3564 s^4 + 2.708 s^3 + 5.285 s^2 + 4.829 s + 1.79
 #2:  ------------------------------------------------------------------
      s^6 + 8.891 s^5 + 26.06 s^4 + 37.26 s^3 + 28.84 s^2 + 11.49 s + 1.79
```

We right away see that this matrix fulfils

$$\mathbf{P}(0) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

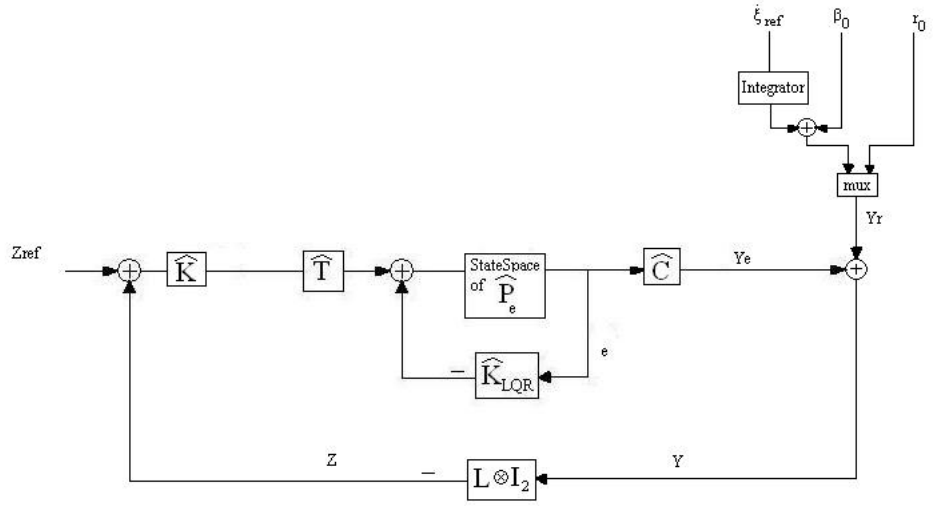

Figure 9: Our interconnected system.

4 FINDING A FORMATION CONTROLLER

Adding together all the different pieces from the previous section results in the block schematics for the interconnected system shown in figure 9.
The following section contains a description of the Simulink model used throughout this work. For this, a **K** is needed. I will here list the controller used in the final version:

$$\mathbf{K} = (1 + \frac{1}{2s} + s) \cdot \mathbf{I}_2$$

Simulations, stability and choice of **K** are treated in 4.2.

4.1 THE SIMULINK MODEL

The Simulink model SimForm and its two subsystems (VehicleSubsystem and Compute T) are on continuous form and are listed in figure 37-39 in appendix A.
The model is very similar to the block schematics of figure 9, on all but two points:
The first one being that the plant-, LQR-controller- and T-matrices vary with the reference radius and reference angular velocity in the Simulink model which gives a better approximation of the model in comparison to constant matrices that are only adequate at points close to the point of equilibrium.

Figure 10:

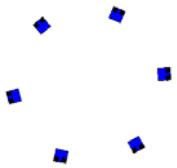

Figure 11:

The second point being the internal angle θ. To be able to reproduce the position and direction of each vehicle at every time in the animation of a vehicle formation (see chapter 5), in addition to r and β, also θ needs to be continually stored to Matlab workspace.
Three Matlab function blocks are used to compute the **A**-, **B**- and $\mathbf{K}_{\mathbf{LQR}}$-matrices at every sample point; ComputeA, ComputeB and ComputeKlqr. Their Matlab code is listed in appendix B.

4.2 SIMULATION AND STABILITY

Before a simulation can be executed, the m-file InitForm (appendix B) needs to be run. It sets start values for r, β and θ as well as regulator parameters. It also defines the interconnection matrix and the offset reference positions Z_{ref}.

4.2.1 Simulation Case 1: Hexagon Formation.

The initial conditions and Z_{ref} used for all the simulations in 4.2.1 are set to the values in InitForm.m simulation case 1 (appendix B). A view of the vehicles in starting position is shown in figure 10 and a view of the vehicles in the sought after position described by Z_{ref} is shown in figure 11 (hexagon).
Now to perhaps the most interesting part of this work, the choice of **K**. Remember what was said in 3.3.2; we want $\mathbf{K}(s)$ on the form

$$\mathbf{K}(s) = \begin{bmatrix} f_1(s) & 0 \\ 0 & f_2(s) \end{bmatrix},$$

where f_1 and f_2 are any arbitrary transfer function. The most logical **K** to start off with seemed to be what can be compared to the standard proportional controller in terms of a regular control system:

$$\begin{aligned} &\mathbf{K} = K \cdot \mathbf{I_2}, \\ &K = 1 \end{aligned} \qquad (6)$$

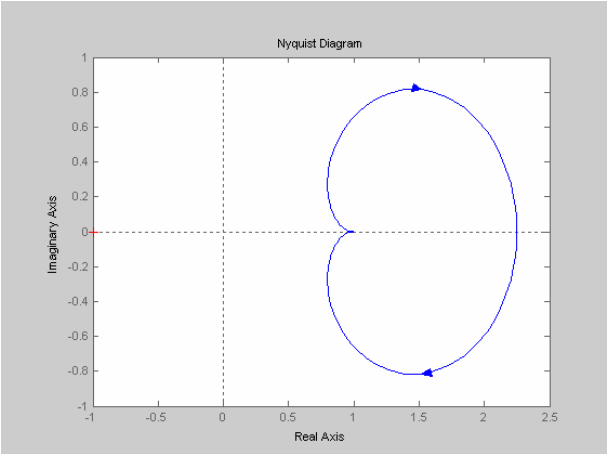

Figure 12: $\lambda = 0.5$

seemed natural to start off with.
To check for stability using this controller, theorem 1 is used. When checking the Nyquist diagram of

$$\det[\mathbf{I}_2 + \lambda_i(s)\mathbf{P(s)K(s)}],$$

P is the closed system transfer matrix from u to Ye, see figure 8. The values of λ_i were calculated in 3.2 and are repeated here:

$$\lambda_1 = 0,\ \lambda_{2,3} = 0.5,\ \lambda_{4,5} = 1.5,\ \lambda_6 = 2.$$

So in essence there are only three Nyquist plots needed to be checked:

$$\lambda = 0.5,\ \lambda = 1.5 \text{ and } \lambda = 2$$

for each simulation case**,** because there are two double poles. For us to have stability each of the plots has to have a summation of zero encirclements of the origin, according to (1).
The m-file that is plotting the Nyquist diagram is StabilTest.m see appendix B.

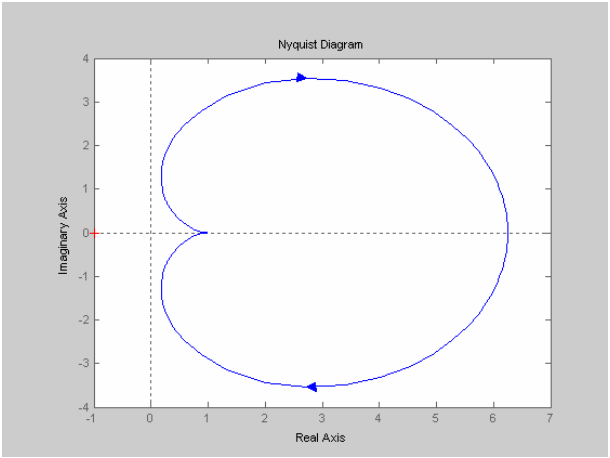

Figure 13: $\lambda = 1.5$

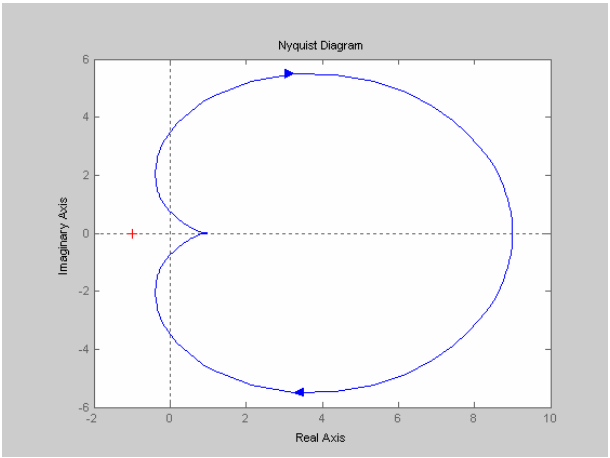

Figure 14: $\lambda = 2$

The plots in figure 12-14 show no encirclement of the origin and we have stability.

For each simulation case, the poles of the closed system transfer function

$$\left(\mathbf{I}_2 + \lambda_i(s)\mathbf{P(s)K(s)}\right)^{-1}\lambda_i(s)\mathbf{P(s)K(s)}$$

were calculated in Matlab for each λ_i. Of course all of the poles have to be in the left half plane for stability. Only the result will be listed, not the actual poles. This was done to complement and verify the stability check using (1).

Also worth mentioning is that plotting the Nyquist curves and calculating the poles required frequently usage of the Matlab commands zpk and minreal (see StabilTest.m).

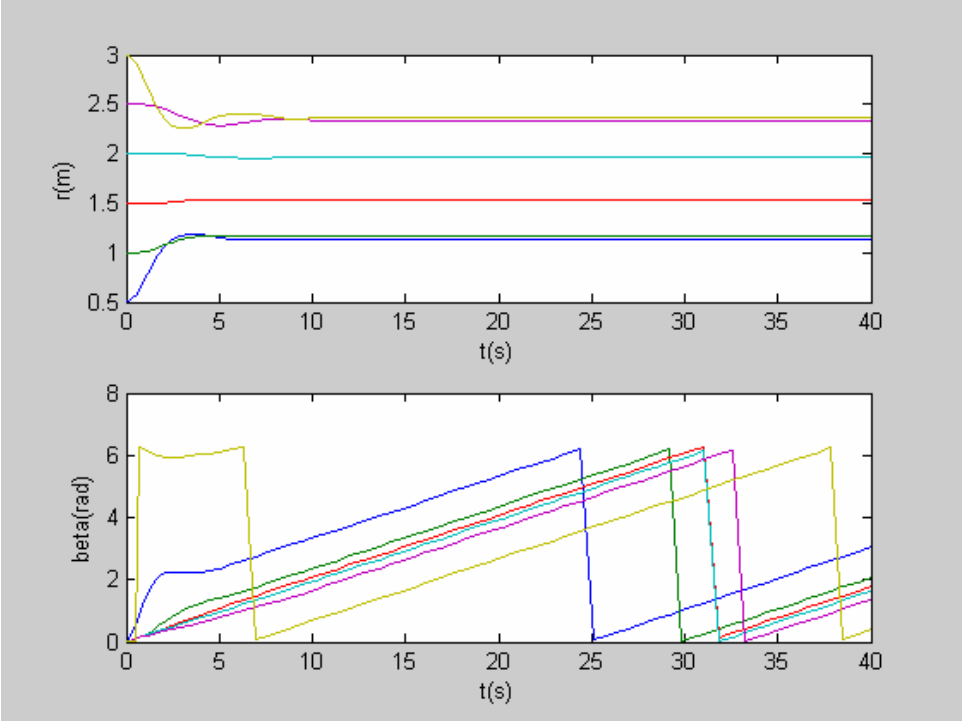

Figure 15: P

For the proportional controller (6), all of the poles were in the left half plane for each λ_i, which is in agreement with the Nyquist plots.

The result of a simulation using controller (6) is shown in figure 15. When analyzing the simulation results it helps to have figure 3 in mind, and to think of each Z_i described in polar coordinates (r, β).

The large vertical lines depend on the fact that β is modulated by 2π. The outcome is surprisingly good, considering the simplicity of the controller. Remember the goal is a hexagon, so we want equal r's and distributed, equally distanced β's.

There is indeed compensation in all of the r's and β's, while they however do not satisfy the stationary values set by Z_{ref}, i.e. there is in stationarity six different values of the r's instead of one, and non equally distanced β's.

What is interesting is that this fulfils a common characteristic of the P-controller of a regular system, i.e. some compensation but a stationary error.

Now the next step to take in a regular control system when you are using a P-controller but are getting stationary error is to add an integrator part to the controller to get rid of the error (PI). So it seems a natural thing to try on our system as well:

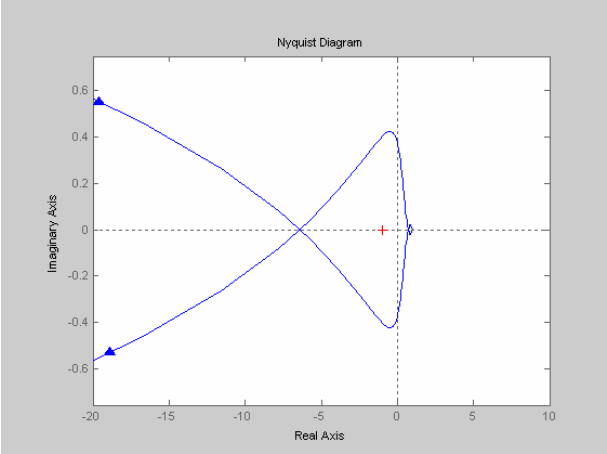

Figure 16: $\lambda = 0.5$

$$\mathbf{K} = K(1 + \frac{1}{T_i s}) \cdot \mathbf{I_2}$$

Using

$$K = 1, \qquad T_i = 1. \tag{7}$$

results in a Nyquist plot shown in figure 16 for

$$\det[\mathbf{I}_2 + \lambda_i(s)\mathbf{P(s)K(s)}],$$

where

$$\lambda = 0.5.$$

So what does this diagram tell us? There is a clear counter clockwise encirclement of the origin, which might lead one to conclude instability. However we don't see the whole picture. The Matlab command Nyquist is simply not good enough to show what happens to the curve for small frequencies. We need to investigate what the curve looks like outside of the diagram.
As we know, the Nyquist diagram is the result of mapping a curve *C* onto the actual transfer function, where *C* consists of a large semi-circle with infinite radius encircling the right half plane, a small semi-circle with infinitely small radius encircling the origin in the right half plane, and the positive and negative complex axes. Of these, the plot of the Nyquist command above shows the mapping of all parts of *C* but the small semi-

circle. So we need to look closer on how the small semi-circle is mapped. The expression

$$\det[\mathbf{I}_2 + \lambda_i(s)\mathbf{P(s)K(s)}]$$

was calculated in Matlab and can for the values it takes on for the small semi-circle be approximated by

$$\frac{J}{s^2},$$

where J is a constant. The small semi-circle mapped on this function would then look like

$$\frac{J}{r^2 e^{2i\varphi}}, \tag{8}$$

where r is the radius of the small semi-circle and φ its argument. Let

$$r \to 0,$$

$$\varphi : -\frac{\pi}{2} \to 0 \to \frac{\pi}{2}$$

Expression (8) is then a curve with infinite radius, and argument:

$$-2\varphi : \pi \to 0 \to -\pi$$

This corresponds to a circle with infinite radius directed in a clockwise manner. Applying this circle manually to figure 16 results in plot shown in figure 17.
This curve encircles the origin once CCW and once CW so that the total number of encirclements of the origin is zero, and so $\lambda = 0.5$ does not contribute to instability.

It turns out that the determinant

$$\det[\mathbf{I}_2 + \lambda_i(s)\mathbf{P(s)K(s)}]$$

can be described by

$$\frac{J}{s^2}$$

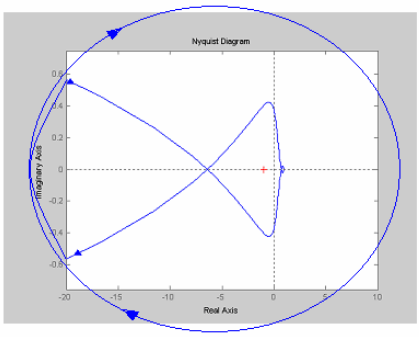

Figure 17: $\lambda = 0.5$

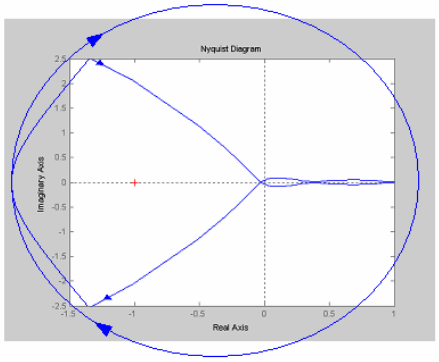

Figure 18: $\lambda = 1.5$

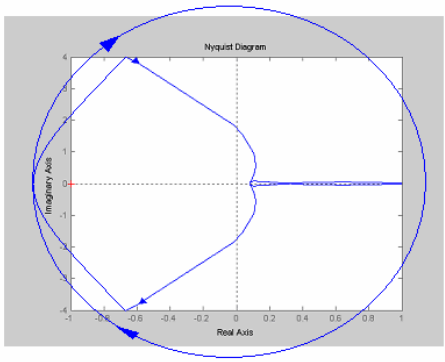

Figure 19: $\lambda = 2$

on the small semi-circle for all the remaining cases treated in this thesis. Applying the "manual semi-circle" method on the cases $\lambda = 1.5$ and $\lambda = 2$ yields the plots in figure 18 and 19.
The $\lambda = 1.5$ case shows a zero net encirclement of the origin, and does not contribute to instability.The $\lambda = 2$ case on the other hand encircles the origin a total number of twice in the CW direction, which means the system is unstable for the controller (7).
A check of the poles for the closed system using (7) shows four poles in the right half plane for $\lambda = 2$ and again we conclude: Using (7) yields an unstable system.
A simulation result using the controller (7) is shown in figure (20) for demonstrative reasons (knowing the result will be unstable).

We have to adjust our controller parameters, we try

$$K = 1,$$
$$T_i = 2. \tag{9}$$

Again using the "manual semi-circle" method gives us the Nyquist curves in figure 21-23.

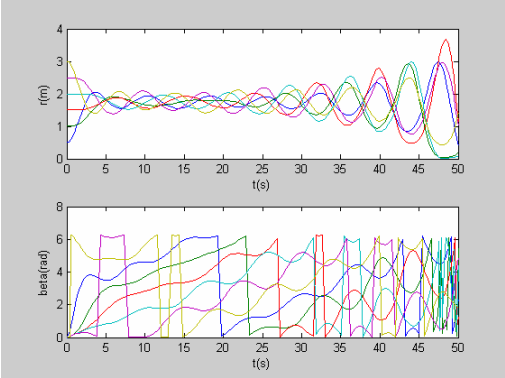

Figure 20: PI, $K = 1 \;\; T_i = 1$

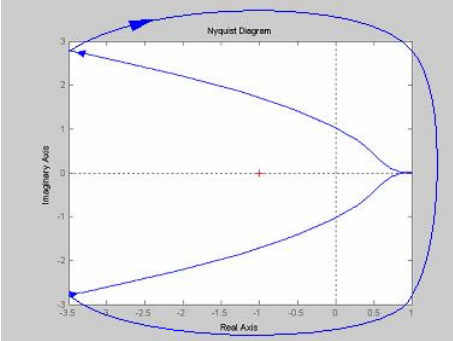

Figure 21: $\lambda = 0.5$

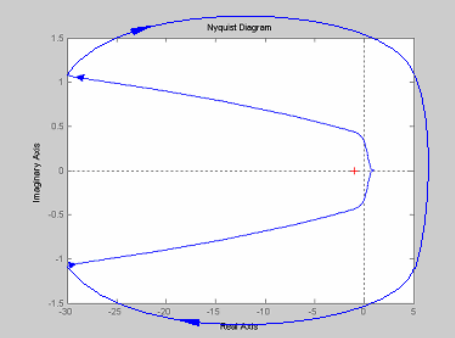

Figure 22: $\lambda = 1.5$

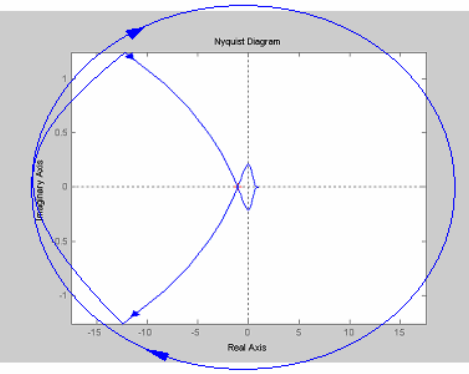

Figure 23: $\lambda = 2$

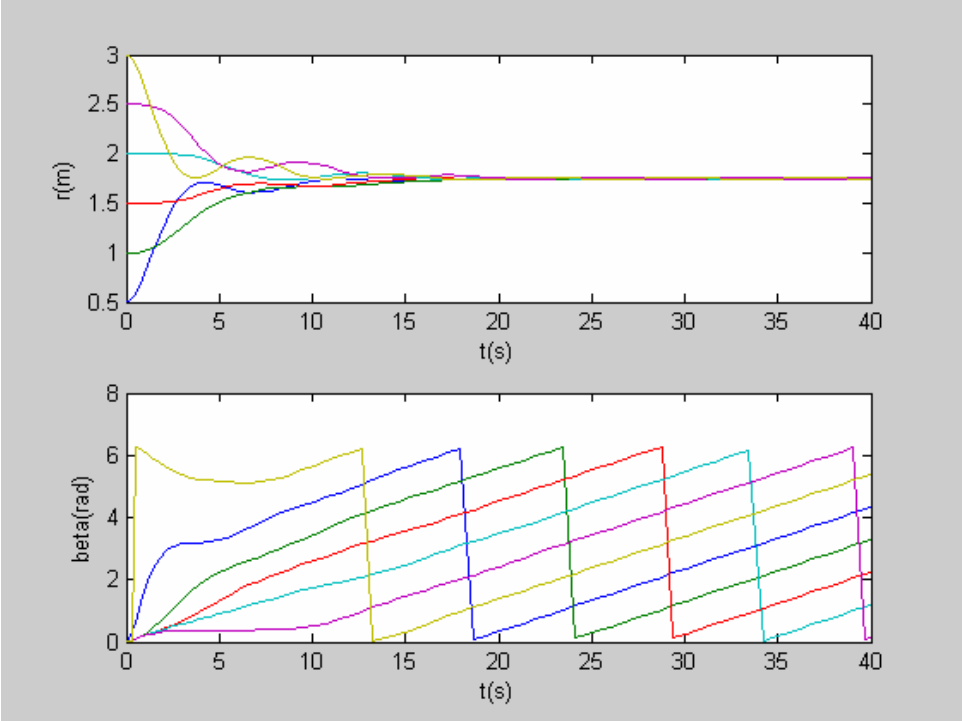

Figure 24: PI, $K = 1 \;\; T_i = 2$

As we can see none of these curves have any net encirclement of the origin and we have stability using the controller (9).
Calculating the poles for the closed system resulted in all poles being in the left half plane for all λ_i, which verifies the result of the Nyquist plots.
A simulation result using controller (9) is plotted in figure 24. The improvement is huge. The error is gone, we are however to some extent experiencing oscillations.

To get a more damped system the obvious choice is to add a derivative part to our controller:

$$\mathbf{K} = K(1 + \frac{1}{T_i s} + T_d s) \cdot \mathbf{I_2}$$

The derivative parameter T_d is set to 1 so that we have

$$\begin{aligned} K &= 1, \\ T_i &= 2, \\ T_d &= 1. \end{aligned} \qquad (10)$$

Looking at stability for this case, the Nyquist plots are shown in figure 25-27.

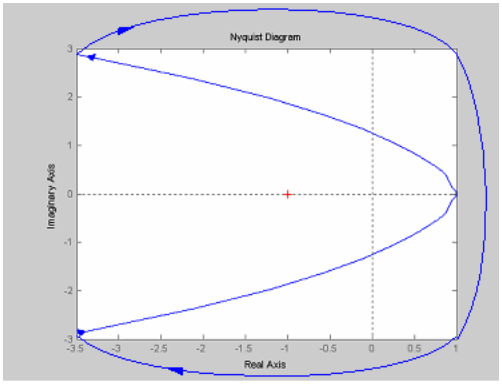

Figure 25: $\lambda = 0.5$

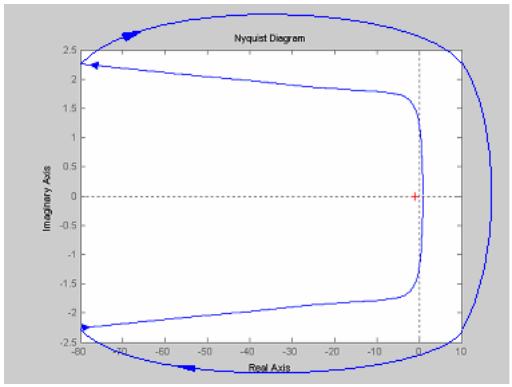

Figure 26: $\lambda = 1.5$

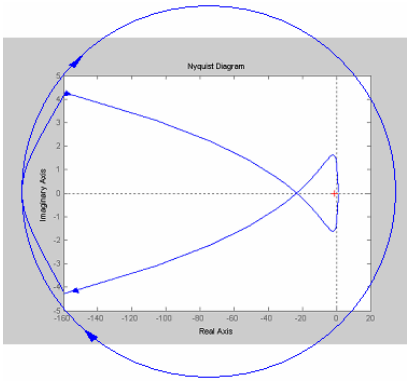

Figure 27: $\lambda = 2$

As we can see none of these curves have any net encirclement of the origin and we have stability using the controller (10).
Calculating the poles for the closed system resulted in all poles being in the left half plane for all λ_i, which verifies the result of the Nyquist plots.

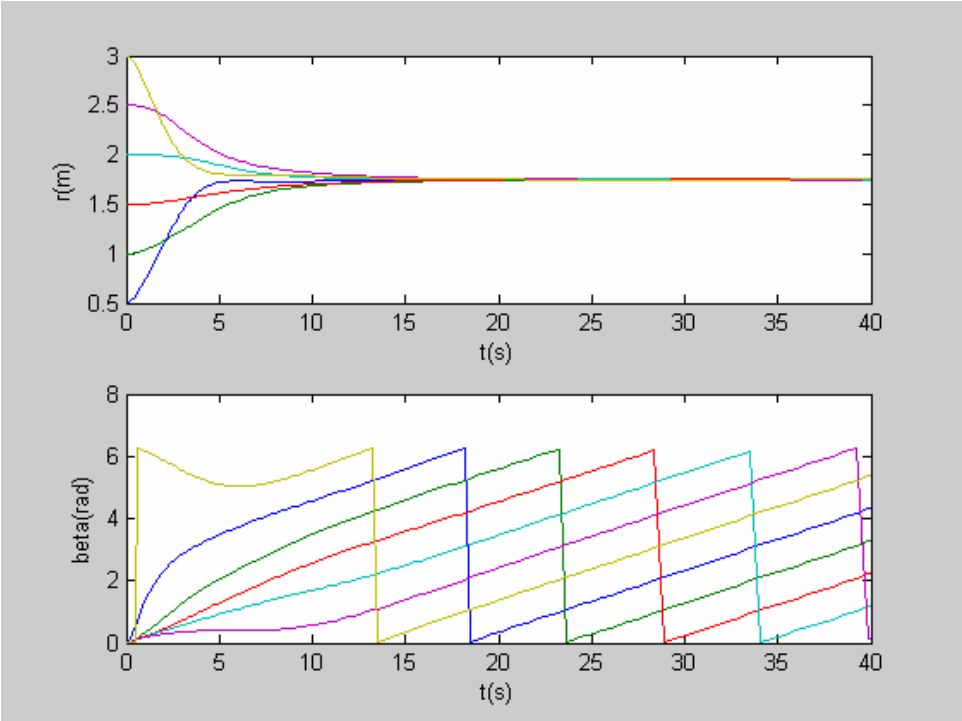

Figure 28: PID

A simulation result using controller (10) is plotted in figure 28. This plot is showing no large drawback. We have stability, the error is gone, we have a well damped and fairly fast system. Also the controller (10) was the best combination of parameters I was able to find for the PID case. The result is satisfying and we settle with this controller.

4.2.2 Simulation Case 2: "Hand of a Clock" Formation

A different problem in terms of initial conditions and Z_{ref} was considered:

Here we start where we finished off in the previous experiment, i.e. six vehicles spread out on a circle with equal angular distances. The sought after formation is here a hand of a clock (of course circulating the origin at a constant angular velocity $\dot{\xi}$) of vehicles. (See InitForm.m simulation case 2 appendix B.)

Using the PID regulator (10) (these controller parameters proved to be the best ones found also for this case) resulted in a simulation plot shown in figure 29.

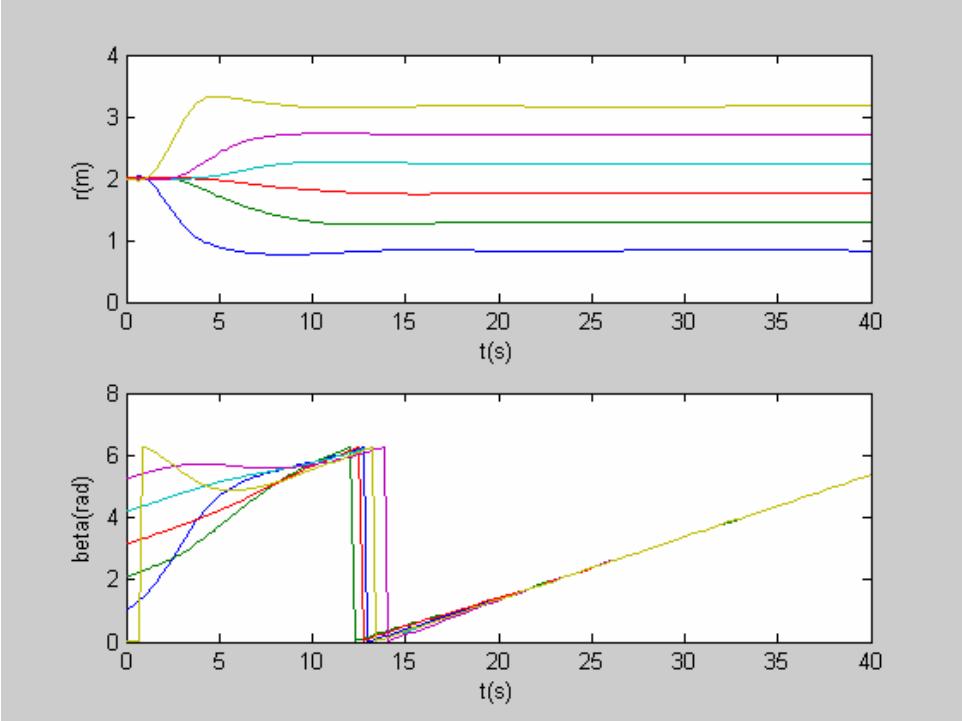

Figure 29: PID

Basically everything looks good; we have equally distributed r's and equal β's in stationarity.

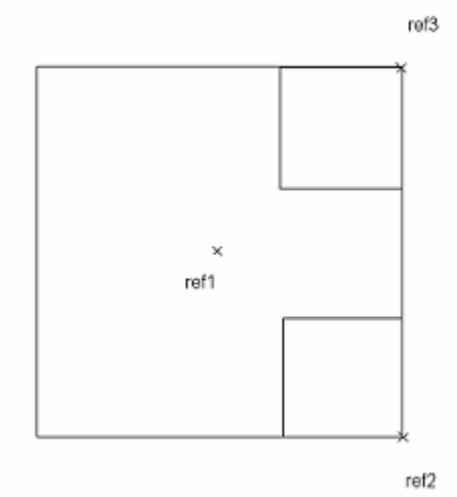

Figure 30: Reference positions on Kelly vehicle used in AnimForm.m

5 ANIMATING THE FORMATION DYNAMICS

The file AnimForm.m (see appendix B) generates an animation of a specific simulation. Each vehicle is modelled as a 0.4 m side square, with two small squares of diagonal 0.1 m in two of the corners of each vehicle to represent the two fans, see figure 30.
So how is it really done? Well the task was to make a film sequence of the vehicles changing formation. Simulink simulates this change of formation. Running the Simulink file SimForm.mdl stores each vehicle's position and angle to Matlab workspace at a number of sample points.
Now drawing all the vehicles at a given sample point gives us a momentary picture of the formation at that point in time. Doing this for all the sample points in a sequence, starting at the first sample point, renders a film clip or an animation of a simulation.
The actual drawing is done by the Matlab command *fill*. It draws the 2D-polygon specified by the corner coordinates that you feed as input parameters to the command. Using *fill* one time to draw the vehicle and two more times to draw its fans, then repeating the procedure for the other five vehicles we get the whole picture.

Here follows a description of the notation used in AnimForm.m:

The local variables *xref1(i,j)* and *yref1(i,j)* correspond to the x- and y-coordinates at sample point *i* for vehicle *j* at position *ref1*, while *theta(i,j)* denotes the internal angle at sample point *i* for vehicle *j*.
The positions *ref2* and *ref3* correspond to two of the corners of the vehicle according to figure 30.

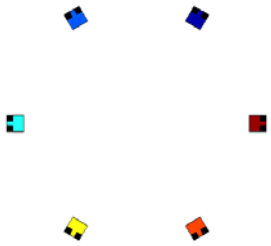

Figure 31: t = 0

Figure 32: t = 5

Figure 33: t = 10

Figure 34: t = 15

Figure 35: t =20

Figure 36: t = 25

"Before" and "after" shots of simulation case 1 taken from an animation using AnimForm.m is shown in figure 10 and figure 11.
A series of shots forming a sequence for simulation case 2, "Hand of a clock" is shown in figure 31-36.

6 RESULTS AND FUTURE WORK

6.1 RESULTS

A controller was found that renders a stabile system, performing well in simulations. This controller $\hat{\mathbf{K}}$, (see figure 4 for reference) looks like

$$\hat{\mathbf{K}} = \begin{bmatrix} \mathbf{K} & 0 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{K} & 0 & 0 & 0 & 0 \\ 0 & 0 & \mathbf{K} & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathbf{K} & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{K} & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbf{K} \end{bmatrix},$$

where each **K** represents a localized controller for each of the six vehicles:

$$\mathbf{K} = \begin{bmatrix} \mathbf{PID} & 0 \\ 0 & \mathbf{PID} \end{bmatrix},$$

where the first **PID** controls the radius of its vehicle relative the two neighbour vehicles radii, and the second **PID** controls the positional angle of its vehicle relative the two neighbour vehicles positional angles. Each **PID** in turn equals

$$\mathbf{PID} = K(1 + \frac{1}{T_i s} + T_d s),$$

where

$$K = 1,$$
$$T_i = 2,$$
$$T_d = 1.$$

6.2 FUTURE WORK

I can think of a number of different topics for future work. If we at first restrict ourselves to the MVWT, i.e. the vehicles on a testbed in this work, there are a few things that come to mind:
Simulating time delays, different kind of disturbances and motion different from circular. One can also try to make more accurate Simulink models, i.e. models that lie closer to reality. Building in collision avoidance would

be interesting. Theta (i.e. how each vehicle is directed around its own axis) controlling could be worth taking a look at, even though it is closely connected to beta (i.e. the angular position for each vehicle in a global coordinate system) in stationarity.
Looking in a broader perspective, expanding the model to 3D-space describing for instance unmanned air vehicles or satellite systems is of course interesting.
I think at this stage only your imagination sets a limit for what you can come up with for future work in this area.

APPENDIX A: SIMULINK MODEL

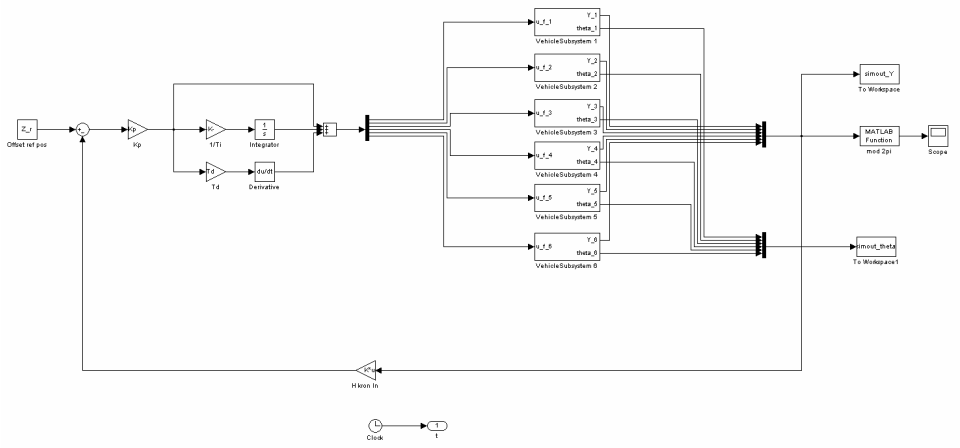

Figure 37: SimForm.mdl

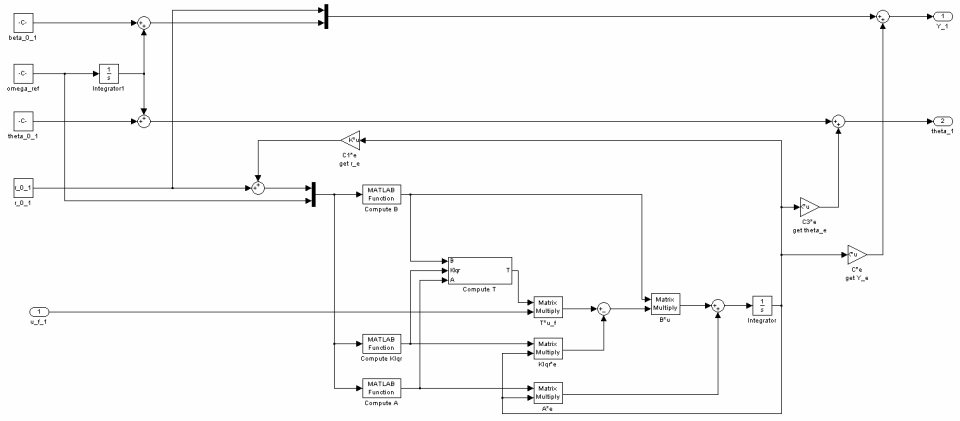

Figure 38: SimForm/Vehiclesubsystem 1

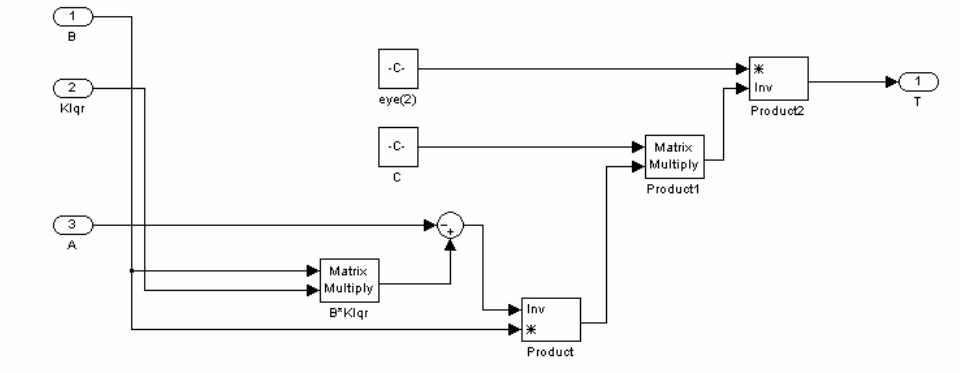

Figure 39: Simform/VehicleSubsystem 1/Compute T

APPENDIX B: M-FILES

The physical parameter values used in the m-files are taken from the file lqr_circles.m by L. Cremean and B. Dunbar at Caltech, California used to gainschedule a Kelly vehicle.

ComputeA.m:

```
function [sys] = ComputeA(u);

% physical parameters for new Kelly vehicle
r_c = 0.108;    % distance from center of mass to caster, m
m = 5.158;      % mass, kg
J = 0.050;      % rotational moment of inertia, kg*m^2
mu = 4.5;       % linear viscous friction coefficient, kg/s
psi = mu*r_c^2; % rotational viscous friction coefficient, kg*m^2/s
r_f = 0.123;    % lateral distance from ctr of mass to fan axis, m

rho = u(1);
xidot = u(2);

A = [zeros(3) eye(3);...
    [xidot^2   mu*rho*xidot/m -mu*rho*xidot/m; 0 xidot^2 -   xidot^2; 0 0
0]...
    [-mu/m 2*rho*xidot 0; -2*xidot/rho -mu/m 0; 0 0 -psi/J]];

sys = A;
```

ComputeB.m:

```
function [sys] = ComputeB(u);

% physical parameters for new Kelly vehicle
r_c = 0.108;    % distance from center of mass to caster, m
m = 5.158;      % mass, kg
J = 0.050;      % rotational moment of inertia, kg*m^2
mu = 4.5;       % linear viscous friction coefficient, kg/s
psi = mu*r_c^2; % rotational viscous friction coefficient, kg*m^2/s
r_f = 0.123;    % lateral distance from ctr of mass to fan axis, m

rho = u(1);
xidot = u(2);
alpha0 = atan(m*xidot/mu);

B = [zeros(3,2)
     -sign(xidot)*sin(alpha0)/m -sign(xidot)*sin(alpha0)/m
     -sign(xidot)*cos(alpha0)/rho/m -sign(xidot)*cos(alpha0)/rho/m
     r_f/J -r_f/J];

sys=B;
```

ComputeKlqr.m:

```
function [sys] = ComputeKlqr(u);

% physical parameters for new Kelly vehicle
r_c = 0.108;    % distance from center of mass to caster, m
m = 5.158;      % mass, kg
J = 0.050;      % rotational moment of inertia, kg*m^2
mu = 4.5;       % linear viscous friction coefficient, kg/s
psi = mu*r_c^2; % rotational viscous friction coefficient, kg*m^2/s
r_f = 0.123;    % lateral distance from ctr of mass to fan axis, m

rho = u(1);
xidot = u(2);
alpha0 = atan(m*xidot/mu);

% LQR design
Q = diag([10,10,3,4,4,2]);
R = diag([1,1]);
A = [zeros(3) eye(3);...
```

```
    [xidot^2   mu*rho*xidot/m -mu*rho*xidot/m; 0 xidot^2 -xidot^2; 0 0
0]...
    [-mu/m 2*rho*xidot 0; -2*xidot/rho -mu/m 0; 0 0 -psi/J]];
B = [zeros(3,2)
    -sign(xidot)*sin(alpha0)/m -sign(xidot)*sin(alpha0)/m
    -sign(xidot)*cos(alpha0)/rho/m -sign(xidot)*cos(alpha0)/rho/m
    r_f/J -r_f/J];

% Assume full state feedback - designed with rho = 2.0, xidot = -0.2.
[Klqr,S,E] = lqr(A,B,Q,R);

sys=Klqr;
```

InitForm.m simulation case 1:

```
% Initialize formation simulation
clear;

r_0_1 = 0.5;
r_0_2 = 1.0;
r_0_3 = 1.5;
r_0_4 = 2.0;
r_0_5 = 2.5;
r_0_6 = 3.0;

omega_ref = 0.2;

beta_0 = 0;

beta_0_1 = beta_0;
beta_0_2 = beta_0;
beta_0_3 = beta_0;
beta_0_4 = beta_0;
beta_0_5 = beta_0;
beta_0_6 = beta_0;

theta_0 = 0;

theta_0_1 = beta_0_1;
theta_0_2 = beta_0_2;
theta_0_3 = beta_0_3;
theta_0_4 = beta_0_4;
theta_0_5 = beta_0_5;
theta_0_6 = beta_0_6;

% Regulator parameters.
Kp=1;
Ti=2;
Td=1;

% Interconnection matrix. Each vehicle senses the distances to its
% two index-wise closest vehicles without any delay or disturbance
% (the Laplacian).
L=[1       -1/2       0          0        0       -1/2
   -1/2    1          -1/2       0        0       0
   0       -1/2       1          -1/2     0       0
   0       0          -1/2       1        -1/2    0
   0       0          0          -1/2     1       -1/2
   -1/2    0          0          0        -1/2    1];

% Compute H kronecker I2
HxIn=kron(L,eye(2));

% Parameter used in low pass filter 1/(1+w)
w=0.001;

% Offset reference position
Z_r=[0
    pi
    0
    0
    0
    0
    0
    0
    0
    0
    0
    -pi];
```

InitForm.m simulation case 2:

```
% Initialize formation simulation
clear;
r_0 = 2;

r_0_1 = 1.0*r_0;
r_0_2 = 1.0*r_0;
r_0_3 = 1.0*r_0;
r_0_4 = 1.0*r_0;
r_0_5 = 1.0*r_0;
r_0_6 = 1.0*r_0;

omega_ref = 0.2;

beta_0 = 0;

beta_0_1 = 1*2*pi/6 + beta_0;
beta_0_2 = 2*2*pi/6 + beta_0;
beta_0_3 = 3*2*pi/6 + beta_0;
beta_0_4 = 4*2*pi/6 + beta_0;
beta_0_5 = 5*2*pi/6 + beta_0;
beta_0_6 = 6*2*pi/6 + beta_0;

theta_0 = 0;

theta_0_1 = beta_0_1;
theta_0_2 = beta_0_2;
theta_0_3 = beta_0_3;
theta_0_4 = beta_0_4;
theta_0_5 = beta_0_5;
theta_0_6 = beta_0_6;

% Regulator parameters.
Kp=1;
Ti=2;
Td=1;

% Interconnection matrix. Each vehicle senses the distances to its
% two index-wise closest vehicles without any delay or disturbance
% (the Laplacian).
L=[1       -1/2       0          0        0       -1/2
   -1/2    1          -1/2       0        0       0
   0       -1/2       1          -1/2     0       0
   0       0          -1/2       1        -1/2    0
   0       0          0          -1/2     1       -1/2
   -1/2    0          0          0        -1/2    1];

% Compute H kronecker I2
HxIn=kron(L,eye(2));

% Offset reference position
Z_r=[-1.4
    0
    0
    0
    0
    0
    0
    0
    0
    0
    1.4
    0];
```

StabilTest.m:

```
clear;
% System parameters:

m=5.05;             % 5.05+-0.05 kg
rf=0.123;           % distance between center of mass of vehicle and fan
axis
J=0.050;            % 0.050+-0.005 kg m2
my=4.4;             % linear friction 3.3 .. 5.5 kg/s (different for each
vehicle)
                    % reference radius 0.1 .. 3
```

```
wr=0.2;             % reference angular velocity 0.1 .. 1 rad/s
ny=0.0565;          % rotational friction 0.049 .. 0.064 kg m2/s
rc=sqrt(ny/my);
a0=atan(m*wr/my);   % internal angle
rr=2;

% System matrices:

             A=[0       0           0             1        0        0
                0       0           0             0        1        0
                0       0           0             0        0        1
                wr^2    my*rr*wr/m -my*rr*wr/m   -my/m     2*rr*wr  0
                0       wr^2       -wr^2         -2*wr/rr -my/m     0
                0       0           0             0        0       -
rc^2*my/J];

             B=[                    0             0
                                    0             0
                                    0             0
                       sign(wr)/rr*[-rr*sin(a0)/m -rr*sin(a0)/m
                                   -cos(a0)/m    -cos(a0)/m   ]
                                    rf/J         -rf/J        ];

             C=[1       0           0             0        0        0
                0       1           0             0        0        0];

Q = diag([10,10,3,4,4,2]);
R = diag([1,1]);
[K,S,E]=lqr(A,B,Q,R); % LQR controller used to stabilize the individual
vehicles.

rank([B A*B (A^2)*B (A^3)*B (A^4)*B (A^5)*B]);
T=eye(2)/(C*((-A+B*K)\B));

G_cl=ss(A-B*K,B*T,C,0); % Reference radius och reference positional angle
are the inputs
P=zpk(G_cl);
PP=tf(G_cl);
s=zpk('s');
poles_of_stabilized_P=eig(G_cl);

Kf=[1+1/2/s+s 0
    0 1+1/2/s+s];
L=[1       -1/2       0           0        0       -1/2
   -1/2    1          -1/2        0        0       0
   0       -1/2       1           -1/2     0       0
   0       0          -1/2        1        -1/2    0
   0       0          0           -1/2     1       -1/2
   -1/2    0          0           0        -1/2    1];
kron(L,eye(2));
lambda=eig(L);

PKf=minreal(P*Kf);

% Checking stability:
temp=(eye(2)+lambda(6)*PKf);
temp2=minreal(temp\(lambda(6)*PKf),1e-2);
poles_of_intercon_sys=eig(temp2)
length(poles_of_intercon_sys);
determ=minreal(temp(1,1)*temp(2,2)-temp(1,2)*temp(2,1));
nyquist(determ);
```

AnimForm.m:

```
NmbrSamplePoints = length(simout_Y.signals.values(:,1));
xref1 = zeros(NmbrSamplePoints,6);
yref1 = zeros(NmbrSamplePoints,6);
theta = simout_theta.signals.values;

for i=1:6,
```

```
    xref1(:,i) = simout_Y.signals.values(:,2*i-
1).*cos(simout_Y.signals.values(:,2*i));
    yref1(:,i) = simout_Y.signals.values(:,2*i-
1).*sin(simout_Y.signals.values(:,2*i));
end

scale=max([max(max(xref1)) max(max(yref1))])+0.5;

for i=1:NmbrSamplePoints,
    clf;
    axis([-scale scale -scale scale]);
    hold on;
    set(gca,'position',[0 0 1 1],'visible','off','nextplot','add');

    % Draw vehicles
    for j=1:6,

        xref2 = xref1(i,j)+0.2*sin(theta(i,j)+pi/4);
        yref2 = yref1(i,j)-0.2*cos(theta(i,j)+pi/4);
        xref3 = xref1(i,j)+0.2*cos(theta(i,j)+pi/4);
        yref3 = yref1(i,j)+0.2*sin(theta(i,j)+pi/4);

        fill([xref1(i,j)-0.2*cos(theta(i,j)+pi/4) xref2 xref3 xref1(i,j)-
0.2*sin(theta(i,j)+pi/4)],...
             [yref1(i,j)-0.2*sin(theta(i,j)+pi/4) yref2 yref3
yref1(i,j)+0.2*cos(theta(i,j)+pi/4)],j); % Vehicle
        fill([xref2-0.1*cos(theta(i,j)) xref2 xref2-0.1*sin(theta(i,j))
xref2-0.1*sin(theta(i,j))-0.1*cos(theta(i,j))],...
             [yref2-0.1*sin(theta(i,j)) yref2 yref2+0.1*cos(theta(i,j))
yref2+0.1*cos(theta(i,j))-0.1*sin(theta(i,j))],'black'); % Left fan
        fill([xref3-0.1*cos(theta(i,j))+0.1*sin(theta(i,j))
xref3+0.1*sin(theta(i,j)) xref3 xref3-0.1*cos(theta(i,j))],...
             [yref3-0.1*cos(theta(i,j))-0.1*sin(theta(i,j)) yref3-
0.1*cos(theta(i,j)) yref3 yref3-0.1*sin(theta(i,j))],'black'); % Right fan

    end

    pause(0.2);

end
```

APPENDIX C: A MANUAL TO SIMULATIONS

Below follows a list of what you have to do to run a simulation followed by a movie sequence:

You first of all need to make sure that you have the following files in your Matlab directory:

AnimForm.m, ComputeA.m, ComputeB.m, ComputeKlqr.m, InitForm.m and SimForm.mdl.

Now type in a Matlab environment:

>> edit InitForm

Set the initial positions of the vehicles, the regulator parameters, the reference angular velocity and reference positions. Save. Type

>> InitForm

Open the Simulink model by typing

>> SimForm

Adjust the length of a simulation in the menue Simulation/Simulation parameters. Start a simulation by Simulation/Start. Open the Scope block to monitor the simulation. Wait until the simulation is finished.
To run a movie sequence type

>> AnimForm

REFERENCES

[1] Ather Gattami and Richard M. Murray. *A Frequency Domain Condition for Stability of Interconnected MIMO Systems*. In Proceedings of American Control Conference, 2004.

[2] J. A. Fax, R. M. Murray. *Graph Laplacians and Stabilization of Vehicle Formations.* 15th IFAC Congress, Barcelona, Spain, 2002.

[3] Lars Cremean, William Dunbar, David van Gogh, Jason Hickey, Eric Klavins, Jason Meltzer, Richard M. Murray. *The Caltech Multi-Vehicle Wireless Testbed.* 2002 Conference on Decision and Control (CDC).

[4] D. V. Gogh. *Development of Linearized error dynamics.* Unpublished paper.