**Konzept der Regelungstechnik**

**Sommersemester 2024**

# Projekt-Wettbewerb

## A State Feedback Controller for the Single-Track Model

*Henning Schlüter, M. Sc.*
*Prof. Dr.-Ing. Frank Allgöwer*

**Universität Stuttgart**
**Institut für**
**Systemtheorie und Regelungstechnik**

# Inhaltsverzeichnis

# 1 Motivation

## 1.1 About this Course

In the laboratory course *Concepts of Automatic Control*, you were meant to implement methods you had learned in lectures on a practical example, orienting on the steps for controller design proposed by Sigurd Skogestad [5]. In this prior course, you were supervised by student and teaching assistants, guiding you through the steps. This time, in contrast, you are supposed to make your way through the design procedure yourself. Because of time limitations, we chose to skip the modeling part and give you a completed model at hand. This model will also be the plant you will be working with. In particular, you will be working on the single-track model, which is well-known in literature on vehicle dynamics control systems [4, 2, 3, 1]. The model has been validated several times and therefore suites our purposes well.

## 1.2 Vehicle Dynamics Control Systems

Vehicle dynamics control systems have a long tradition in vehicle engineering. The first anti-lock braking system was engineered by Gabriel Voisin in the 1920s for airplanes and by Karl Wessel in 1928 for automobiles. The latter gave rise to the 1936 Bosch patent on the corresponding control system. With the advances in microcontroller design, the system became commercial in 1969. The first traction control system was developed by Ferdinand Porsche in 1939. Since these developments, the path for a rigorous control system for vehicle stability was paved; the later anti-lock braking system was yet capable of reducing torque around the yaw axis. Consequently, the first electronic stability control for automotives was engineered by Bosch in 1995. In these latter developments, model-based controller design and on-controller simulation of vehicle dynamics had significant roles. Thus, herein, we will ask you to develop a state feedback controller for an appropriate model of the vehicle dynamics at hand. This task is of practical interest particularly in the high-end automotive industry.

# 2 A Systematic Approach to Solve Control Problems

In the following, we present, explain and summarize a systematic approach to solve general control problems. This means, given a particular plant and a corresponding control task, the information given in this section should enable you to solve the task systematically.

The approach presented is based on [5] where a more detailed discussion on this issue can be found. The following list which has been taken from [5] directly summarizes the procedure.

1. **Study** the plant and obtain initial information about the control objectives.
2. **Model** the system and simplify the model, if necessary.
3. **Analyze** the resulting **model**; determine its properties.
4. **Decide** which variables are to be controlled (**controlled outputs**).
5. **Decide** on the **measurements and manipulated variables**; what sensors and actuators will be used and where will they be placed.
6. **Select** the **control configuration**.
7. **Decide** on the **type of controller** to be used.
8. **Decide** on **performance specifications**, based on the overall control objectives.
9. **Design** a controller.
10. **Analyze** the resulting controlled system to see if the specifications are satisfied; if not, modify the specifications or the type of controller.
11. **Simulate** the resulting **controlled system** on a computer and if possible on a pilot plant.
12. **Repeat** from step 2, if necessary.
13. Choose hardware and software and **implement** the controller.
14. **Test** and validate the control system; tune the controller (online), if necessary.

The whole procedure is illustrated in Figure 2.1. Further, Figure 2.1 shows the different layers of abstraction which are used while solving the task. Starting with the real plant, a detailed mathematical model is derived, analyzed and simplified. The simplified model then is used for controller design. The designed controller is tested in simulations using the simple/design model in a first step and the detailed/simulation model in a second step. Finally, the controller is implemented on the real plant.

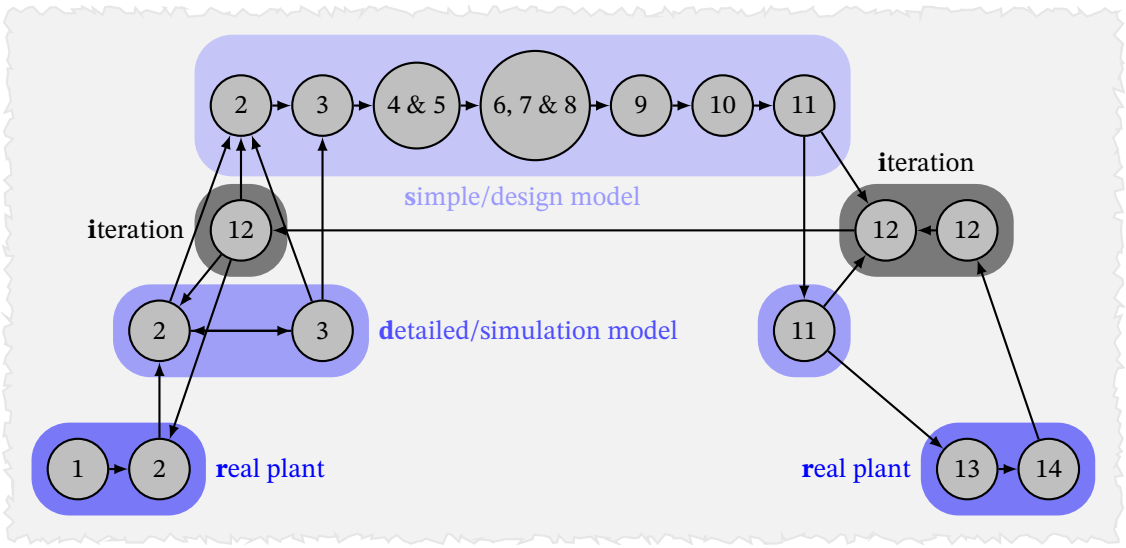


Abbildung 2.1: A systematic approach to solve control problems.

1. **Study the plant and obtain initial information about the control objectives:** In this very first step, the goal is to obtain as much (preliminary) information as possible about the system and how it works. Gathering information can be done, e.g., by reading technical manuals, performing a literature review, consulting colleagues and experts or by just playing with the system and its components. The latter is particularly helpful to get a good intuition which - besides hard facts, of course - often helps you to solve the problem. Finally, before starting the design procedure, you should think about how the main task can be approached, what the overall control objectives are and what kind of constraints and requirements this might imply on the controller. Usually, this kind of questions cannot be answered at this very early stage. However, keeping them in the back of your mind throughout the whole design procedure prevents you from following the wrong track and avoids blind spots.

2. **Model the system and simplify the model, if necessary:** The goal of the modelling step is to derive a mathematical model of the system – usually in terms of ordinary differential equations. In order to do so, you can use first principles and model the system according to the laws of physics. When doing so, the parameters of the system model directly represent physical properties. This often allows to directly measure these parameters. If it is not possible to measure all the parameters directly, parameter identification can be a useful tool. A different way to model a dynamical system is to postulate a particular structure for the system behavior (e.g., PT2) and determine numerical values of the parameters within this structure via some identification procedure. No matter how it is done, modelling is a very important step in the procedure of a systematic control design as all the steps following are based on the model derived in this step. Obviously, the most advanced control design is worthless if the design model does not represent the real system appropriately. Thus, this fact immediately leads to the first requirement of a useful model which is high accuracy. However, deriving a model with high accuracy may demand too much effort. Consequently, the modelling process always includes a trade-off between accuracy and modelling effort.

Further, in many cases a very accurate model is also a model with high complexity – which cannot always be dealt with. Thus, the second requirement of a useful model is to keep the complexity of the model in an acceptable range. What an acceptable range of complexity is depends very much on the purpose which the model is used for. For controller design for example, fast dynamics, delays, nonlinearities, constraints, and discontinuities are typical effects which can or have to be neglected in the design model. However, as these effects might influence the behavior of the system considerably, the use of a more accurate system model for simulation purposes can be very beneficial in the control design procedure. Thus, it often makes sense to derive a quite accurate simulation model and to simplify it to a design model by neglecting or approximating effects which cannot be coped with in the controller design.

3. **Analyze the resulting model; determine its properties:** In this step, you analyze system properties of interest. This can be done by simply running simulations representing scenarios of interest. However, calculating system theoretical properties based on the mathematical model is in general more helpful and important for the following steps in the design procedure. Typically, properties of interest are steady states, the dynamic behavior, in particular stability as well as controllability and observability (after defining inputs and outputs, cf., following steps). Further, e.g., system structure, robustness properties and the influence of disturbances might give useful insight. Finally, for any obtained property you should check whether the derived results are in agreement with the observable physical properties of the system and your intuition. If this is not the case, your model might be erroneous, oversimplified or involves nonexisting phenomena.

4. **Decide which variables are to be controlled (controlled outputs):** Decide which variables you want to control. The controlled variables should be chosen such that it is (easily) possible to formulate the overall control objectives in terms of these variables. Further, you should consider that the choice of the outputs (in combination with the inputs) defines not only the input-output behavior but also the internal dynamics. For the internal dynamics you desire them to be stable or sometimes even nonexistent.

5. **Decide on the measurements (measured outputs) and manipulated variables (control inputs); what sensors and actuators will be used and where will they be placed:** In this step you decide how the system is connected to the outside world. For each variable you want to measure you need to place a sensor at the system. For each variable you want to manipulate you need to place an actuator at the system. Sometimes you will have very little freedom in this decision as you are to work with an already completely designed system setup and have to deal with the system as it is. However, in other cases you can influence how many and which kind of sensors and actuators are to be implemented. In this case, this is a very important step in the design procedure as it strongly influences the achievable performance. Fortunately, a sound system theoretic analysis of the plant in particular on observability and controllability, can give precious advises on a good decision. Of course, this decision is not only influenced by the system theoretic analysis, but to the same extent, by physical and economical considerations. Finally, note that in general measured and controlled outputs are not required to coincide – although this is a very popular case at least for illustrative issues (cf., e.g., standard control loop).

6. **Select the control configuration:** Select a control configuration, e.g., the standard control loop. In general, the control configuration defines which blocks (besides obvious ones like plant and controller, e.g., measuring filter, decoupling, disturbance compensation, feedforward, or a prefilter) are used in the control loop and how they are connected among each other and with the plant. The connections between the blocks are realized by signals like, e.g., reference, control input, measured and controlled outputs. The control configuration is typically described by a block diagram. In the following design steps, it is desirable to keep to the structure of the control configuration, e.g., by mimicking it in your Simulink model.

7. **Decide on the type of controller to be used:** Decide which type of controller you use. First, you decide whether you do the controller design in frequency or in time domain. Then, you specify the type of the controller and a corresponding design method like, e.g., P-/PI-/PID-controller or state-feedback in combination with an observer done, e.g., by pole placement or LQR design. Depending on how strict „type of controller“ is interpreted this is either a whole family of controllers or only one particular structure with an already fixed number of parameters (in the first interpretation P- and PI-controllers are in the same family, in the second they are already different types of controllers).

8. **Decide on performance specifications, based on the overall control objectives:** Reformulate the verbally defined overall control objectives mathematically as definite requirements on the closed loop. Note that the requirements on the closed loop implicitly impose requirements on the controller. Hence, you have to take care of formulating the requirements such that there exists a realizable controller. Further, the performance specifications have to be considered when the parameters of the controller are determined and tuned in the next step. Thus, the mathematical performance specification should be formulated such that they fit to the type of controller and design method chosen in the previous step. For example, if you have chosen a frequency-domain method you should define the performance specifications in frequency domain, e.g., as requirements on the bandwidth of the closed loop.

9. **Design a controller:** In this step you determine the parameters of the controller such that the performance specifications can be met. This step is strongly connected to the analysis in the next step. Thus, the two steps are typically performed synchronously by an immediate calculation and visualization of the system properties of interest.

10. **Analyze the resulting controlled system to see if the specifications are satisfied; if not, modify the specifications or the type of controller:** In this step, you utilize different methods to analyze the open as well as the closed loop, e.g., Bode and Nyquist plots or the step response of the system. This step is typically considered as the central step of controller design. However, which performance you are able to achieve strongly depends on your decisions in the previous steps. Thus, iterating the design procedure and revising some of your decisions might become necessary to achieve the overall control objectives.

11. **Simulate the resulting controlled system on a computer and if possible on a pilot plant:** Before implementing your controller on the physical system, you should test it in simulations. For these simulations, it makes sense to use models simultaneously increasing complexity and accuracy. Thus, you start with the design model to verify your design by

just checking whether the controlled system behaves like expected. If this is the case, you use a more accurate simulation model to check whether your controller can cope with the effects neglected in the design model and whether the performance degradation caused by these effects are still within an acceptable range. Finally, as a good simulation model mimics the connection between plant and controller, this helps you to avoid a wrong wiring and often momentous sign errors.

12. **Repeat from Step 2, if necessary:** Sometimes even a systematic and profound design does not meet the overall control objectives at first try. So you have to start again at the beginning. However, after the first try you have studied the system quite intensive and often have a perception of what went wrong and how you can improve the design process. Thus, the „initial information“ for the second try is much better which will guide you to, e.g., a modelling including previously unmodelled but relevant effects or to a better choice of actuators/sensors.

13. **Choose hardware and software and implement the controller:** In this step, you have to decide how you realize your controller and connect it with the plant. Sometimes, the hardware and software might be predetermined and cannot be changed. In this case, possible limitations like available memory and computational power as well as the achievable sampling time have to be considered earlier in the design process in particular in the choice of the controller and its specifications. If there is still the possibility to choose the hardware and the software, besides the system theoretic requirements and wishes you have to consider physical and economical aspects (similar as in Step **5.**). As these aspects often imply, considerable limitations it might be beneficial to already have them in mind in earlier design steps. Finally, we note that, when using digital devices, your controller has to run in discrete-time. As the design process is often done in continuous-time and the conversion from continuous to discrete is nontrivial and error-prone you should take care of it.

14. **Test and validate the control system; tune the controller (online), if necessary:** In the final step, you test your controller at the real system. In general, you should start with easy and slow tasks and maneuvers in combination with mildly tuned controllers to avoid putting yourself or the system setup at risk. Then, you can increase the complexity of the tasks and the velocity of maneuvers as well as the aggressiveness of the control. However, in any case, only implement controllers with parameters you have checked before by simulation.

# 3 Organization

## 3.1 Structure of the Course

In the laboratory course *Concepts of Automatic Control*, student and teaching assistants were guiding you through the steps described in Chapter 2. Also, the handbook provided a time-schedule and a specific goal for every particular element of the schedule. This time, the organizational part will be on you, cf., there are fixed dates for the start and end of the course; yet, everything in between can be scheduled by you. However, at the end of the course, certain elements will be mandatory to complete the course. Among them will be a report that you will have to write. We will discuss these in detail in Chapter 7. Other objectives will not be mandatory, but instead be measured by a certain cost functional. Minimizing this functional is the competitive part of the course, cf., the group that achieves the lowest functional wins the contest. The functional will also be discussed in Chapter 7. The remainder of this handbook provides the model (and therefore the plant) you will be working on, and also the control objective. The corresponding MATLAB-files, that are frequently referred to herein, are deposited in the respective ILIAS-folder.

## 3.2 Important dates

**02.05.24** Kick-off lecture via Webex and upload of the course material to ILIAS. (The kick-off lecture presents the idea of the course, the model, and the control objective. A recording of the lecture will remain online for the rest of the semester.)

**05.05.24, 23:55** C@MPUS registration and poll for group arrangement closes, both are mandatory for participation.

**04.06.24** register for the exam on C@MPUS, only register if you already passed „KRT Praktikum“.

**05.05.24 – 23.06.24** Competition. (Questions can be asked and will be answered in the ILIAS-Forum. In special cases, appointments with the course assistant can be arranged.)

**23.06.24** Hand-in. (You will have to upload the elements you are asked to complete (as described in Chapter 7) until 23.06.24 at 23:55 in order to complete the course.)

**23.06.24 – 01.07.24** Evaluation. (The course assistants evaluate your controllers and reports.)

**01.07.24** Closing lecture in presence. (We present the results of some selected groups, tell you who passed the course and who won the competition.)

# 4 The Single-Track Model

The single-track model is a dynamical system widely used to study the behavior of automobiles, e.g., during the design phase to iterate parameters, or for implementation of observers on micro-controllers in the serial vehicle [4, 2, 3, 1]. It is a simplified model of the more complex twin-track model where each axis (front and rear) carries only one wheel, each. Therein, the differential equations

$$\dot{x} = \upsilon \cos(\psi - \beta) \tag{4.1}$$

$$\dot{y} = \upsilon \sin(\psi - \beta) \tag{4.2}$$

are describing the behavior of the vehicles longitudinal and lateral velocity $\dot{x}$ and $\dot{y}$, respectively, as function of the yaw angle $\psi$ and the side-slip angle $\beta$. The yaw angle describes the actual orientation of the vehicle with respect to its initial configuration (or with reference to the inertial coordinate system) and the side-slip angle is the angle enclosed between the vehicles longitudinal axis and its velocity $\upsilon = \sqrt{\dot{x}^2 + \dot{y}^2}$, which has the derivative

$$\dot{\upsilon} = \frac{1}{m}\Big(F_{x,\mathrm{r}} \cos(\beta) + F_{x,\mathrm{f}} \cos(\delta + \beta) - F_{y,\mathrm{r}} \sin(\beta) - F_{y,\mathrm{f}} \sin(\delta + \beta)\Big), \tag{4.3}$$

with $m$ the vehicle mass and $F_{i,j}$ forces, where $i = x$ indicates that the force is directed longitudinally, $i = y$ indicates that the force is directed laterally, $j = r$ indicates that the force acts on the rear wheel and $j =$f indicates that the force acts on the front wheel. The above forces are given by the formula

$$F_{x,\mathrm{f}} = -\operatorname{sign}(\upsilon \cos(\beta))(1 - \zeta)F_{\mathrm{b}} - \operatorname{sign}(\upsilon \cos(\beta))\mu(\upsilon)\frac{mgl_{\mathrm{r}}}{l} \tag{4.4}$$

$$F_{x,\mathrm{r}} = \frac{1}{R} i(G) i_0 T_M(\phi, G) - \operatorname{sign}(\upsilon \cos(\beta))\zeta F_{\mathrm{b}} - \operatorname{sign}(\upsilon \cos(\beta))\mu(\upsilon)\frac{mgl_{\mathrm{f}}}{l} \tag{4.5}$$

$$F_{y,\mathrm{f}} = D_{\mathrm{f}} \sin\Big(C_{\mathrm{f}} \arctan\big(B_{\mathrm{f}}\alpha_{\mathrm{f}} - E_{\mathrm{f}}(B_{\mathrm{f}}\alpha_{\mathrm{f}} - \arctan(B_{\mathrm{f}}\alpha_{\mathrm{f}}))\big)\Big) \tag{4.6}$$

$$F_{y,\mathrm{r}} = D_{\mathrm{r}} \sin\Big(C_{\mathrm{r}} \arctan\big(B_{\mathrm{r}}\alpha_{\mathrm{r}} - E_{\mathrm{r}}(B_{\mathrm{r}}\alpha_{\mathrm{r}} - \arctan(B_{\mathrm{r}}\alpha_{\mathrm{r}}))\big)\Big), \tag{4.7}$$

where the equations for the lateral tire forces (4.6) and (4.7) are the Pacejka tire model, which is a heuristic model for lateral tire forces. Therein, the parameters $B_j$, $C_j$, $D_j$, $E_j$, are usually obtained from measurements and subsequent curve fitting, whereas $\alpha_{\mathrm{f}}$ and $\alpha_{\mathrm{r}}$ are the rear slip-angle and the front-slip-angle, respectively, given by

$$\alpha_{\mathrm{f}} = \delta - \arctan\left(\frac{l_{\mathrm{f}}\dot{\psi} - \upsilon \sin\beta}{\upsilon \cos\beta}\right), \qquad \alpha_{\mathrm{r}} = \arctan\left(\frac{l_{\mathrm{h}}\dot{\psi} + \upsilon \sin\beta}{\upsilon \cos\beta}\right). \tag{4.8}$$

In formula (4.4) and (4.5), which are obtained from the *actio est reactio* axiom, $mg$ is the gravitational force of the vehicle, scaled approximately by the distribution of mass given by the ratios $l_{\mathrm{f}}/l$ and $l_{\mathrm{r}}/l$, where $l$ is the length of the car and $l_{\mathrm{f}}$, $l_{\mathrm{r}}$ are the lengths from the center of mass to

the front and rear end of the car, respectively. Further, $\mu(v)$ is the velocity-dependent friction, determining the influence of the normal force on the movement. Also, $F_\mathrm{b}$ is the braking force, distributed between the axes by $\zeta$ (determined by valve positions in the brake fluid circuit), and $T_M$ is the motor torque at pedal position $\phi$ and gear $G$, scaled by the wheel radius $R$ and the transmissions of the motor $i_0$ and the gearbox $i$, which is itself dependent on the gear. Usually, the motor torque is given through measurements and then deposited in a look-up table, and the friction is a discontinuous function. Here, both are approximated heuristically by

$$T_\mathrm{M}(\phi, G) = 200\phi(15 - 14\phi)\left(1 - \frac{\left(\frac{30}{\pi}n(G)\right)^{5\phi}}{4800^{5\phi}}\right) \tag{4.9}$$

$$\mu(v) = r_0 + r_1|v| + r_2|v|^2 + r_3|v|^3 + r_4|v|^4\,, \tag{4.10}$$

where $n$ is the rotary frequency of the motor, given through

$$n(G) = \frac{vi(G)i_0}{R - RS}\,, \tag{4.11}$$

where $S$ is the wheel slip

$$S = \begin{cases} \frac{R\dot\varphi - v}{R\dot\varphi} & \text{if} v \le R\dot\varphi \\ \frac{v - R\dot\varphi}{v} & \text{else.} \end{cases} \tag{4.12}$$

We will however assume $S = 0$ at any time. In the latter, $\dot\varphi$ is the wheel rotary frequency and is obtained from the differential equation

$$\ddot\varphi = \frac{RF_{x,\mathrm{r}}}{I_R}\,. \tag{4.13}$$

Furthermore, $\psi$ and $\beta$ in (4.1), (4.2) are obtained from

$$\dot\beta = \omega - \frac{1}{mv}\big(F_{x,\mathrm{r}}\sin(\beta) + F_{x,\mathrm{f}}\sin(\delta+\beta) + F_{y,\mathrm{r}}\cos(\beta) + F_{y,\mathrm{f}}\cos(\delta+\beta)\big) \tag{4.14}$$

$$\dot\psi = \omega \tag{4.15}$$

$$\dot\omega = \frac{1}{I_z}\big(F_{y,\mathrm{f}}l_\mathrm{f}\cos(\delta) - F_{y,\mathrm{r}}l_\mathrm{r} + F_{x,\mathrm{f}}l_\mathrm{f}\sin(\delta)\big) \tag{4.16}$$

and the second-order dynamics are given by

$$\begin{aligned}\ddot x = \frac{1}{m}\Big(&\big(F_{x,\mathrm{r}} + F_{x,\mathrm{f}}\cos(\delta) - F_{y,\mathrm{f}}\sin(\delta)\big)\cos(\psi) - \\ &\big(F_{y,\mathrm{r}} + F_{x,\mathrm{f}}\sin(\delta) + F_{y,\mathrm{f}}\cos(\delta)\big)\sin(\psi)\Big)\end{aligned} \tag{4.17}$$

$$\begin{aligned}\ddot y = \frac{1}{m}\Big(&\big(F_{x,\mathrm{r}} + F_{x,\mathrm{f}}\cos(\delta) - F_{y,\mathrm{f}}\sin(\delta)\big)\sin(\psi) + \\ &\big(F_{y,\mathrm{r}} + F_{x,\mathrm{f}}\sin(\delta) + F_{y,\mathrm{f}}\cos(\delta)\big)\cos(\psi)\Big)\end{aligned} \tag{4.18}$$

$$\ddot\psi = \frac{1}{I_z}\big(F_{y,\mathrm{f}}l_\mathrm{f}\cos(\delta) - F_{y,\mathrm{r}}l_\mathrm{r} + F_{x,\mathrm{f}}l_\mathrm{f}\sin(\delta)\big)\,, \tag{4.19}$$

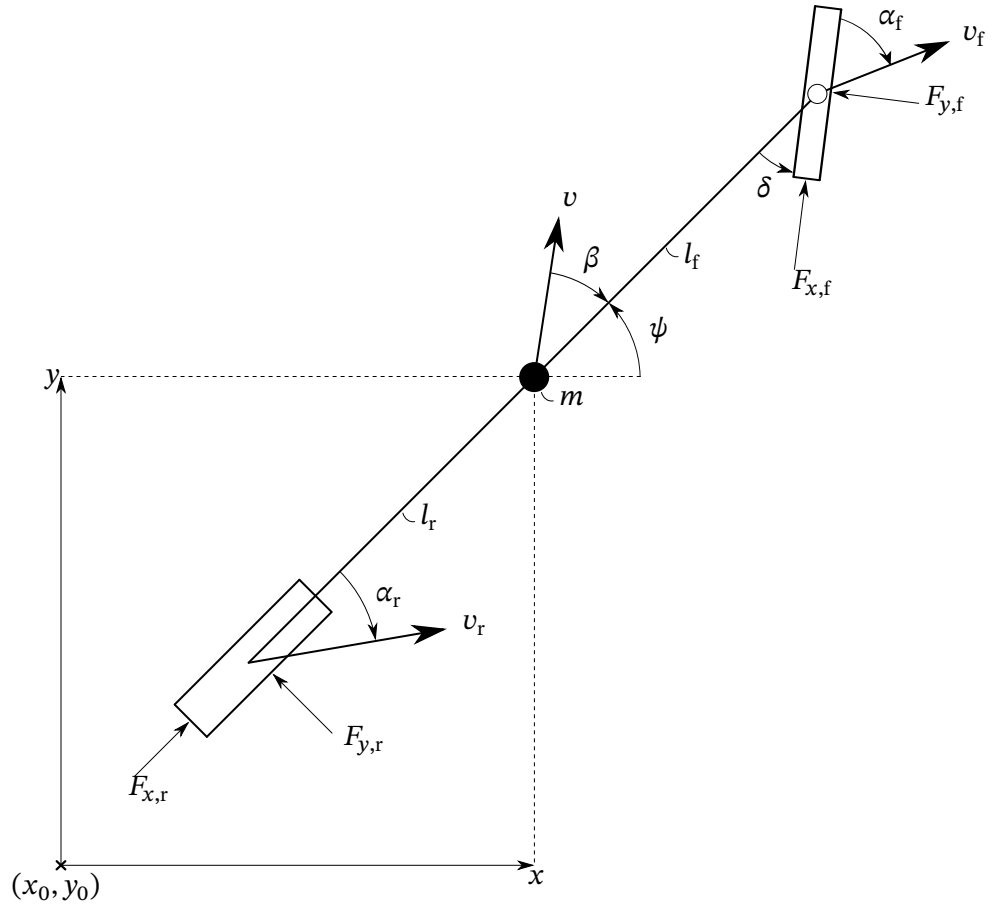


Abbildung 4.1: Single-Track Model

so that the model is composed of the differential equations (4.1)-(4.3) and (4.13)-(4.19) with states $x$, $y$, $v$, $\beta$, $\psi$, $\omega$, $\dot{x}$, $\dot{y}$, $\dot{\psi}$, $\dot{\varphi}$. Writing this as

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{v} \\ \dot{\beta} \\ \dot{\psi} \\ \dot{\omega} \\ \ddot{x} \\ \ddot{y} \\ \ddot{\psi} \\ \ddot{\varphi} \end{bmatrix} = f\left( \begin{bmatrix} x \\ y \\ v \\ \beta \\ \psi \\ \omega \\ \dot{x} \\ \dot{y} \\ \dot{\psi} \\ \dot{\varphi} \end{bmatrix}, \begin{bmatrix} \delta \\ G \\ F_b \\ \zeta \\ \phi \end{bmatrix} \right), \tag{4.20}$$

the MATLAB-function `singletrack.m` returns the value of $f(\cdot)$ for the respective values of $x$, $y$, $v$, $\beta$, $\psi$, $\omega$, $\dot{x}$, $\dot{y}$, $\dot{\psi}$, $\dot{\varphi}$. The input variables are $\delta$, $G$, $F_b$, $\zeta$, and $\phi$. In Table 4.1, the variables are listed for your convenience, together with their respective values. Also, we provide a sketch of the most important variables in Figure 4.1.

Tabelle 4.1: Single-Track Model Variables and Notation

| states | name | initial value | MATLAB variable |
|---|---|---|---|
| $x$ | longitudinal position | -2.5 | `x` |
| $y$ | lateral position | 0 | `y` |
| $v$ | velocity | 0 | `v` |
| $\beta$ | side-slip angle | 0 | `beta` |
| $\psi$ | yaw angle | $\frac{\pi}{2}$ | `psi` |
| $\omega$ | yaw rate | 0 | `omega` |
| $\dot{x}$ | longitudinal velocity | 0 | `x_dot` |
| $\dot{y}$ | lateral velocity | 0 | `y_dot` |
| $\dot{\psi}$ | yaw rate (redundant) | 0 | `psi_dot` |
| $\dot{\varphi}$ | wheel rotary frequency | 0 | `varphi_dot` |

| forces and others | name |
|---|---|
| $F_{x,\mathrm{r}}$ | longitudinal rear force |
| $F_{x,\mathrm{f}}$ | longitudinal front force |
| $F_{y,\mathrm{r}}$ | lateral rear force |
| $F_{y,\mathrm{f}}$ | lateral front force |
| $T_{\mathrm{M}}$ | motor torque |
| $\mu$ | friction |
| $S$ | wheel slip |
| $n$ | rotary frequency of the motor |

| inputs | name | allowed values | MATLAB variable |
|---|---|---|---|
| $\delta$ | steering angle | $\lvert\delta\rvert \leq 0.53$ | `delta` |
| $G$ | gear | $G \in \{1,2,3,4,5\}$ | `G` |
| $F_{\mathrm{b}}$ | braking force | $0 \leq F_{\mathrm{b}} \leq 15000$ | `F_b` |
| $\zeta$ | braking force distribution | $0 \leq \zeta \leq 1$ | `zeta` |
| $\phi$ | gas pedal position | $0 \leq \phi \leq 1$ | `phi` |

| parameters | name | value |
|---|---|---|
| $m$ | mass | 1239 |
| $g$ | gravity constant | 9.81 |
| $l_{\mathrm{f}}$ | front length | 1.19016 |
| $l_{\mathrm{r}}$ | rear length | 1.37484 |
| $R$ | wheel radius | 0.302 |
| $I_z$ | vehicle moment of inertia | 1752 |
| $I_R$ | wheel moment of inertia | 1.5 |
| $i(1)$ | first gear transmission | 3.91 |
| $i(2)$ | second gear transmission | 2.002 |
| $i(3)$ | third gear transmission | 1.33 |
| $i(4)$ | fourth gear transmission | 1 |
| $i(5)$ | fifth gear transmission | 0.805 |
| $i_0$ | motor transmission | 3.91 |
| $B_{\mathrm{f}}$ | stiffness factor | 10.96 |
| $C_{\mathrm{f}}$ | shape factor | 1.3 |
| $D_{\mathrm{f}}$ | peak value | 4560.4 |
| $E_{\mathrm{f}}$ | curvature factor | -0.5 |
| $B_{\mathrm{r}}$ | stiffness factor | 12.67 |
| $C_{\mathrm{r}}$ | shape factor | 1.3 |
| $D_{\mathrm{r}}$ | peak value | 3947.81 |
| $E_{\mathrm{r}}$ | curvature factor | -0.5 |
| $p_0$ | coefficient | -37.8 |
| $p_1$ | coefficient | 1.54 |
| $p_2$ | coefficient | -0.0019 |
| $q_0$ | coefficient | -34.9 |
| $q_1$ | coefficient | -0.04775 |
| $r_0$ | coefficient | 0.009 |
| $r_1$ | coefficient | $7.2(10^{-5})$ |
| $r_2$ | coefficient | 0 |
| $r_3$ | coefficient | 0 |
| $r_4$ | coefficient | $5.0388(10^{-10})$ |

# 5 Control Objective

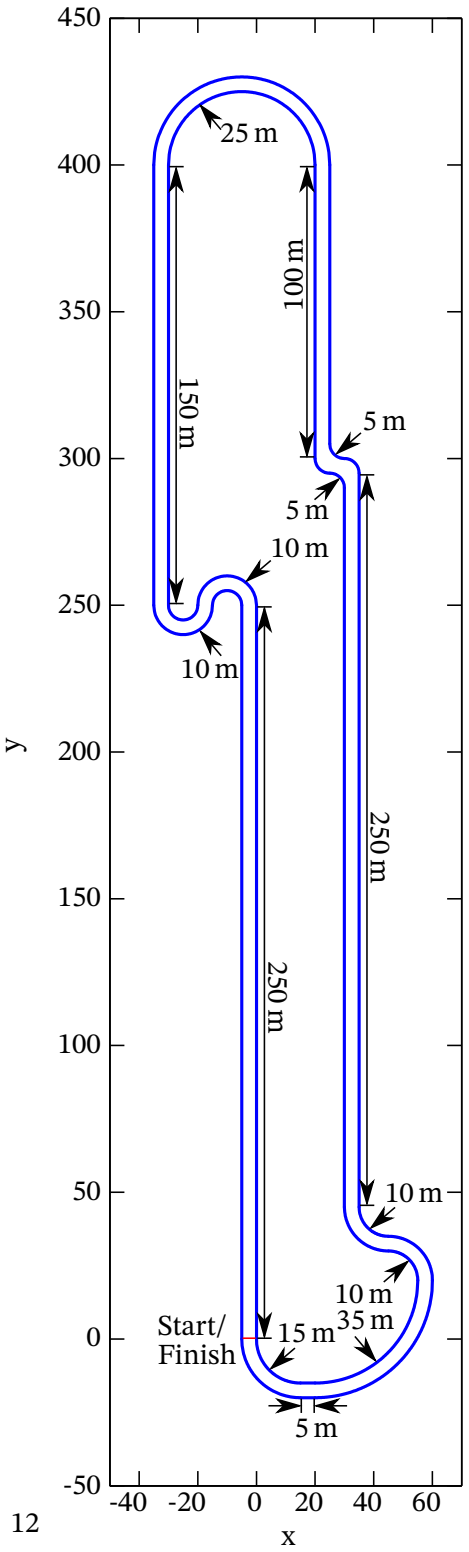


Abbildung 5.1: Racetrack

You are asked to design a state feedback controller for the single-track model. In particular, you ought to construct a state feedback controller such that the single-track model performs a lap around the racetrack depicted in Figure 5.1. We assume that your controller has full knowledge of the racetrack as well as full knowledge about the states of the system. Thus, if the racetrack data is given by the vectors $t_\mathrm{r}$ (inner racetrack boundary) and $t_\mathrm{l}$ (outer racetrack boundary), construct a function $K$ of the form

$$\begin{bmatrix} \delta \\ G \\ F_\mathrm{b} \\ \zeta \\ \phi \end{bmatrix} = K\left(\delta^*, G^*, F_\mathrm{b}^*, \zeta^*, \varphi^*, x, y, v, \beta, \psi, \omega, \dot{x}, \dot{y}, \dot{\psi}, \dot{\varphi}, t_\mathrm{r}, t_\mathrm{l}\right) \tag{5.1}$$

such that the single-track model completes the racetrack. In doing so, the vehicle must not leave the racetrack, cf., stay within the boundaries given by $t_\mathrm{r}$ and $t_\mathrm{l}$. This control problem is often called an autonomous driving problem (with known environment). Note that you neither have to construct an observer for your state feedback nor have to construct an estimator for the racetrack boundaries, as we assume to be able to measure all states as well as the racetrack boundaries perfectly.

Denoting the time at which your vehicle passes the finishing line of the track by $t_\mathrm{f}$, the team with the lowest value for $t_\mathrm{f}$ wins the competition.

In order to simulate the resulting closed-loop, you will have to implement the function $K$ in MATLAB.

# 6 Software Architecture

To simulate the single-track model, to implement your controller, and to check its performance on the racetrack, we use MATLAB. For your convenience, the racetrack as well as the single-track model are implemented in appropriate `*.m` files already. Using these files, your closed-loop will look as depicted in Figure 6.1. Therein, we use the MATLAB variables

```
X=[x;y;v;beta;psi;omega;x_dot;y_dot;psi_dot;varphi_dot]
X_dot=[x_dot;y_dot;v_dot;beta_dot;psi_dot;omega_dot; ...
       x_dot_dot;y_dot_dot;psi_dot_dot;varphi_dot_dot]
U=[delta;G;F_b;zeta;phi]
```

consistently with the notation in the `*.m` files (compare Table 4.1).

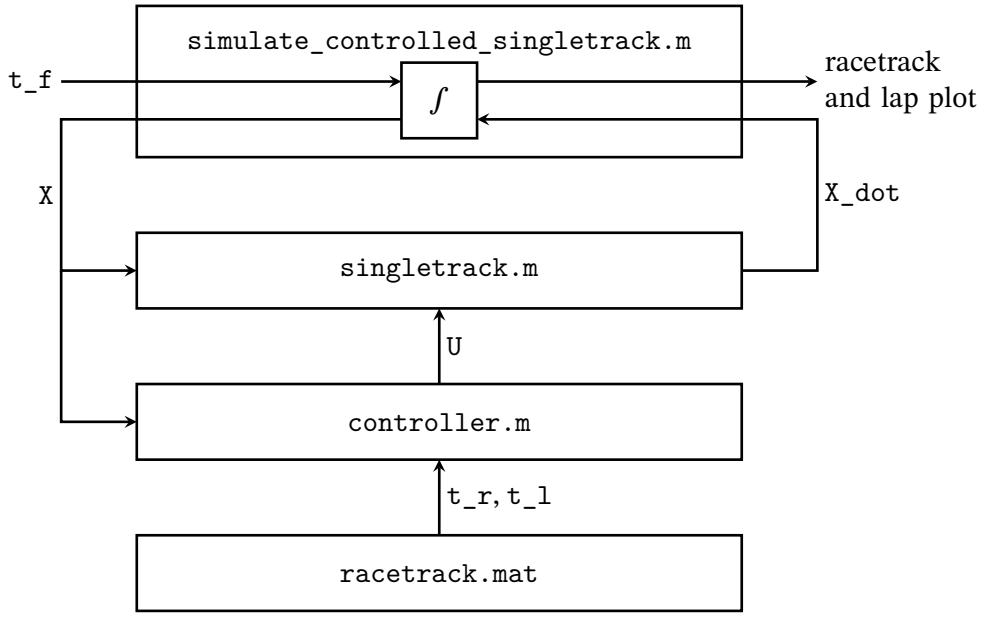


Abbildung 6.1: Software Architecture for the Simulation of the Single-Track Model

As depicted, the function `simulate_controlled_singletrack.m` has the simulation time `t_f` as input and returns a plot of the racetrack together with the `x`, `y` data resulting from the simulation of the closed loop. It calls the integrator `ode1.m`, which itself calls the function `singletrack.m`, cf., the vector field $f$ of the single-track model, and integrates it until time `t_f`. Meanwhile, `singletrack.m` calls `controller.m`, which is the MATLAB implementation of the funtion $K$, receiving the state of the single-track model as its input, but also being able to access the racetrack data `t_r` and `t_l` (cf., $t_\mathrm{r}$ and $t_\mathrm{l}$, respectively) from `racetrack.mat` in order to return the input variables of the single-track model.

In addition, the function `simulate_controlled_singletrack.m` calls the auxiliary files `racetrack.m`, creating the racetrack, and `plot_racetrack.m`, creating the plot of the racetrack together with the `x`, `y` data resulting from the simulation of the closed loop. All files are necessary for the simulation to run properly and can be found on ILIAS.

Thus, provided that the function `controller.m` is defined, the program receives the simulation time `t_f` as input and returns a plot of the racetrack together with a plot of the `x`, `y` data resulting from the simulation of the closed loop such that one can check the controller performance. The file `controller.m` is a raw function, cf., it has its inputs and outputs defined, but not its content. In particular, you will have to program this function and provide an appropriate value for `t_f`.

**Only change the file `controller.m`. All other files have to remain unchanged in order for your controller to work properly during the evaluation.**

All `*.m` files include comments from their author. Feel free to check the source codes of all `*.m` files for your own interest, but also for a more sound understanding of the simulation. If you have questions or comments on the code, feel free to contact the course advisor.

# 7 What is Needed to Complete the Course

You will need to submit

- an implementation of the state feedback controller $K$ in form of the function `controller.m`,
- a LaTeX document describing how you approached the problem and how you constructed $K$,
- a numerical value for the simulation time `t_f`, cf., $t_f$, the time that the single-track model takes to complete one lap of the racetrack with your state feedback controller (you can round up to the next integer value here; if necessary, the precise values as characterized in Chapter 5 will be determined for the three fastest teams),
- an approximate value for the simulation duration $t_{sim}$ that your computer needs to simulate the controlled single-track model

by 23.06.24 at 23:55 in the ILIAS exercise **Abgabe**.

Your controller needs to fulfill the following conditions:

- complete one lap in less than 180 $s$, cf., $t_f < 180\,\mathrm{s}$, without leaving the racetrack,
- not give any output in the MATLAB Command Window,
- not require any other files provided by you.

# Literaturverzeichnis


[1] R. N. Jazar. *Vehicle Dynamics*. Springer, 2009.

[2] M. Mitschke. *Dynamik der Kraftfahrzeuge*. Springer, 1990.

[3] M. Mitschke and Wallentowitz H. *Dynamik der Kraftfahrzeuge*. Springer, 2004.

[4] D. Schramm, M. Hiller, and R. Bardini. *Modellbildung und Simulation der Dynamik von Kraftfahrzeugen*. Springer, 2010.

[5] S. Skogestad and I. Postlethwaite. *Multivariable Feedback Control: Analysis and Design*. Wiley, 2005.