A Virtual Motorcycle Rider Based on Automatic Controller Design

Thomas Schmitt
Vorarlberg Univ. of Appl. Sc.
Austria
Thomas.Schmitt@students.fhv.at

Dirk Zimmer
ETH Zürich
Switzerland
DZimmer@Inf.ETHZ.CH

François E. Cellier
ETH Zürich
Switzerland
FCellier@Inf.ETHZ.CH

Abstract

This paper introduces a new and freely available *Modelica* library for the purpose of simulation, analysis and control of bicycles and motorcycles (single-track vehicles). The library is called *MotorcycleLib* and focuses on the modeling of virtual riders based on automatic controller design.

For the single-track vehicles, several models of different complexity have been developed. To validate these models and their driving performance, virtual riders are provided. The main task of a virtual rider is to track either a roll angle profile or a pre-defined trajectory using path-preview information. Both methods are implemented and several test tracks are also included in the library.

Keywords: virtual rider; automatic controller design; state-space controller, bicycle and motorcycle modeling; pole placement

1 Introduction

Among the vehicle models, models of bicycles and motorcycles turn out to be particularly delicate. Whereas a four-wheeled vehicle remains stable on its own, the same does not hold true for a single-track (two-wheeled) vehicle. For this reason, the stabilization of such a vehicle, a control issue, requires special attention.

A key task for a virtual rider is to stabilize the vehicle. To this end, a controller has to generate a suitable steering torque based on the feedback of appropriate state variables of the vehicle (e.g. lean angle and lean rate). One major problem in controlling single-track vehicles is that the coefficients of the controller are strongly velocity dependent. This makes the manual configuration of a controller laborious and error-prone. To overcome this problem, an automatic calculation of the controller's coefficients is desired. This calculation requires an eigenvalue analysis of the corresponding uncontrolled vehicle which is performed in order to determine the self-stabilizing area. The library includes the means for such an analysis and its results can be interpreted by three different modes that qualitatively describe the vehicle's motion [12]. This enables a convenient controller design and hence several control laws that ensure a stable driving behavior are provided. The corresponding output represents a state feedback matrix that can be directly applied to ready-made controllers which are the core of virtual riders. The functionality of this method is illustrated by several examples in the library.

In 2006, F. Donida et al. introduced the first Motorcycle Dynamics Library in Modelica [5] and [4]. The library focuses on the tire/road interaction. Moreover different virtual riders (rigidly attached to the main frame or with an additional degree of freedom (d.o.f.) allowing the rider to lean sideways) capable of tracking a roll angle and a target speed profile are presented. Until now these virtual riders include fixed structure controllers only [4]. This means that virtual rider stabilizes the vehicle only correctly within a small velocity range.

Using the automatic controller design functions provided by the *MotorcycleLib* this major deficiency can be overcome. Furthermore, to validate the motorcycle's performance, the virtual rider is capable of either tracking a roll angle profile (open-loop method) or a pre-defined path (closed-loop method).

2 Bicycle and Motorcycle Models

The mathematical modeling of single-track vehicles is a challenging task which covers a wide range of models of varying complexity. The library provides several single-track vehicle models of different complexity. The models are composed of multibody el-

ements and are based on bond graphs [2] and multi-bond graphs [17]. Basically two types of models are provided. Some include out-of-plane modes only, while others include both in-plane and out-of-plane modes. Roughly speaking, out-of-plane modes are related to stability and handling of single-track vehicles whereas in-plane modes are dealing with riding comfort. The wheels used in this library are either provided by D. Zimmer's *MultiBondLib* [17] or by M. Andres' *WheelsAndTires* library [1]. The former are ideal, whereas in the latter models non-ideal effects such as slip can be considered. For the bicycle, both 3 and 4 d.o.f. out-of-plane mode models are included in the library. The former are composed of four rigid bodies, namely a front frame, a rear frame including a rigidly attached rider and two wheels, connected via revolute joints. The wheels are infinitesimally thin (knife-edge). The latter introduce an additional d.o.f. that allows the rider's upper body to lean sideways. Both models are based on those introduced by Schwab et al. [12] and [11].

The out-of-plane mode motorcycle model is a 4 d.o.f. model that is based on a model established by V. Cossalter [3]. Basically, V. Cossalter's model is the same as the one introduced by R. S. Sharp in 1971 [13]. This model allows a lateral displacement of the rear frame since the wheels are no longer ideal. Due to the fact that the wheels of D. Zimmer's *MultiBondLib* [17] are ideal, the model is reduced to 3 d.o.f.Later, with reference to the *WheelsAndTires* library [1], it is possible to consider non-ideal effects of wheels and tires and thus simulate the lateral displacement of the wheels caused by tire slip. The animation of a 3 d.o.f. motorcycle is depicted in Figure 1. To incorporate in-

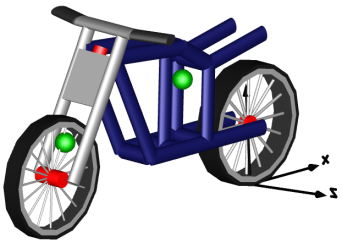

Figure 1: Animation of a 3 d.o.f. motorcycle model

plane modes two more complex models are included in the library. The first model was originally developed by C. Koenen during his Ph.D. Thesis [9]. R. S. Sharp and D. J. N. Limebeer introduced the SL2001 model which is based on Koenen's model [15]. They reproduced Koenen's model as accurately as possible and described it by means of multibodies. The model developed in this library is based on the SL2001 motorcycle. The second model is based on an improved more state-of-the-art version of the former developed by R. Sharp, S. Evangelou and D. J. N. Limebeer [14]. A very detailed description of these models can be found in S. Evangelou's Ph.D. Thesis [6]. Such models are composed of a front frame including the front forks and handle bar assembly, a rear frame including the lower rigid body of the rider, a swinging arm including the rear suspensions, the rider's upper body, a front and a rear wheel. Furthermore several additional freedoms due to twist frame flexibility at the steering head, suspensions, non-ideal tire models and aerodynamics are taken into account. The animation of the SL2001 model is depicted in Figure 2. In con-

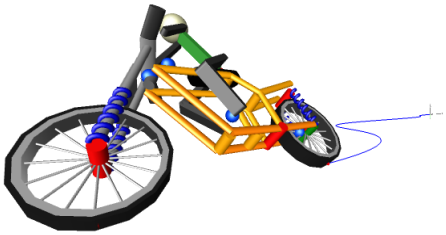

Figure 2: Animation of the SL2001 motorcycle model

trast to the former models each body is created in a fully object-oriented fashion. As with the out-of-plane models these models only include all degrees of freedom in combination with the *WheelsAndTires* library. Without this library several freedoms are inhibited.

It is important to keep in mind that vehicles in combination with ideal wheels include so called holonomic constraints. Such constraints are based on location and in case of single-track vehicles prevent them from sinking into the ground.

3 Eigenvalue Analysis

Due to geometry and gyroscopic forces Klein and Sommerfeld [8] found out that a single-track vehicle is self-stabilizing within a certain velocity range. That is, the vehicle performs a tail motion in the longitudinal direction. Below this range the steering deflections caused by gyroscopic forces are too small in order to generate enough centrifugal force. Thus the amplitude of the tail motion increases and the vehicle falls over. Although, these interactions are damped by the trail[1] it is still impossible to achieve stable behavior. Hence

[1]The trail is the distance between the front wheel contact point and the point of intersection of the steering axis with the ground line (horizontal axis).

the rider has to apply a steering torque to ensure that the vehicle stays upright. Above this range, for high speeds, the gyroscopic forces are almost unnoticeable for the rider. That is, the amplitude of the tail motion is close to zero. More precisely, although the vehicle feels stable, after a certain time, it falls over like a capsizing ship. However, by applying a steering torque it is rather simple to stabilize the vehicle. In most cases it is sufficient that one solely touches the handle bars in order to compensate for the instabilities.

An eigenvalue analysis is performed in order to determine the self-stabilizing range of an uncontrolled bicycle or motorcycle. For this purpose the state variables of the vehicle that are responsible for stability are of interest. These are the steer angle δ, the lean angle ϕ, and their derivatives.

$$x = \begin{pmatrix} \delta \\ \dot{\delta} \\ \phi \\ \dot{\phi} \end{pmatrix}$$

In case of vehicles with an additional d.o.f. allowing the rider's upper body to lean sideways, the state variables γ and $\dot{\gamma}$ are also taken into account, where γ is the lean angle of the rider's upper body relative to the rear frame and $\dot{\gamma}$ the corresponding lean rate. All the other state variables (e.g. lateral- and longitudinal position) of the state vector have no influence on the stability of single-track vehicles. Now, the eigenvalues (one for each state variable) are calculated as a function of the vehicle's forward velocity $\lambda = f(v)$ (e.g. $v = 10ms^{-1}$ to $v = 50ms^{-1}$). Thus, for each specific velocity the model is linearized. The result of such an analysis are three different velocity ranges at which the motion of the vehicle changes qualitatively. Figure 3 depicts a typical result of such an analysis. The first velocity range is below the stable region, the second one is within, and the third one above the stable region. Positive eigenvalues, or more precisely eigenvalues with a positive real part, correspond to unstable behavior whereas eigenvalues with a negative real part correspond to stable behavior. Eigenvalues including an imaginary part emphasize that the system is oscillating whereas eigenvalues without an imaginary part are non-oscillating. A stable region exists, if and only if all real parts of the eigenvalues are negative. In the following, the modes of single-track vehicles are explained with reference to Figure 3.

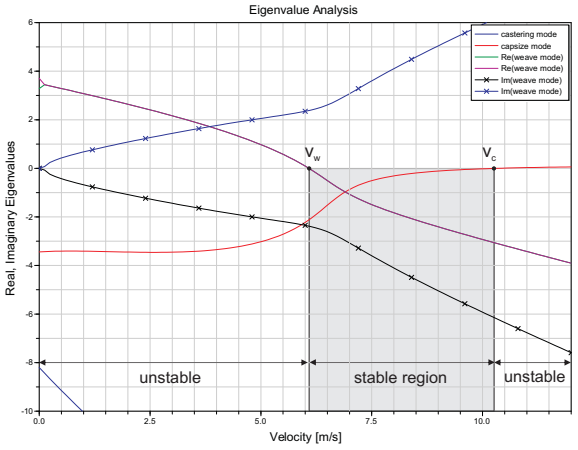

Figure 3: Result of the eigenvalue analysis for a 3 d.o.f. motorcycle model. The stable region is determined by eigenvalues with a negative real part. Here it is from $v_w = 6.1ms^{-1}$ to $v_c = 10.3ms^{-1}$.

3.1 Weave Mode

The weave mode begins at zero velocity. This mode is non-oscillating in the beginning and after a certain velocity passes over into an oscillating motion. The non-oscillating motion at very low speeds states that the bicycle is too slow to perform a tail motion and thus falls over like an uncontrolled inverted pendulum. As soon as it passes a certain value of approximately $v_w = 0.12ms^{-1}$ the real parts of the eigenvalues merge and two conjugate complex eigenvalues appear. Hence, a tail motion in the longitudinal direction emerges. This motion is still unstable but becomes stable as soon as the real parts of the eigenvalues cross zero. This happens at a velocity of about $v_w = 6.1ms^{-1}$. For all velocities greater than v_w this motion is stable.

3.2 Capsize Mode

The capsize mode is a non-oscillating motion that corresponds to a real eigenvalue dominated by the lean. As soon as the bicycle speed passes the upper limit of the stable region of about $v_c = 10.3ms^{-1}$, it falls over like a capsizing ship. However, above the stable region the bicycle is easy to stabilize although the real eigenvalue is positive. In the paper [12] of Sharp et al. this motion is called "mildly unstable" as long as the absolute value of the eigenvalues is smaller than $2s^{-1}$.

3.3 Castering Mode

The castering mode is a non-oscillating mode that corresponds to a real negative eigenvalue dominated by the steer. In this mode the front wheel has the tendency to turn towards the direction of the traveling vehicle.

4 Controller Design

4.1 An Introduction to State-Space Design

In general, the state-space representation of a linear system is given by:

$$\dot{x} = A \cdot x + B \cdot u, \quad x(0) = x_0 \tag{1}$$

$$y = C \cdot x + D \cdot u \tag{2}$$

where x is a $(n \times 1)$ state vector, y is a $(m \times 1)$ output vector, A is referred to as system matrix with a dimension of $(n \times n)$, B is a $(n \times r)$ input matrix, C is a $(m \times n)$ output matrix and D the "feedthrough" matrix with a dimension of $(m \times r)$. Usually D is set to zero, except if the output directly depends on the input. The state vector x at time $t = 0$ includes the initial conditions, sometimes referred to as initial disturbances x_0. The block diagram of a system described in state-space is illustrated in Figure 4. One major advan-

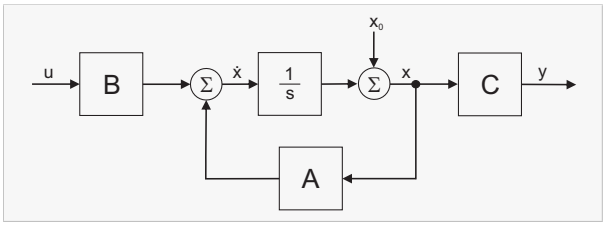

Figure 4: Block diagram of a system described in state-space

tage of state-space control compared to classic control is that each state of the system can be controlled. In order to control the system, the state vector x is fed back. The state feedback control law for a linear time-invariant system is given by:

$$u(t) = -F \cdot x(t) \tag{3}$$

where F is a constant matrix.

By substituting u of Equation 1 with Equation 3 the state equation results in

$$\dot{x} = A \cdot x - B \cdot F \cdot x = (A - B \cdot F)x \tag{4}$$

The block diagram of the equation above is shown in Figure 5.

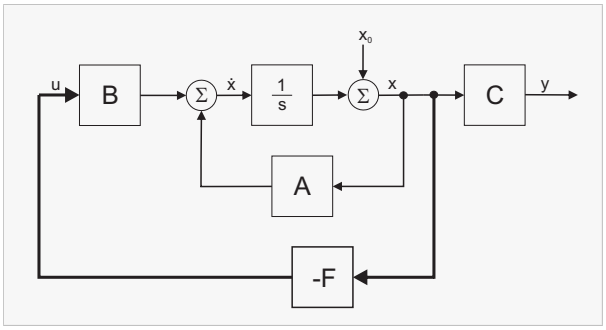

Figure 5: State feedback

The elements of the feedback matrix F have to be chosen in such a way that the *initial disturbances* $x_0(t)$ for $t \to \infty$ converge towards zero

$$\lim_{t \to \infty} x(t) = 0 \tag{5}$$

and that the system becomes stable.

The main task of the state feedback control is to find appropriate coefficients for the feedback matrix F in order to achieve the desired dynamical behavior of the system. One method that fulfills all the requirements is the so-called pole placement technique (refer to [7]). Figure 6 illustrates the graphical interpretation.

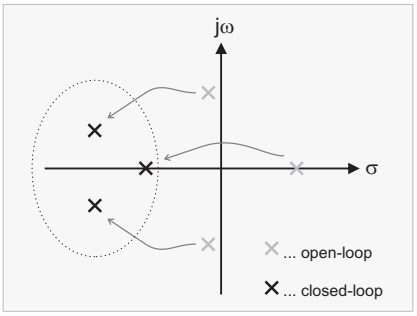

Figure 6: Graphical interpretation of the pole placement technique. Eigenvalues (poles) of the system located in the left-half plane correspond to stable behavior.

4.2 State-Space Controller Design Based on a Preceding Eigenvalue Analysis

The library includes several different stabilizing controllers. Although classic controllers and linear quadratic regulators (LQR) are included in the library, the focus lies in state-space controller design via the pole placement technique. In the simplest case the lean angle and the lean rate of the vehicle are fed back in order to generate an appropriate steering torque. However, since a physical interpretation of these eigenval-

ues is not possible an alternative approach is introduced in this paper. This approach is based on a preceding eigenvalue analysis. That is, exactly the same state variables, namely the steer angle δ, the lean angle ϕ, and their derivatives, are used to design the controller (see Figure 7). Of course if the upper body of

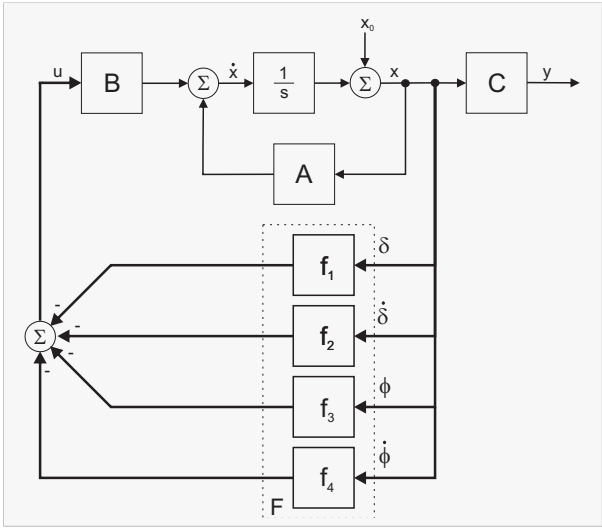

Figure 7: State-space controller based on a preceding eigenvalue analysis

the rider is movable, the states γ and $\dot{\gamma}$ are taken into account as well. Thus, a physical interpretation of the poles is available.

As already mentioned the eigenvalues are a function of the velocity, i.e. the trajectory of each eigenvalue is thus perfectly known. With this knowledge the velocity dependent coefficients of the state feedback matrix can be conveniently calculated. To this end, three different approaches were developed. In the first approach all eigenvalues (poles) of the system are simply shifted towards the left-half plane (see figure 6) by the same value (offset). A typical result for the controlled version of the 3 d.o.f. motorcycle is illustrated in Figure 8.

Two improved control laws have been established. Both are based on solely shifting those poles towards the left-half plane that are unstable. Within the stable region the motorcycle needs no control and thus no offset. Above the stable region (for velocities greater than v_c) the behavior of the bicycle is dominated by the capsize mode. Hence, it is absolutely sufficient to shift just this pole towards the left-half plane and leave all other poles unchanged. Below the stable region, for velocities lower than v_w, the instability of the motorcycle is caused by the weave mode (see Figure 9). To ensure stable behavior the two real parts of the conjugate complex poles have to be shifted towards the left-half

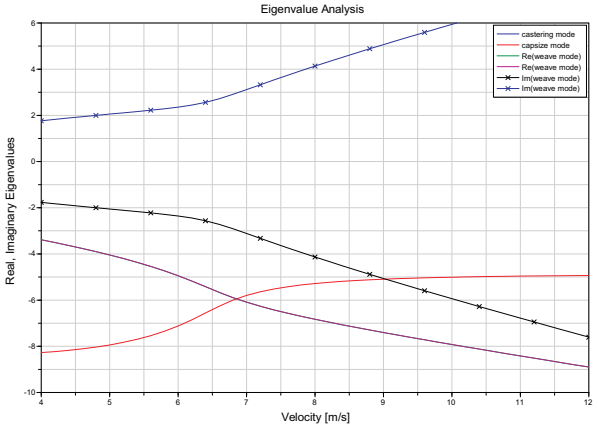

Figure 8: Result of the controller design for a velocity range from $4ms^{-1}$ to $12ms^{-1}$, where the offset is $d = 5$.

plane. Now, a control law for the regions below and above the stable region is set up:

$$control\ law \begin{cases} v < v_w : & d = d_w \cdot (v_w - v) \\ v_w < v < v_c : & d = 0 \\ v_c < v : & d = d_c \cdot (v - v_c) \end{cases}$$

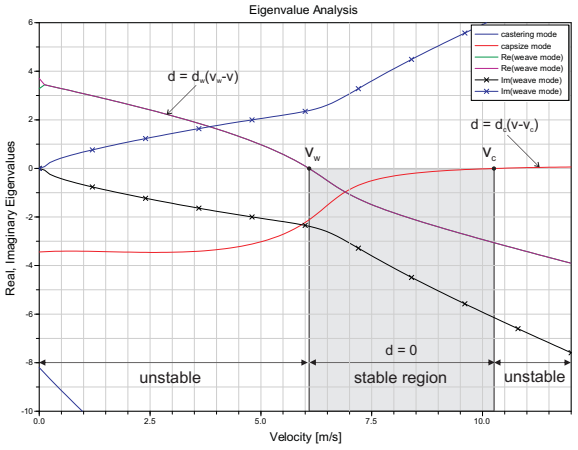

Figure 9: Controller design with reference to a preceding eigenvalue analysis. The stable region is left unchanged - below v_w, the weave mode eigenvalues are modified - above v_c, the capsize mode eigenvalue is modified.

Figure 10 shows the result of the individual controller design. Although the results of the individual controller are rather good, there is still potential for improvements. For velocities equal to v_w or v_c the eigenvalues that are responsible for stability are close or equal to zero. To be more precise, for such velocities the stability of the motorcycle is critical since a real part equal to zero has no damping. Somewhere in

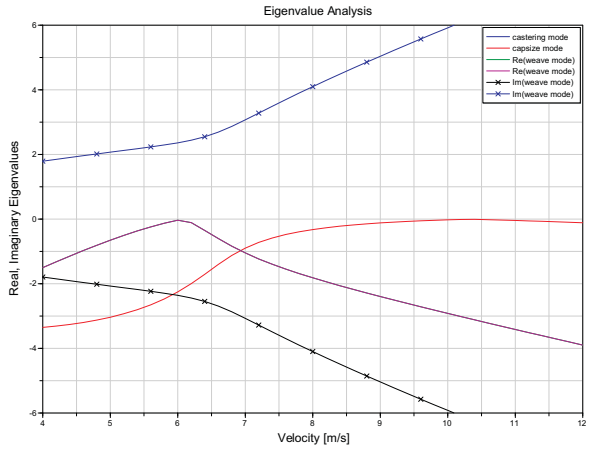

Figure 10: Result of the individual controller design for a velocity range from $4ms^{-1}$ to $12ms^{-1}$, where $v_w = 6.1ms^{-1}$, $v_c = 10.3ms^{-1}$, $d_w = 1.5$ and $d_c = 0.1$.

the stable region the weave and the capsize mode have an intersection point v_i. Instead of the previous control law, the improved control law results in:

$$control\ law \begin{cases} v < v_i: & d = d_0 + d_w \cdot (v_i - v) \\ v_i < v: & d = d_0 + d_c \cdot (v - v_i) \end{cases}$$

A graphical interpretation of the control law is illustrated in Figure 11.

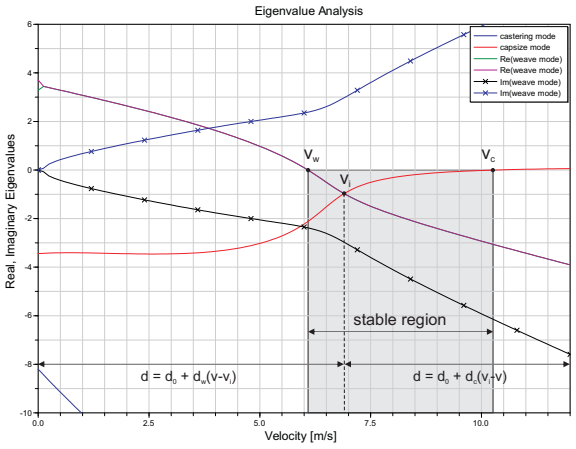

Figure 11: Controller design with reference to a preceding eigenvalue analysis. Below v_i, the weave mode eigenvalues are modified - Above v_i, the capsize mode eigenvalue is modified. In addition, the weave and capsize eigenvalues can be shifted by an offset d_0.

Figure 12 depicts the result of the improved individual controller design.

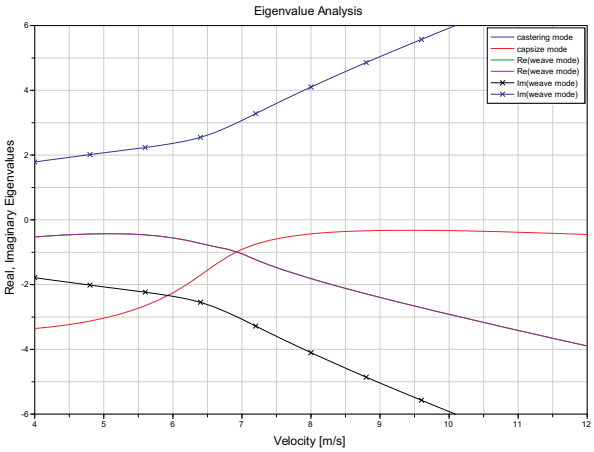

Figure 12: Result of the improved individual controller design for a velocity range from $4ms^{-1}$ to $12ms^{-1}$, where $v_i = 6.9ms^{-1}$, $d_w = 0.75$, $d_c = 0.1$ and $d_0 = 0$

4.3 Results

The results are several pole placement functions that automatically calculate the controller coefficients, i.e. the elements of the feedback matrix. The corresponding output represents a state feedback matrix that can be directly applied to ready-made controllers. The algorithm of the functions is based on *Ackermann's formula*. Unfortunately, it is just valid for single-input, single-output (SISO) systems. In order to design a multiple-input, multiple-output (MIMO) controller, e.g. for vehicles including rider's capable of leaning sideways, a MATLAB *m-file* based on the *place*-function is provided.

Finally, the coefficients of the state feedback matrix are automatically fed into a ready-made controller which is incorporated into a virtual rider. With respect to the virtual rider the vehicle's performance can now be evaluated.

5 Development of a Virtual Rider

5.1 Roll Angle Tracking

For virtual riders capable of tracking a roll angle profile, several test tracks are provided. So far, no reference input was used, i.e. the set-value of the state variables was zero. Instead of a set-values equal to zero, the roll angle profile (e.g. of a standard 90°-curve) is fed into the virtual rider. The corresponding block diagram is illustrated in Figure 13. Since each vehicle has its own specific profile, some records including such profiles are provided. The corresponding Mod-

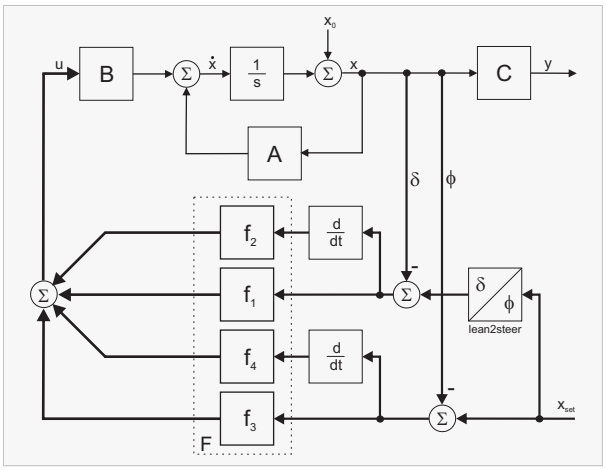

Figure 13: Block diagram of a virtual rider composed of a state-space controller and an additional block in order to calculate the corresponding steer angle. The reference input x_{set} is the desired roll angle profile.

elica model is depicted in Figure 14. The incorporated

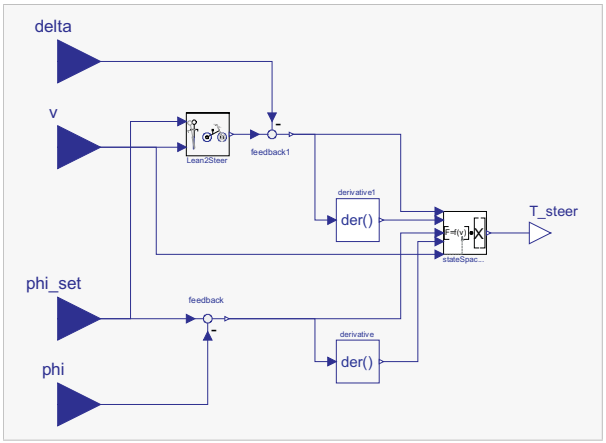

Figure 14: Wrapped model of the virtual rider composed of a state-space controller for a user defined velocity range. The inputs (blue) are lean and steer angle, lean angle set-value and the velocity of the motorcycle, the output T_{steer} (white) is the steering torque

controller is shown in Figure 15.

5.2 Path Tracking

In order to track a pre-defined trajectory using path preview information, a randomly generated path is included in the library. The path generation was done with MATLAB. This path is defined by its lateral profile [16]. To emulate the behavior of a human rider, single-point path preview is performed by the virtual rider. That is, the rider looks a pre-defined distance ahead in order to follow the path. It is worth noting that

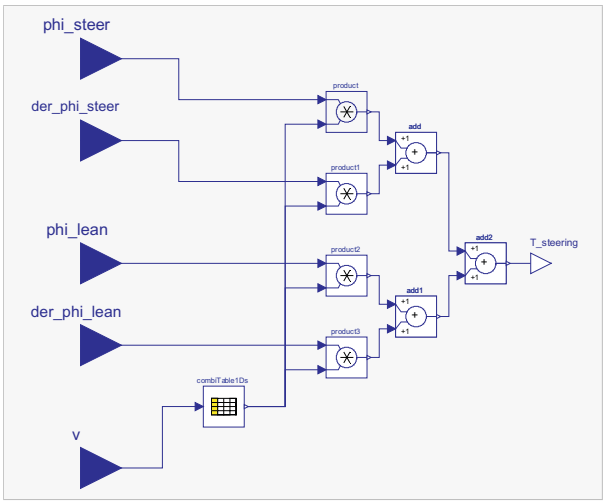

Figure 15: Wrapped model of a state-space controller for a specific velocity range. The table includes the state feedback matrix coefficients which by default are stored in *place.mat*

a similar deviation pattern is actually observed from human riders. In order to track a path, the controllers have to be extended [16] (see Figure 16). In the sim-

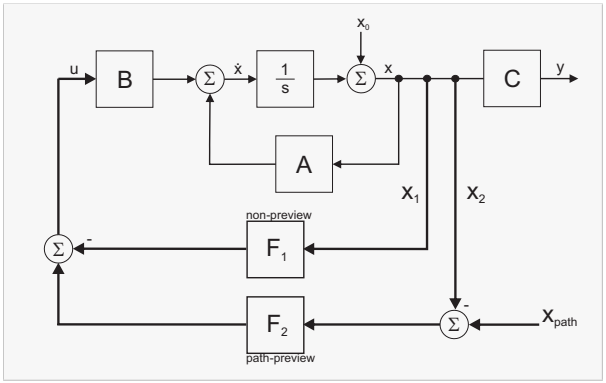

Figure 16: Basic structure of a state-space path preview controller. The state vector x_1 includes the states that are responsible for the stability (non-preview), whereas $x_2 = (x_{lat}\ \dot{x}_{lat})^T$ includes the states required for path tracking.

plest case the lateral position x_{lat} of the rear frame's center of mass is fed back in order to generate an additional steering torque that keeps the vehicle on the desired path. For the utilized state-space controllers the lateral rate $\dot{x}_{lat}$ is additionally taken into account. The corresponding Modelica model is basically the same as the one depicted in Figure 14. To cover the path tracking capabilities two additional inputs, namely x_{lat} and $\dot{x}_{lat}$ are included. For the state-space path tracking controller the pole placement functions were extended in order to conveniently design such a controller.

6 Examples

The first example demonstrates a 3 d.o.f. motorcycle stabilized by a virtual rider. The animation of the uncontrolled vehicle with a velocity of $4ms^{-1}$ is depicted in Figure 17. In order to determine the self-stabilizing

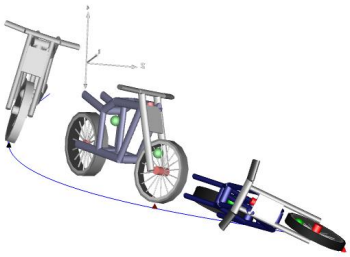

Figure 17: Animation result of the uncontrolled 3 d.o.f. motorcycle. After about 2s the motorcycle falls over like an uncontrolled inverted pendulum

range of the motorcycle an eigenvalue analysis is carried out. The results are shown in Figure 3. According to these results it can be seen that the vehicle is truly unstable for a velocity of $4ms^{-1}$. Furthermore, it can be seen that an offset of $d = 2$ is absolutely sufficient to achieve stable behavior. With this information the coefficients of the state feedback matrix are calculated. For this purpose the pole placement function based on the first approach is executed. The corresponding output is stored in the controller of the ready-made virtual rider introduced in Figure 14. The model of the controlled motorcycle is depicted in Figure 18. The ani-

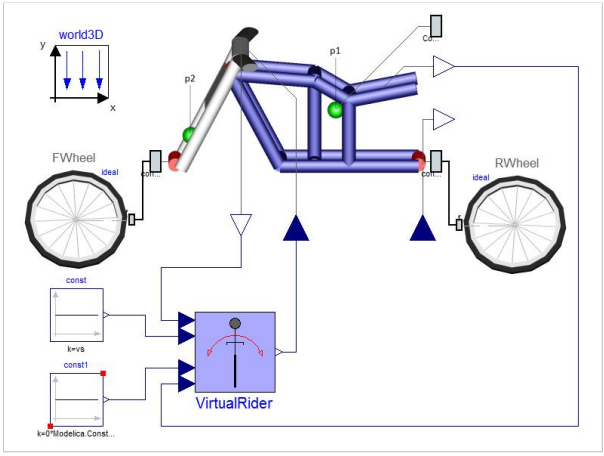

Figure 18: Example: controlled 3 d.o.f. motorcycle. The wrapped model of the virtual rider corresponds to Figure 14.

mation of the controlled vehicle is shown in Figure 19.

In the second example the motorcycle tracks a roll

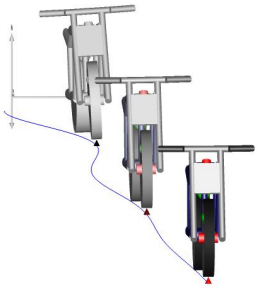

Figure 19: Animation result of the controlled 3 d.o.f. motorcycle

angle profile. The utilized model is depicted in Figure 18. Instead of a *constant source* block, the model of a 90°-curve is included. The coefficients of the feedback matrix were automatically calculated for an offset $d = 5$. The resulting eigenvalues are equal to those depicted in Figure 8. The animation result is depicted in Figure 20.

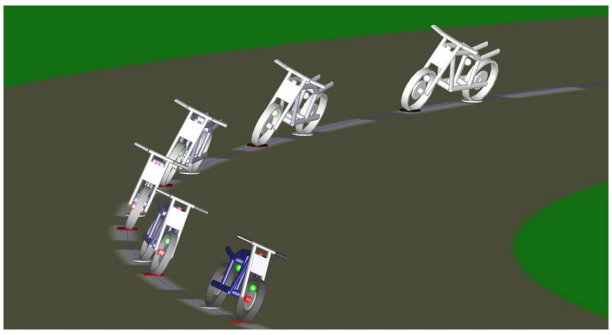

Figure 20: Animation result of a 3 d.o.f. motorcycle tracking a 90°-curve

In the last example the motorcycle tracks a predefined path. The utilized model is depicted in Figure 21.

Again, the coefficients of the feedback matrix are automatically calculated for an offset $d = 5$. Additionally, an offset $d_{lat} = 5$ is needed in order to keep the motorcycle on the desired path. The results are shown in Figure 22.

7 Structure of the Library

The structure of the *MotorcycleLib* is depicted in Figure 23. For each single-track vehicle a separate subpackage is provided. The basic bicycle sub-package is composed of a rigid rider and a movable rider subpackage. Both include the corresponding wrapped model and a function in order to perform an eigenvalue analysis. The motorcycle sub-package also in-

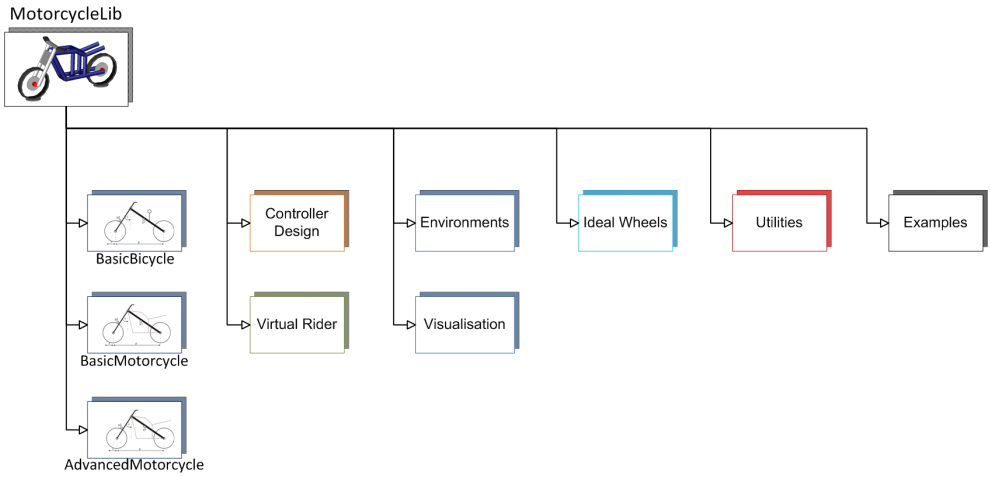

Figure 23: Library structure

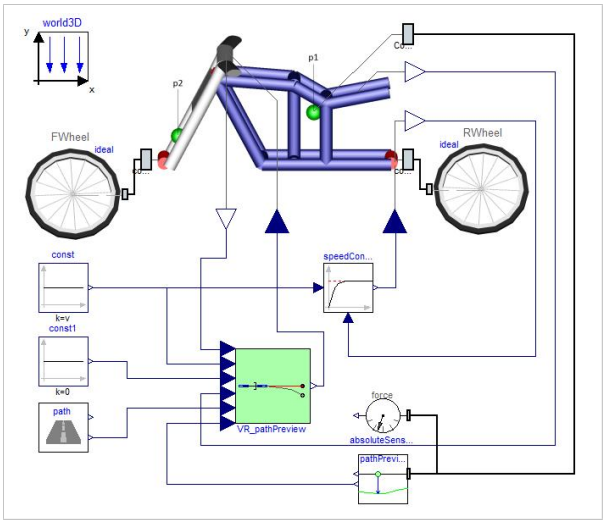

Figure 21: Example: model of a 3 d.o.f. motorcycle tracking a pre-defined path

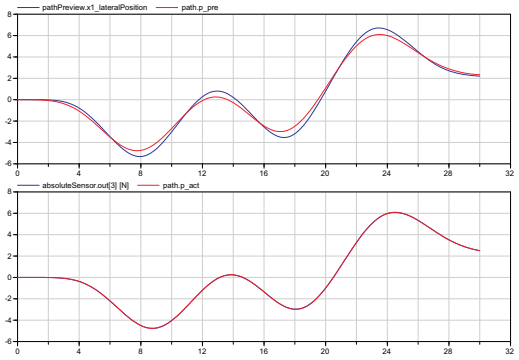

Figure 22: Simulation result of a motorcycle tracking a pre-defined path. Upper plot: The red signal is the path a pre-defined distance ahead, the blue signal is the preview distance of the rider. Lower plot: The red signal is the actual path, the blue signal is the traveled path measured at the rear frame's center of mass

cludes a wrapped model and an eigenvalue analysis function. The structure of the advanced motorcycle models is much more detailed since each part (e.g. front frame) is created in a fully object-oriented fashion. It is composed of a parts, an aerodynamics and a stability analysis sub-package. The parts sub-package includes several different front and swinging arms, a rear frame, the rider's upper body, a torque source (engine), an elasto-gap and a utilities sub-package. In the latter one, models of characteristic spring and damper elements are stored. The aerodynamics sub-package includes a lift force, a drag force and a pitching moment model.

The controller design sub-package contains pole placement functions in order to design appropriate controllers. The virtual rider sub-package includes, among others, a virtual rigid rider and a virtual movable rider sub-package. In both the riders are capable of either tracking a roll angle profile or a pre-defined trajectory. To this end, several different controllers (e.g. classic, state-space and LQR) are incorporated into the virtual riders.

The environments sub-package provides tracks for both roll angle tracking and path tracking. In addition, the models for single-point path tracking are included. The visualization sub-package provides the graphical information for the environments sub-package. In the ideal wheels sub-package the visualization of the rolling objects from D. Zimmer's *MultiBondLib* were modified such that the appearance is similar to real motorcycle wheels. The utilities sub-package provides some additional functions and models which are partly

used in the library. The purpose of the examples subpackage is to provide several different examples that demonstrate how to use the library.

8 Conclusion

The library provides an appropriate eigenvalue function for each vehicle. Beside the controller design such an analysis is beneficial for the optimization of the vehicle's geometry. By changing the geometry or the center of mass' locations of a vehicle, the eigenvalues of the system are changing as well. It is thus possible to optimize the design of a vehicle regarding self-stability.

Furthermore, due to the results of the eigenvalue analysis it is now possible to conveniently design a state-space controller valid for a specific velocity range of the vehicle. Thus, for the calculation of the state feedback matrix coefficients, a pole placement function was developed. In order to design an LQR, MATLAB functions are provided.

To test the performance of the vehicles, the virtual riders are capable of tracking both, a roll angle profile and a pre-defined path. Therefore, several test tracks are included in the library.

A very detailed description of this paper can be found in the corresponding master's thesis [10].

References

[1] M. Andres. Object-oriented modeling of wheels and tires. Master's Thesis, 2009.

[2] F. E. Cellier and À. Netbot. The modelica bondgraph library. In *Proceedings of the 4th International Modelica Conference, Hamburg*, pages 57–65, 2005.

[3] V. Cossalter. *Motorcycle Dynamics*. 2006. 2nd edition.

[4] F. Donida, G. Ferreti, S. M. Savaresi, and M. Tanelli. Object-oriented modeling and simulation of a motorcycle. *Mathematical and Computer Modelling of Dynamic Systems*, 14, No. 2:79–100, 2008.

[5] F. Donida, G. Ferreti, S. M. Savaresi, M. Tanelli, and F. Schiavo. Motorcycle dynamics library in modelica. *Proceedings of the Fifth International Modelica Conference*, 5:157–166, 2006.

[6] S. Evangelou. *The control and stability analysis of two-wheeled road vehicles*. PhD thesis, Imperial College London, September, 2003.

[7] Otto Föllinger. *Regelungstechnik, Einführung in die Methoden und ihre Anwendungen*. Hüthig, 2008. 10. durchgesehene Auflage.

[8] F. Klein and A. Sommerfeld. Über die theorie des kreisels. *Quarterly Journal of Pure and Applied Mathematics*, Chapter 9, Section 8:863–884, Leipzig, 1910.

[9] C. Koenen. *The dynamic behaviour of motorcycles when running straight ahead and when cornering*. PhD thesis, Delft University, 1983.

[10] Thomas Schmitt. Modeling of a motorcycle in dymola/modelica. Master's thesis, Vorarlberg University of Applied Sciences, 2009.

[11] A. L. Schwab, J. D. G. Kooijman, and J. P. Meijaard. Some recent developments in bicycle dynamics and control. *Fourth European Conference on Structural Control*, page 8, 2008.

[12] A. L. Schwab, J. P. Meijaard, and J. M. Papadopoulos. Benchmark results on the linearized equations of motion of an uncontrolled bicycle. *KSME International Journal of Mechanical Science and Technology*, pages 292–304, 2005.

[13] R. S. Sharp. The stability and control of motorcycles. *Journal Mechanical Engineering Science*, Volume 13:316–329, 1971.

[14] R. S. Sharp, S. Evangelou, and D. J. N. Limebeer. Advances in the modelling of motorcycle dynamics. *Multibody System Dynamics*, Volume 12:251–283, 2004.

[15] R. S. Sharp and D. J. N. Limebeer. A motorcycle model for stability and control analysis. *Multibody System Dynamics*, Volume 6:123–142, 2001.

[16] R. S. Sharp and V. Valtetsiotis. Optimal preview car steering control. *Vehicle System Dynamics*, Supplement 35:101–117, 2001.

[17] D. Zimmer and F. E. Cellier. Multibond graph library. *Proceedings of the Fifth International Modelica Conference*, pages 559–568, 2006.