# “Curve Driving” Project Work

DHBW Automotive Software Engineering – Stuttgart, 

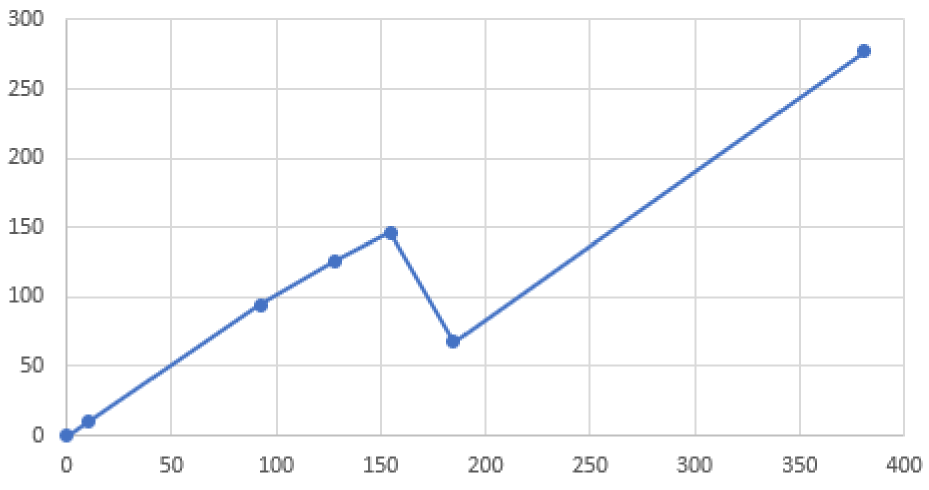
## 1. Introduction and Overview

The project work explores the automation of curve driving. It is based on a simplifies single track model as provided in the lecture:

```
ds = v*mydt;
bearing = bearing + (ds/D)*Lib.sin(beta);
if (bearing > pi) {
    bearing = bearing - 2*pi;
}
if (bearing < -pi) {
    bearing = bearing + 2*pi;
}
x = x + ds*Lib.cos(bearing);
y = y + ds*Lib.sin(bearing);
```

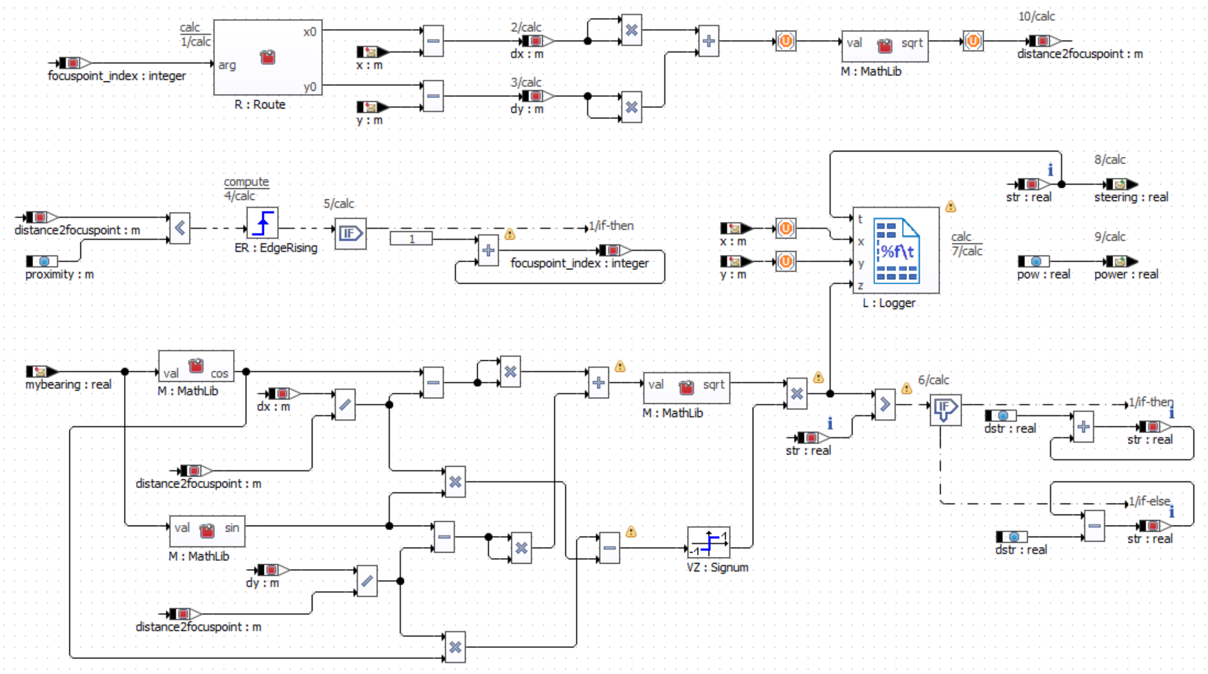
where the variable `beta` denotes the steering angle of the wheel (not the steering wheel which typically would differ by a factor of 18) and `bearing` the orientation of the vehicle. `D` is the fixed distance of the two vehicle axes.

As per example there is a simple track to follow provided in some **Route** class:

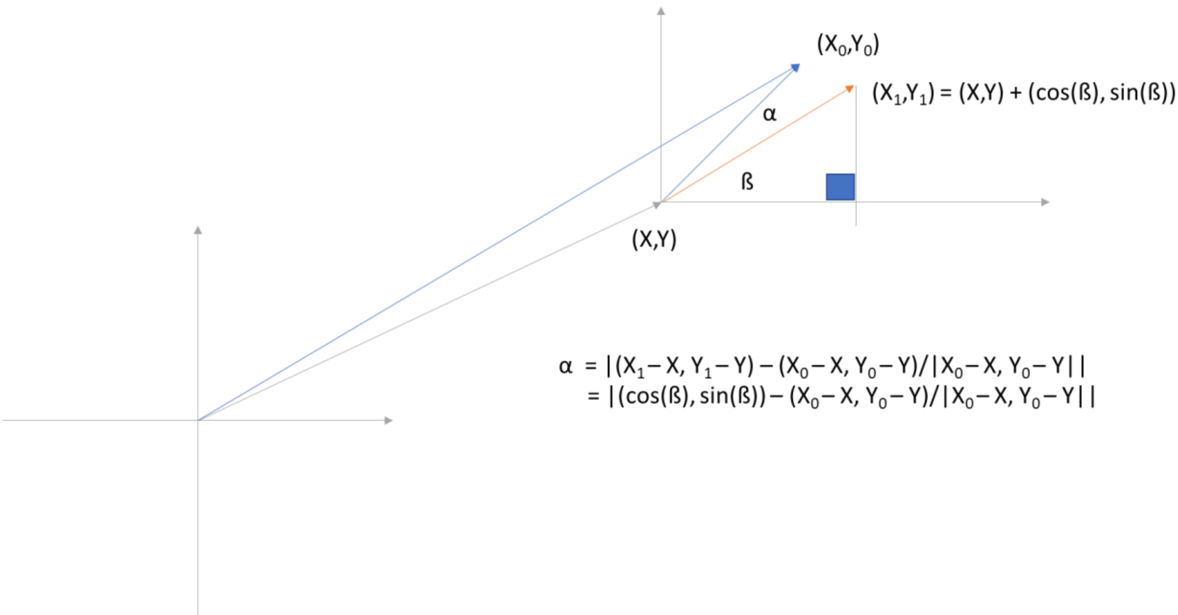



As the drive model the given **myDrive_4** class must be used with a flat landscape i.e. setting the input quantity `myg` to `0.0[a]`. Simulation shall happen on a 10ms time frame.

A draft implementation may look like the below. It may be improved by the introduction of some re-usable components of a vector distance and a projected distance of some vector on to another.




The following picture may help with the background of the above. Note, that the vector from current position there (X,Y) to the next focus point ($X_0$,$Y_0$) needs to the normed before calculation of the steering (with a small angle assumption) can be performed.




Note further, that one needs to check for the positive or negative direction of the steering angle α with a rotation operation (in the block diagram related to the sign function).

# 2. Requirements and Deliverables

Please refer to “ID“ column in your documentation. A * denotes extra tasks / requirements.

| ID | Check |
|---|---|
| R1 | The **Route** shall be driven most the time with some 60 km/h accelerating from 0 km/h at the start point at x = y = 0. |
| R2 | Algorithmically the automated driver shall steer from data point to data point, focusing on the next one once a **proximity distance** of the current focus point is somehow about the vehicle’s turning radius. |
| R3 | The **maximum steering angle** shall be 0.5 rad. |
| R4 | The **steering angle velocity** must not exceed 0.01 rad per 10 ms. |
| R5 | The vehicle shall come to a **full stop** close to the last point of the Route. |
| D1 | **Plan** all necessary tasks based on three-point-estimates and monitor progress according to below requirements. Provide overall 95% estimate too. Deliver the Excel-Sheet used. |
| D2 | Demonstrate the **feasibility/limitations** of the following tasks by simple arguments, calculation or simulation experiments. |
| D3 | Compare a calculated and simulated **turning radius** given requirement R5. |
| D4 | Provide some library add-ons to support (as needed) **vector operations** like the norm and the angle between two vectors, the distance of a point to a line, etc. or enhance the Route class accordingly. |
| D5 | Redesign the algorithm depicted in the block diagram above with the aid of D4 to **drive the Route** according in line with the requirements R1 to R5. |
| D6 | Provide some **validation method** to compare the actual route with the targeted one and check the solution as implemented via D5 accordingly. |
| D7* | Design another **personal** control algorithm or calibration to beat D5/D6 results. |
| D8* | Compare the tests of D6 (and D7) with **different tracks** from real road examples. |
| D9* | Check how one should adapt the velocity to limit **lateral acceleration**. |
| D11 | Document all tasks properly in a **PDF document** and provide documentation and artefacts (i.e. **ASCET export archive** and **estimation Excel sheet**). |
| D12* | **Reason beyond**, about the required accuracy of maps, the impact of road constructions and the like. |
| D13* | **Reflect**: Which other observations or comments are in place concerning the planning, the modeling, the requirements, the prescribed functions, or your solution, the testing, and the selected graphical approach. |

# 3. Documentation and Modeling Guidelines

For the project work please follow these rules:

a. Document your modeling with proper screenshots (or a screen video).
b. Simulations, mathematics, or unit tests motivate or prove all arguments. Simple solutions are preferred (but don't oversimplify). Hint: You may change the above parameters for development purposes (to cut down waiting time).
c. All design decisions (e.g. definition of classes, selection of BDE vs. ESDL, selection of parameter vs. literal or system constant, simulation settings like timing, data types including units, default values and ranges for variables and parameters) are motivated in writing. Use basic principles like encapsulation, minimize messages, classes (and functions therein), as well as other artifacts. Remove unused elements from classes.
d. Every text is properly readable with a minimum font size of 10 pt and all screenshots have a proper legend and are referred in the text. Define and use proper naming convention with meaningful English names.
e. Graphical modeling is subject to best practice i.e. at most 30 elements shall be present in block diagrams with at most 10 line crossings (minimize line length). All texts (e.g. of sequence calls) are clearly readable and allow a natural reading order (left top to right down). Add comments to highlight or explain important facts. Don't use Containers as they hide potentially important information e.g. sequence calls.
f. Experiments contain all necessary data for the arguments to be made (and not more) suitably grouped. Screenshots have a white background. Scale, line thickness, and color must allow clear distinction of what needs to be distinguished to underpin arguments.
g. Make use of unit tests where beneficial and document them properly with logged data.