**Project assignment in the course 'Mathematical Methods of Computer Graphics' in SS2023**

**Project:**

**Simulation of a roller coaster or a sightseeing flight using Bezier interpolation and visualization of the differential geometric quantities**

In this semester we have mainly dealt with the elementary differential geometry of parameterized (space) curves. In the case of space curves, the focus was on the quantities of curvature and torsion or Frenet's tripod and the parallel frame.

These variables should now also be used in the geometric modeling / visualization of the roller coaster track. The entire line should be composed of individual cubic Bezier curve sections, namely as a closed circuit that is continuously differentiable ('kink-free') and 'curvature-constant' (C2 or G2).

The **Berzier interpolation** is to be used for the realization of the line. The course of the route is to be formed from individual curve points that are to be read in from a text file. **The curve points are to be interpolated in such a way that the result is a closed, continuous, kink-free and curvature-jerk-free circuit.**

The resulting curve should therefore be 2x continuously differentiable. It should be closed and realized with cubic Berzier curve segments.

In this project, we are orienting ourselves on the software internship at **the Technical University of Braunschweig** Institute for Scientific Computing - on the 'roller coaster editor/simulator'. The documentation for the software project is in the EMIL course, in Chapter 4 it shows the geometric modeling. This is explained simply in the documentation, i.e. it is presented in a way that is easy to understand, and has to do with mathematical methods.

**According to the content of our lecture, the following variables are to be calculated and visualized separately for the respective route:**

**• Curvatures κ1(t), κ2(t) and torsion τ(t) each as a function of the parameter t**

**• the magnitudes of the 1st, 2nd and 3rd derivatives (velocity, acceleration and jerk) each as a function of the parameter t**

**• Turn with rotating Frenet tripod (control by aileron and elevator)**

**• Curve with rotating parallel frame (steered by rudder and elevator)**

~~The aim is to model the route as a surface in which a straight line or a section of a circle is shifted along the curve (i.e. in the direction of the velocity vector) and is in the plane spanned by the normal vectors. The straight/circle piece rotates like the respective tripod with the curve point.~~

The carriage of the roller coaster should be simulated by a sphere or a cuboid or simply a point that moves along the curve together with the respective tripod.

Another point in the project is the creation of the samples for the course of the route.

There are predefined routes in the form of trk files with which the simulation should run. These are stored in the Moodle course.

**In addition, a separate (non-trivial) route should be designed.** The points of the route are then to be saved in a file which is later read in for the Bezier interpolation**. The own design should be realized as a sampled torus knot or another variant of a discretized Fourier knot.**

By saving the sample values/curve points in a uniform format, other routes for the roller coaster can also be interpolated at the end and their behavior can be compared geometrically and dynamically.

A simple text file should be taken with three floating-point numbers in a line for each point of the route and a separator (space or tab or semicolon), i.e. the good old CSV format.

Of course, you can also choose a different format. But then you also have to provide a program for converting the CSV format and its format and back again, so that your implementation of the project can also be tested with other data, i.e. curves.

The programming should be done with **Python** or **Java script** or alternatively with **Matlab**.

 The visualization can also be done in Blender, whereby the programming is done in Python and Blender is used more as a display tool/framework program.

The program code should be self-documenting, i.e. it should be relatively easy to understand with sufficient comments.

**Submission deadline: 01.10.2023**

**Submission by uploading the files in the Moodle course (program code, track files and short documentation on how to use the program, if necessary conversion programs for the tracks' own file formats)**