Digital 3D Geometry Processing
Exercise 4 – Curves

Handout date: Mon, 08.10.2018
Submission deadline: Thu, 18.10.2018, 23:00 h

What to hand in

A .zip compressed file renamed to `Exercise`n`-Group`i`.zip` where n is the number of the current exercise sheet and i is the number of your group. It should contain:

- Hand in **only** the files you changed (headers and source). It is up to you to make sure that all files that you have changed are in the zip.
- A `readme.txt` file containing a description on how you solved each exercise (use the same numbers and titles) and the encountered problems. Indicate what fraction of the total workload each project member contributed.
- Other files that are required by your `readme.txt` file. For example, if you mention some screenshot images in `readme.txt`, these images need to be submitted too.
- For the theory exercise, put `TheoryExercise.pdf` with your solutions in the same .zip you submit for the code.
- Submit your solutions to Moodle before the submission deadline. Late submissions will receive 0 points! The total points of this homework is **100**.

1 Theory Exercise (40 pt)

1.1 Curvature of Curves (20pt)

Match each curve from Figure 1 with the corresponding curvature function:

$$k_1(s) = \frac{s^2-1}{s^2+1}, \quad k_2(s) = s, \quad k_3(s) = s^3 - 4s, \quad k_4(s) = \sin(s)s. \tag{1}$$

Write your solutions in a file named `TheoryExercise.pdf`. If, for example, the curvature function $k_1(s)$ corresponds to the curve **a**, write **1-a** as your solution. Briefly explain your answers.

1.2 Surfaces Area (20pt)

King Archimedes wants to renovate his palace. The most striking structure is a spherical half-dome of 20m in diameter that covers the great hall. The king wants to cover this dome in a layer of pure gold. He has decided to split the work into two parts, each one covering a vertical slice of the dome of the same height (see Figure 2). For each part he

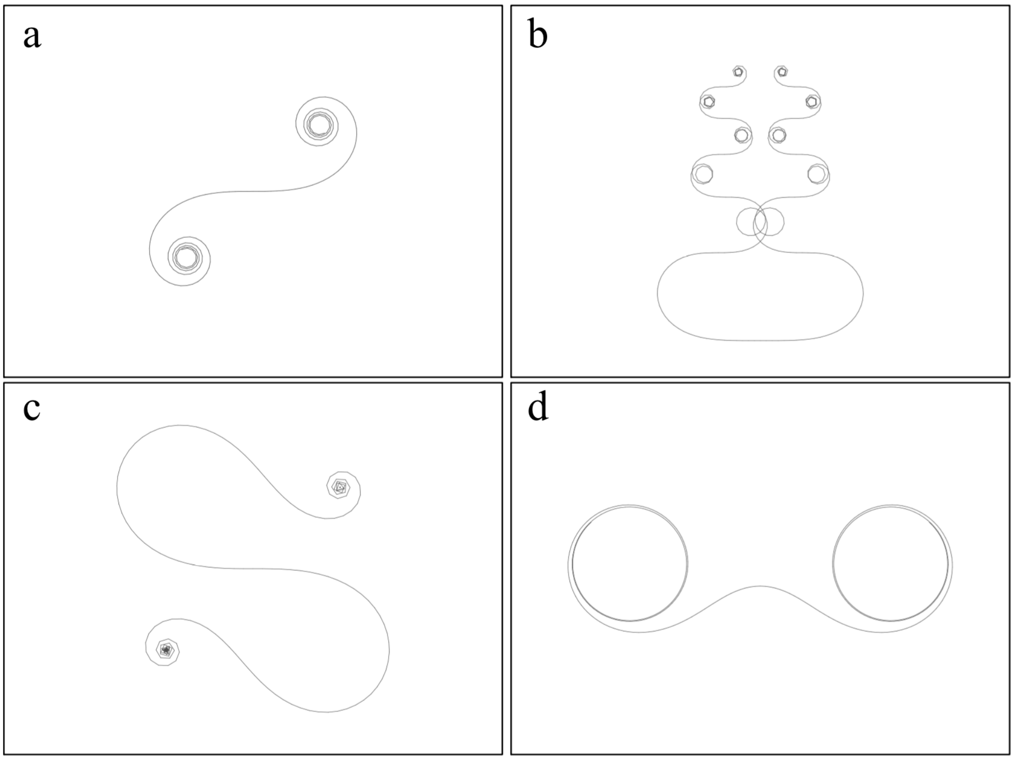

Figure 1: Curves reconstructed from given curvature functions $k_i(s)$.

hires different people and gives them 700kg of gold. The task is to cover the surface of one vertical slice with a layer of gold of 0.1mm thickness. The amount of gold that is left over is the salary for doing the job. Which slice should you pick if you want to make the most profit? Explain your answer in `TheoryExercise.pdf`.

How does your answer change when you have n slices instead of just two?

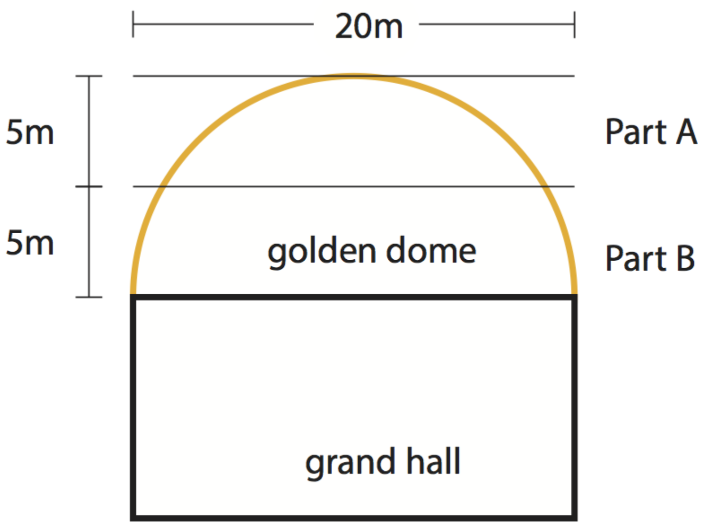

Figure 2: Sketch of King Archimedes dome.

2 Coding Exercise (60 pt)

2.1 The Framework

To load the project, open the CMakeLists.txt from the main directory using Qt Creator. In order to compile the code, you need to have some external libraries installed in your system.

- On Ubuntu (15.04, 15.10, ...) follow these insturctions

```
~$ sudo apt-get install libglew libglew-dev
~$ sudo apt-get install libglfw3 libglfw3-dev
```

- On Mac OSX (10.10.2, 10.11, ...) install `homebrew` from `http://brew.sh` and follow these instructions

```
~$ brew install glfw3
~$ brew install glew
```

- On Windows we provide the binaries for `glfw3` (tested with Visual Studio 2015 and 2017) and `glew`. If you encounter linking errors, you might need to build `glfw` and `glew` on your own. Download the source files from GLFW3 and GLEW, build, and copy the binaries to the folder: `dgp2018-exercise4/externals/{glfw3, glew}/lib`. We provide `glfw3` binaries for Visual Studio Compiler 10, 11, 12 and 14, you can change lines 28 and 31 in `dgp2018-exercise4/cmake/FindGLFW3.cmake` to match with your version. Note that when configuring your project in QT Creator, you need to choose "MSVC2017" rather than "MinGW".

2.1 The Task

For the exercise 4 you will need to fill in the missing code in the file `curves.cpp`. For smoothing use keys `S` and `C` to iterate, and keys `1-4` to display different curves.
Given a curve in $\mathbb{R}^2$ as a closed polyline with vertices $\{V_i\}_{i=1}^{M}$, where $V_1 = V_M$, implement the following two examples of curve smoothing:

- Move every vertex towards the centroid of the two neighbors. More specifically, every vertex V_i should shift to vertex V_i' according to

$$V_i' = (1-\varepsilon)V_i + \varepsilon\frac{V_{i-1} + V_{i+1}}{2} \tag{2}$$

  where ε is a small time step (you can experiment with different values for the time step). Iterate this procedure. After each iteration uniformly scale the curve to its original length around its current centroid.
  Implement this smoothing by filling in the `laplacianSmoothing()` function in the file `curves.cpp`. This function should perform one iteration of smoothing. Press key `S` to run this function and iterate.

- Move every vertex towards the center of the osculating circle. Consider an osculating circle at vertex V_i as a circumscribed circle O of a triangle defined with vertices V_{i-1}, V_i and V_{i+1}. This can be done by shifting every vertex V_i to vertex V_i' according to

$$V_i' = V_i + \varepsilon \, \frac{C - V_i}{\|C - V_i\|^2} \tag{3}$$

where ε is a small time step and C is the center of the circumscribed circle O. Iterate this procedure. After each iteration uniformly scale the curve to its original length around its current centroid.

Implement this smoothing by filling in the `osculatingCircle()` function in the file `curves.cpp`. This function should perform one iteration of smoothing. Press key `C` to run this function and iterate.

To run the smoothing algorithms on different input curves select one of the four curves by pressing keys `1`, `2`, `3` or `4`.