Optimization, Robotics & Biomechanics (ORB)
Institute of Computer Engineering (ziti)
Heidelberg University

Prof. Dr. Katja Mombaur,
Benjamin Reh,
Silvan Lindner

Robotics 1 (WS 2018/2019)

Exercise Sheet 0

Presentation during exercises in calendar week 43

General information

- New sheets are handed out on Tuesday.
- The solutions are to be presented during the exercises the following week:
  - Students declare at the beginning of the exercise which assignments they are able to present the results of.
  - The actual student to present a result will be picked randomly from the above.
  - For admission to the written exam a student must declare a minimum of 50 % of the assignments for presentation.
- Assignments must run on the computers in the CIP-Pool for the presentation of the results.
- **Exception:** This sheet is different from the following sheets. It is meant as an introduction and will be worked on within the exercise in calender week 43. It does not contribute to the exam admission.
- You can connect to your CIP-Pool accounts remotely using SSH at `pool.IWR.uni-heidelberg.de`

Exercise 0.1 – Basic algebra with the `Eigen` library

`Eigen` is a header-only `C++` linear algebra library that will be used in the course. This assignment guides through the first steps by demonstrating simple operations using vectors and matrices.

1. Take a look at the files provided in the `"eigen"` directory. Notice that we are using CMake as build-system for this `C++` project. In order to build a `C++` project with CMake, use the following steps

   (a) Create a `"build"` folder in the exercise folder and navigate to it

```
cd eigen
mkdir build
cd build
```

(b) Run CMake to automatically generate a make-file. This must be invoked only once (and each time the `"CMakeLists.txt"` is changed)

```
cmake ..
```

(c) Execute the make-file to build your code. This step is required each time a project-file has been modified.

```
make
```

This way the generated files and the source files are separated, which is generally good practice. Always work with the files in your source directory, while build files are generated in the build directory after invoking `make`.

2. Familiarize yourself with the provided `"CMakeLists.txt"`.

3. In `Eigen` all vectors are column vectors. The type `Eigen::Vector3d` depicts a column vector with 3 entries of the type `double`. Simliar the type `Eigen::Matrix3d` stands for a 3×3 matrix. A good way to start is looking at the tutorial from `Eigen` `https://dritchie.github.io/csci2240/assignments/eigen_tutorial.pdf` for the following steps.

   (a) Open the `main.cpp` in the editor of your choice. You find already an example on how to define matrices and vectors. Define a rotation matrix $\mathbf{R}$ as given in equation (1). Define a vector $a = (1, 2, 0)^T$ and compute the product $b = \mathbf{R} \cdot a$.
   Around which axis does matrix $\mathbf{R}$ rotate vector a and what is the angle of rotation?

$$\mathbf{R} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

   (b) The vectors a and b should now be orthogonal. Verify that by computing the dot product $d = a \cdot b \stackrel{!}{=} 0$

   (c) Compute the normalized cross product $c = |a \times b|$ of the two vectors. The result should be a vector pointing in the direction of the orientation $c = (0, 0, 1)^T$.

Exercise 0.2 – First steps in MeshUp

In this exercise, you will explore the visualization tool MeshUp . MeshUp uses skeleton based animation and OpenGL to visualize models and 3D motions.

To start MeshUp , open a terminal window and type in `meshup`. Open the example model file `"samplemodel.lua"` and the example animation file `"sampleanimation.txt"`.

Both files are simple text files and can be edited using a standard text editor such as `gedit`.

1. Familiarize yourself with the MeshUp application.

2. Copy the files `"samplemodel.lua"` and `"sampleanimation.txt"` in a seperate folder and edit the animation file to make the model wave with its right arm.

   Hint 1: Press `F5` to load the changes directly in MeshUp .

   Hint 2: `View Settings` allows activating the local coordinate systems. The X, Y and Z axis are red, green and blue respectively.

3. Edit `"samplemodel.lua"` in the following ways:

   (a) Double the size of the head

   (b) Display spheres in the joints. Use the file `"meshes/unit_sphere_medres.obj"` as `src` attribute. Choose a 0.15 radius and a green (0.0, 1.0, 0.0) color.
   Hint: Each frame can be assigned multiple meshes under `"visuals"`. No additional frames need to be created.

   (c) Reduce the length of the arms and legs by a factor of 2 (Note: this means you also have to adjust the values of `joint_frame = { r = { ... } }`)

Notes:

- MeshUp can be downloaded from `https://github.com/ORB-HD/MeshUp`. Some documentation is available there, as well.
- When installing MeshUp from source, the following dependencies need to be installed first:
  ```
  sudo apt install build-essential cmake
  sudo apt install qtbase5-dev liblua5.1-0-dev libboost-all-dev libeigen3-dev
       libavcodec-dev libavformat-dev libswresample-dev libswscale-dev
  ```
- The skeleton semantics for the model visualization is based on Lua, a scripting language `http://www.lua.org`, which makes the model representation highly flexible.