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 4

Presentation during exercises in calendar week 48

Exercise 4.1 – Kinematics with RBDL

The Rigid Body Dynamics Library RBDL is an efficient and powerful tool which we will use to compute forward and inverse kinematics (IK) and later on in the lectures also dynamics. RBDL is installed on the computers in the pool.

Unfortunately, the RBDL version installed on the VirtualBox image is the stable version. For this exercise the `dev`*-branch is needed. Notes on how to install RBDL from the* `dev`*-branch is provided at the end of this sheet.*

1. For this exercise we consider the IK algorithm as a black box. An initial value q_{start} for the joint positions is given and the algorithm tries to find the best solution q to fullfill the given position and/or orientation constraints.

   From the lecture you know that IK problems can have a different number of solutions. Describe in word, in which situations the following number of solutions occur:

   (i) No solution
   (ii) 1 solution
   (iii) $2, \dots, n < \infty$
   (iv) A continuum of solutions

2. Download and build the provided code snippet in `kuka_rbdl`. If the compilation succeeds and you can run the executable, you are able to use RBDL .

   Start meshup to visualize the result of the computation:

```
meshup kuka+cube.lua animation.csv
```

   You should be able to see the Kuka robot following the trajectory of a red cube.

3. Take a look at the provided code and try to understand what it does:
   - Inside the `kuka+cube.lua` model file there is a "mechanic" defined to rotate a translated red cube around the x-axis.
   - In the first step, a forwards kinematic computation is performed to obtain the position (3d vector) and orientation (3×3 matrix) of that cube.
   - The second step is to set up the IK for the Kuka robot. The position and orientation of the cube is set as contraints for the endeffector (TCP) and the IK is computed.
   - This is repeatedly performed for different rotations α of the cube.

4. Explore the given code to familiarize yourself with different aspects of the IK problem:

   a) Increase the z value of the `joint_frame` attribute in segment `l2` in the lua file to 0.5. Do not forget to re-run the program.

   When reaching the highest point at around 2 s an interesting effect can be observed. Which problem of IK occurs here?

   b) Change the initial `q_start` from $[0, -\frac{\pi}{2}, \frac{\pi}{2}, \pi, 0, 0,]$ to $[0, \frac{\pi}{2}, -\frac{\pi}{2}, \pi, 0, 0,]$. Observe the changes in the trajectory. Does it solve the problem from a) ? What new problem does this trajectory have?

   c) Disable one of the contraints while leaving the other active. What can be seen?

   d) Remove the line

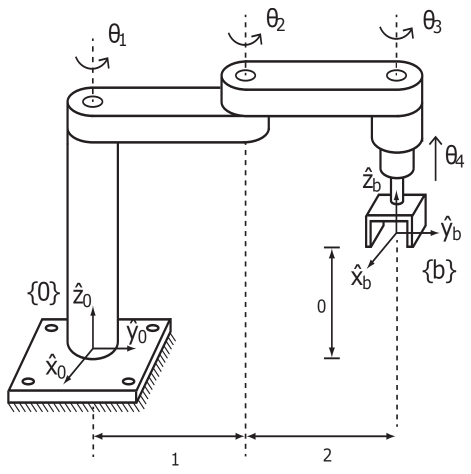
   ```
   q_start = q;
   ```

   Why do the poses of the robot jump around?

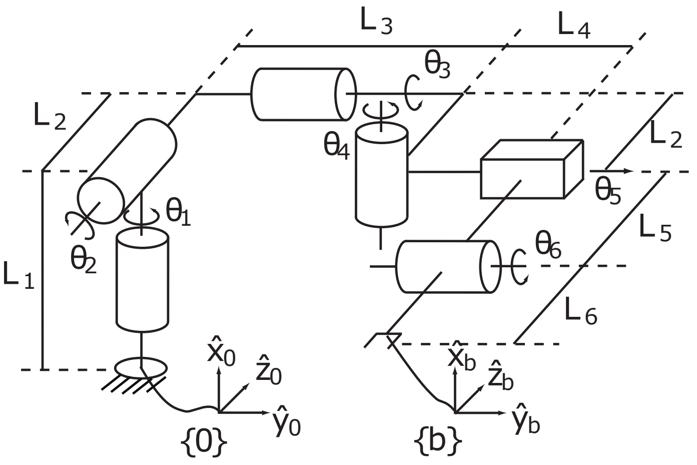
Exercise 4.2 – Homogeneous Transformations

1. The SCARA robot given in figure 1a is shown in its zero position. Determine the end-effector position dependent on θ_1, θ_2, θ_3 and θ_4 in word-frame by calculating forward kinematics using homogeneous coordinates.

2. The spatial open chain mechanism of figure 1b is shown in its zero position, with fixed and end-effector frames chosen as indicated. Determine the end effector zero position configuration.



(a) An RRRP SCARA robot.



(b) A spatial RRRRPR open chain.

Figure 1: Collection of mechanisms (R: revolute joint, P: prismatic joint).

Exercise 4.3 – Denavit-Hartenberg Parametrization

Attaching reference frames to the links of a spatial kinematic chain solely using homogeneous transformations is an ambiguous task. Each transformation consists of 6 degrees-of-freedom and relations between consecutive frames (e.g. rotation axis) need to be defined.

Rather than attaching reference frames to each link in an arbitrary fashion, the Denavit-Hartenberg convention (see lecture) defines a set of rules for assigning reference frames to a kinematic chain consisting of one-degree-of-freedom joints. Each of those frames can be uniquely defined with a set of only 4 parameters, called *Denavit-Hartenberg parameters.*

i	a_i	d_i	α_i	ϑ_i
1	0	0	0	ϑ_1
2	R_2	0	$\pi/2$	ϑ_2
3	0	D_3	0	ϑ_3
4	R_4	0	$\pi/2$	ϑ_4

Table 1: Denavit-Hartenberg parameter

1. Make a sketch of a robot that is described by the Danavit-Hartenberg parameters from table 1. Draw all frame-axes and visualize the parameters $\alpha \neq 0$ (rotation axes), as well as displacements D_3, R_2 and R_4.

2. Use the Denavit-Hartenberg parametrization from table 1 to calculate all homogeneous transforms between successive DH-frames. How could you compute the full forward kinematic?

Installing RBDL

- Prerequisites for the installation are the same as given in sheet00 for MeshUp . Additionally mercurial (`hg`) is needed.
- Download the repository using mercurial:

```
hg clone https://bitbucket.org/rbdl/rbdl
cd rbdl
```

- Checkout the `dev` branch:

```
hg checkout dev
```

- Build and install RBDL with the luamodel-Plugin:

```
mkdir build
cd build
cmake .. -DRBDL_BUILD_ADDON_LUAMODEL=ON
make
sudo make install
```