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

Prof. Dr. Katja Mombaur,
Kevin Stein

Robotics 2 (SS 2019)

Exercise Sheet 1

Presentation during exercises in calendar week 20

Exercise 1.1 – Create your own humanoid model

Based on the double pendulum from the first exercise and the provided skeleton file, create your own humanoid robot model.

As shown in 1 the robot should have 15 degrees of freedom (`DoF`) and be connected to the `ROOT_Link` by a 6 `DoF` floating-base. Combine the hip and thigh joint into a single joint.

Assign reasonable masses to the segments, place the center of mass in the middle of each segment and compute the inertia matrices. Use the provided sample animation to demonstrate that your model is correct.

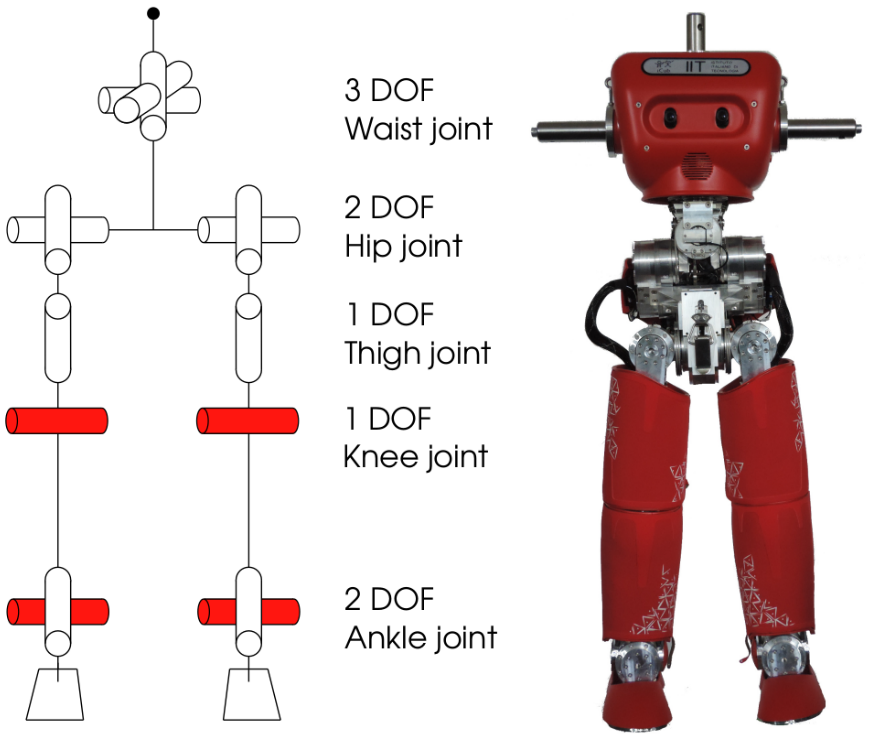

Figure 1: The iCub humanoid robot

Exercise 1.2 – Check your model

Load your model into RBDL and compute the following quantities for all joints being in zero position and compare them to what you expect from your model file:

- Total mass of the robot
- Number of `DoF`
- `CoM` position
- Distance between the feet
- Height of the pelvis above the feet

Helpful functions:

```
    CalcBodyToBaseCoordinates (Model Model, VectorNd Q, int
       body_id, Vector3d body_point_position)
    CalcCenterOfMass (Model model, VectorNd q, VectorNd qdot,
       VectorNd * qddot, double mass, Vector3d com)
```

Exercise 1.3 – Make your robot move

In this exercise we want to compute a simple motion for our humanoid. We use inverse kinematics to compute joint angle trajectories for given end-effector trajectories. Our robot should perform a squatting movement while rotating the pelvis. Set up the `InverseKinematicsConstraintSet` and add a constraint for each end-effector that you want to control. The feet should be flat on the ground. The Pelvis should follow a sinus trajectory in up and down direction and change it's orientation around the z-Axis. The base link should be parallel to the ground.

Helpful functions:

```
    InverseKinematicsConstraintSet CS;
    CS.AddFullConstraint(int body_id, Vector3d
       body_point_position, Vector3d target_position, Matrix3d
       target_orientation);
    CS.AddOrientationConstraint(int body_id, Matrix3d
       target_orientation);
    CS.AddPointConstraint(int body_id, Vector3d
       body_point_position, Vector3d target_position)
    InverseKinematics(Model model, VectorNd q_init,
       InverseKinematicsConstraintSet CS, VectorNd q_res);
```

Check your results in MeshUp. Use the arrows feature of MeshUp to visualize the `CoM` position for each frame.

Arrows

Meshup offers the possibility to draw linear and rotational arrows, called ”forces” and ”torques” in the MeshUp nomenclature. This interface is a bit peculiar, so please be aware of the following:

- The values for arrows are written in a separate CSV-file that must have the `.ff` ending. The format is
  `t, plx, ply, plz, mlx, mly, mlz, prx, pry, prz, mrx, mry, mrz .`

  `t` is the time, `plx` is the x-position of the linear arrow, `mly` is the y-magnitude (length) of the linear arrow, and `mrz` is the magnitute of the rotational arrow for z, and so forth. . .
- To load them, start MeshUp with
  `meshup modelfile.lua animationfile.txt arrowfile.ff`
- The position of the arrow is defined by its tip.
- The magnitude of the arrows are scaled down internally. You will have to scale them up until you can see the arrows.
- Although Meshup is in use within the group for several years, there are still bugs which will eventually be fixed.
  - When using arrows, disable shadows as it sometimes prevens arrows from being visible.
  - Several bugs already have been fixed since the beginning of this lecture. It is advisable to use the latest version of Meshup.

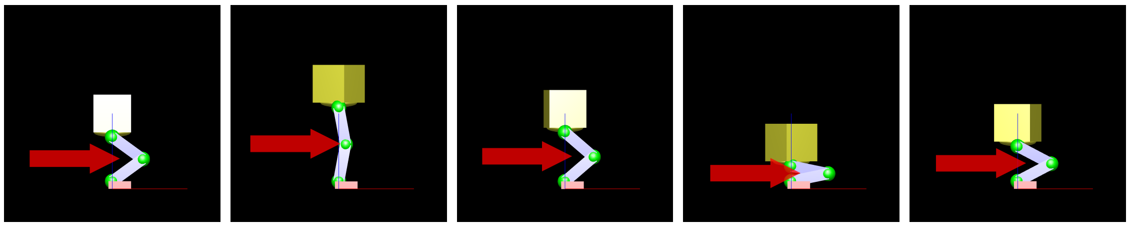

Figure 2: A squatting robot shaking it’s head

Notes:

- RBDL can be downloaded from `https://github.com/ORB-HD/rbdl-orb` Some documentation is available there, as well.
- The documentation is contained in the code and can be extracted with the tool doxygen. To create the documentation simply run

  ```
  doxygen Doxyfile
  ```

  which will generate the documentation in the subdirectory ./doc/html. The main page will then be located in ./doc/html/index.html.