RobWork Workcell Structure and Programming
Exercise 3.3

Guðmundur Geir Gunnarsson

Univeristy of Southern Denmark

gunu@mmmi.sdu.dk

September 12, 2017

Overview

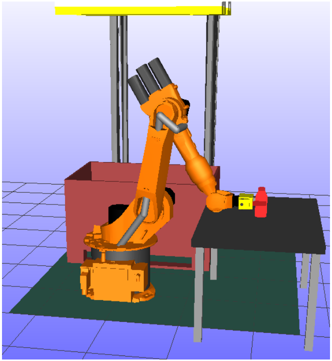
Programming Exercise 2.2

RobWork Workcell Structure

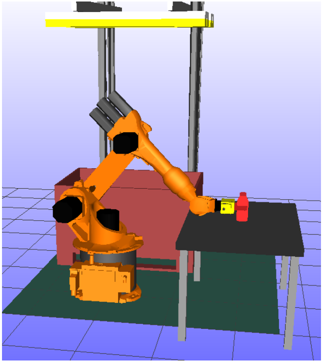
Programming Exercise 3.3

Announcement

Programming Exercise 2.2



- `q={1.713,-1.395,2.415,-2.975,1.027,-1.658}`
- `q={1.713,-1.395,2.415,0.165,-1.028,1.482}`

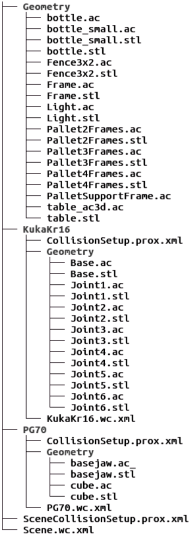


- `q={-1.427,-2.287,-1.608,0.208,0.764,-1.722}`
- `q={-1.426,-2.288,-1.608,3.352,-0.766,1.418}`

- Additional solutions if joint limits are relaxed

RobWork Workcell Structure

- A workcell consists of:
  - Geometries
  - Devices
  - Scene definitions (Frame definitions)
  - Collision Setup
- Each device is structured as a workcell



```
├── Geometry
│   ├── bottle.ac
│   ├── bottle_small.ac
│   ├── bottle_small.stl
│   ├── bottle.stl
│   ├── Fence3x2.ac
│   ├── Fence3x2.stl
│   ├── Frame.ac
│   ├── Frame.stl
│   ├── Light.ac
│   ├── Light.stl
│   ├── Pallet2Frames.ac
│   ├── Pallet2Frames.stl
│   ├── Pallet3Frames.ac
│   ├── Pallet3Frames.stl
│   ├── Pallet4Frames.ac
│   ├── Pallet4Frames.stl
│   ├── PalletSupportFrame.ac
│   ├── table_ac3d.ac
│   └── table.stl
├── KukaKr16
│   ├── CollisionSetup.prox.xml
│   ├── Geometry
│   │   ├── Base.ac
│   │   ├── Base.stl
│   │   ├── Joint1.ac
│   │   ├── Joint1.stl
│   │   ├── Joint2.ac
│   │   ├── Joint2.stl
│   │   ├── Joint3.ac
│   │   ├── Joint3.stl
│   │   ├── Joint4.ac
│   │   ├── Joint4.stl
│   │   ├── Joint5.ac
│   │   ├── Joint5.stl
│   │   ├── Joint6.ac
│   │   └── Joint6.stl
│   └── KukaKr16.wc.xml
├── PG70
│   ├── CollisionSetup.prox.xml
│   ├── Geometry
│   │   ├── basejaw.ac_
│   │   ├── basejaw.stl
│   │   ├── cube.ac
│   │   └── cube.stl
│   └── PG70.wc.xml
├── SceneCollisionSetup.prox.xml
└── Scene.wc.xml
```

Tasks for today

- Do Programming Exercise 3.3
- Construct a RobWork workcell with a UR robot manipulator
- Geometries are from CAD a file
- Use datasheet (on BlackBoard) to get measurements
- Download workcell `UR5WorkCellCut.zip` from BlackBoard
- Edit the `Device.wc.xml` file

RobWork XML files

- Frame definitions
  - Positions: x, y, z (red, green, blue) in $[m]$
  - Rotations: RPY $(\theta_z, \theta_y, \theta_x)$ in $[Deg]$
  - Type: Revolute or prismatic
- Joint limits: Have already been set
- Drawables
  - Graphics for a joint
  - `refframe` gives the coordinate frame for the graphics
  - Pose is relative to `refframe`
  - **WARNING:** The pose of the graphics objects is given in absolute coordinates w.r.t. the robot

Programming Exercise 3.3

- Guide to the first two joints.
- Based on slides by Lars Carøe Sørensen

Programming Exercise 3.3

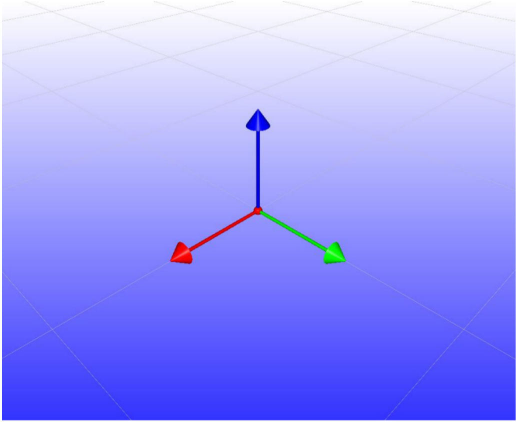


Figure: World/Robot/Base frame

Programming Exercise 3.3

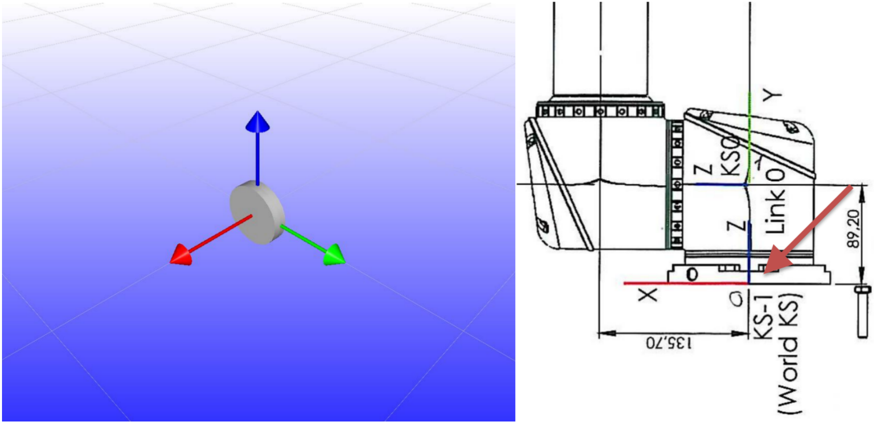


Figure: Insert robotFlange and base (all pos and rot zero)

Programming Exercise 3.3

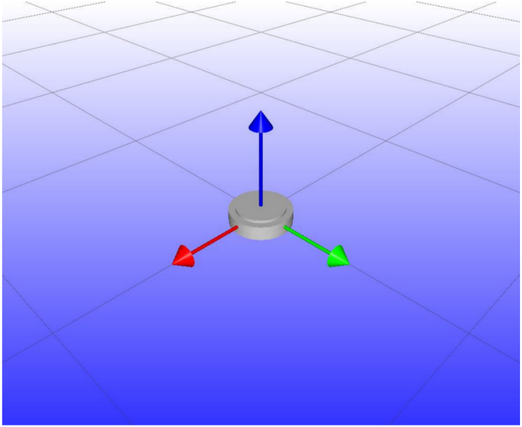


Figure: Drawable: rotate 90° about y ($P = 90^\circ$)

Programming Exercise 3.3

- Base and robotFlange in place. XML is:
-
  ```
  <Drawable name="flangeGeo" refframe="Base">
  <RPY> 0 90 0</RPY> <Pos> 0 0 0</Pos>
  <Polytope file="geometry/robotFlange" />
  </Drawable>
  <Drawable name="flangeGeo" refframe="Base">
  <RPY> 0 90 0</RPY> <Pos> 0 0 0</Pos>
  <Polytope file="geometry/base" />
  </Drawable>
  ```

Programming Exercise 3.3

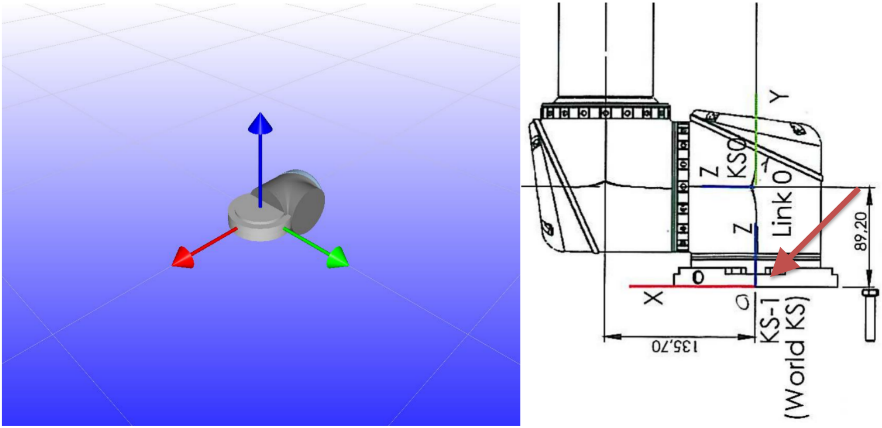


Figure: Insert Joint0 (all pos and rot zero)

Programming Exercise 3.3

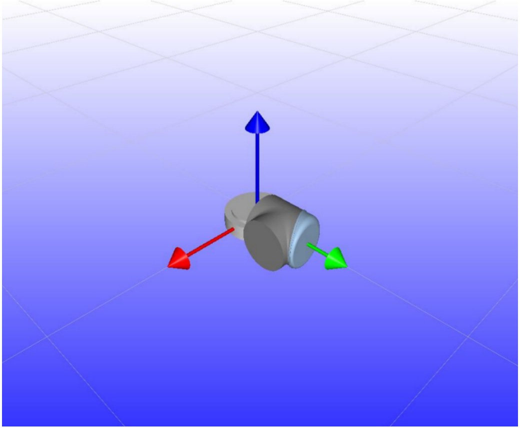


Figure: Drawable0: rotate 270° around z ($R = 270°$)

Programming Exercise 3.3

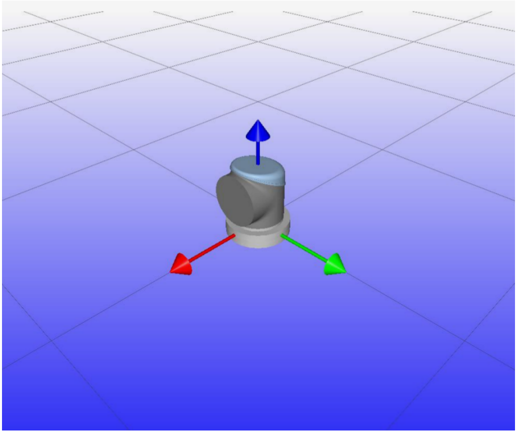


Figure: Drawable0: rotate 90° around y ($P = 90°$)

Programming Exercise 3.3

- Joint0 in place

-
  ```
  <Joint name="Joint0" type="Revolute">
  <RPY> 0 0 0 </RPY> <Pos> 0 0 0 </Pos>
  </Joint>
  <Drawable name="Joint0Geo" refframe="Joint0">
  <RPY> 270 90 0 </RPY> <Pos> 0 0 0 </Pos>
  <Polytope file="Geometry/joint0" />
  </Drawable>
  <Q name="Home">0</Q>
  ```

Programming Exercise 3.3

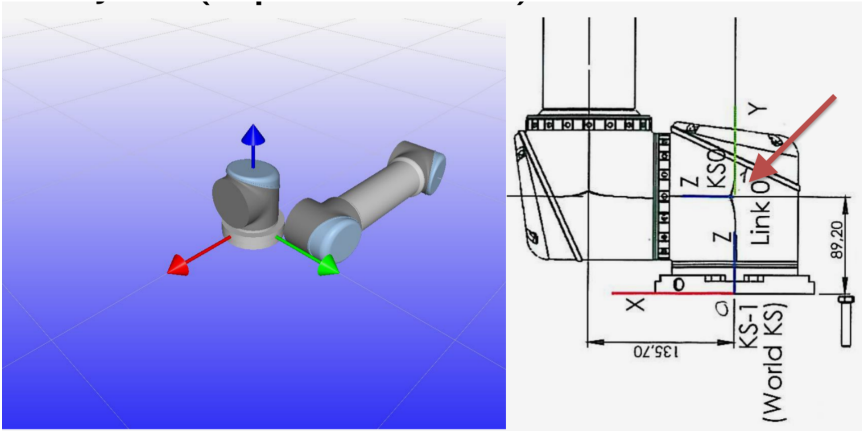


Figure: Insert Joint1 (all pos and rot zero!)

Programming Exercise 3.3

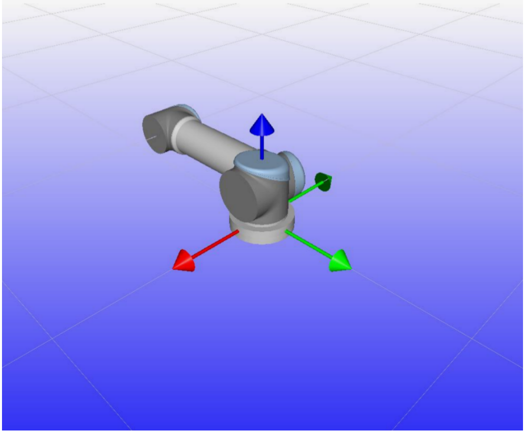


Figure: Joint1: rotate frame ($R = 90^{\circ}$)

Programming Exercise 3.3

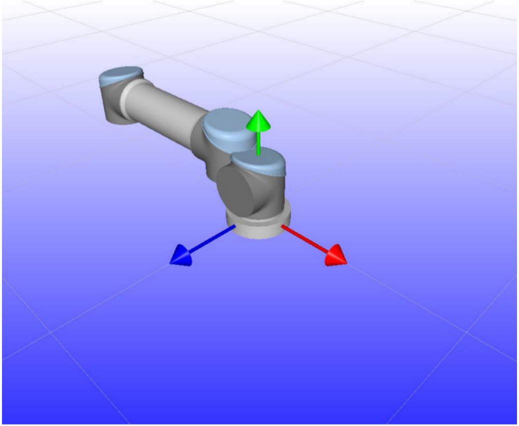


Figure: Joint1: rotate frame ($Y = 90^\circ$)

Programming Exercise 3.3

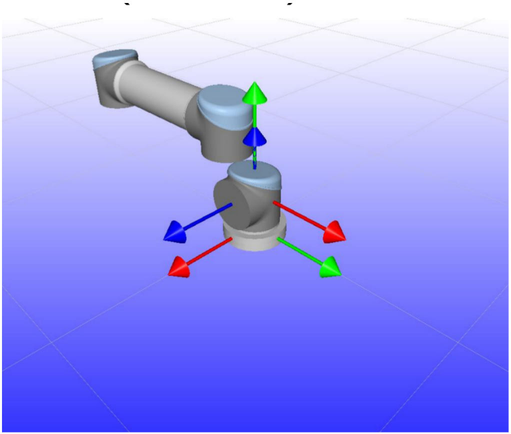


Figure: Joint1: move frame ($z = 0.08920$)

Programming Exercise 3.3

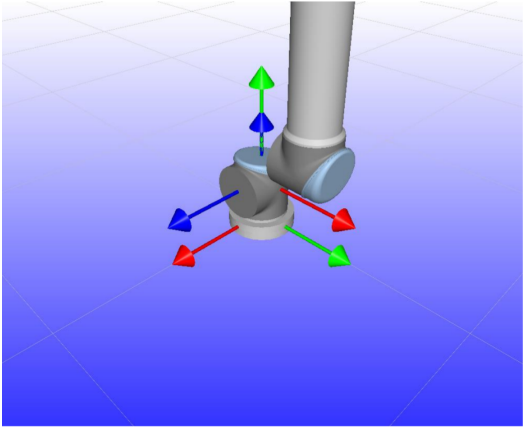


Figure: Drawable1: rotate drawing ($R = 270^{\circ}$)

Programming Exercise 3.3

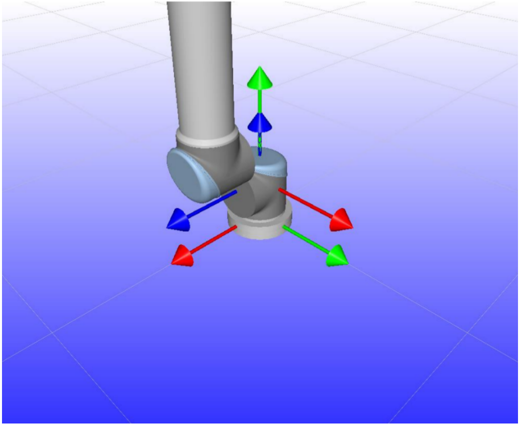


Figure: Drawable1: rotate drawing ($Y = 90°$)

Programming Exercise 3.3

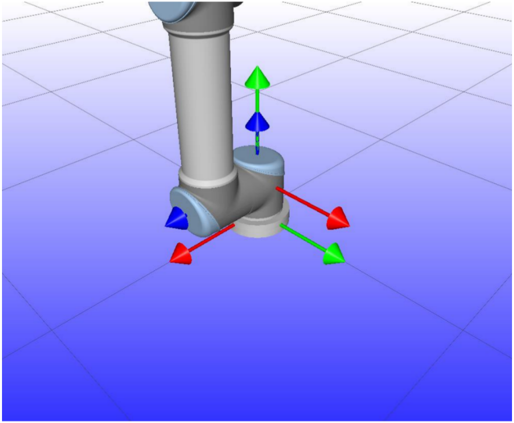


Figure: Drawable1: move drawing ($y = -0.08920$)

Programming Exercise 3.3

- Joint1 in place.
-
  ```
  <Joint name="Joint1" type="Revolute">
  <RPY> 90 0 90 </RPY> <Pos> 0 0 0.0892 </Pos>
  </Joint>
  <Drawable name="Joint1Geo" refframe="Joint1">
  <RPY> 270 0 90 </RPY> <Pos> 0 -0.0892 0</Pos>
  <Polytope file="Geometry/joint1"/>
  </Drawable>
  <Q name="Home">0 0</Q>
  ```

Tips

- Be systematic in your approach. Either:
  - Rotations before positions
  - Positions before rotations
- Remember to make the home Q vector (end of XML) the right size
- Use the diagram from the datasheet for:
  - Dimensions of the robot
  - Position/Orientation of frames
- There are small misalignments in the drawables. Ignore these!

Groups

- Groups for the mandatory exercises and the project
- Before you leave today, you have to tell me who you are in a group with
- Two people per group
- Send me an email with the name and email addresses of your group members
- Send to `gunu@mmmi.sdu.dk`