Übungen zu "Autonomous Grasping"
WS 2017/18 Blatt 9

Abgabe: 6.9.2019

The following tasks will use Neo/NST. While `neo` is available from the `rcinfo` package `nst`, we need to access the 32bit version here. For this reason, please use the shell script in `/vol/ni/share/lehre/robotik/grasping/neo/neo.sh`.

In preparation of this assignment, copy the whole folder `/vol/ni/share/lehre/robotik/grasping/neo` to your home directory and work therein. Neo/NST loads additional units listed in the file `.neofolders` that should be accessible in the current directory.

```
cp -r /vol/ni/share/lehre/robotik/grasping/neo ˜/grasping
cd  /grapsing
```

Neo/NST is a graphical programming environment. Please familiarize with Neo/NST by loading and playing around with an initial circuit:

```
./neo.sh -i arm.NST
```

The unit `gui_loop` is executed as soon as the mouse pointer is within the `NST:gui` window. This unit in turn executes its operands marked by the red rubber frame, i.e.

- calling the CBF controller (`cbf`)
- fetching the current joints (`cbf:res`)
- actuating the simulated model (`prog`)
- render the model (`vx:Render:Update`)

The `cbf` controller loads its controller from an `XML` file. When clicking the little 'm' you can modify the unit settings, namely choose the `XML` file, reference which simulation scene (`vx:scene`), and which kinematic tree to use. The following kinematic trees are available:

- [Right—Left].arm: up to the `ToolFrame`
- [Right—Left].arm-with-hand: including the hand (as a kinematic tree)
- arms: both arms up to the `ToolFrame` (as a kinematic tree)
- arms-with-hands: both arms with hands attached

Aufgabe 9.1, Change Endeffector: Starting from circuit `arm.NST`, switch from finger tip control to arm control only. To this end:

- In file `xml/arm.xml`, in the `SensorTransform` element, change the `SegmentName` to `Right.ToolFrame` (instead of `Right.hand.ff1`) and save as new file.
- In the circuit, change to the new XML file, **m**odifying the unit `cbf`.

Try other segments of the robot as well, e.g. the elbow (Right.arm.sSeg5) or other finger tips (Right.hand.mf1, Right.hand.th1). You can retrieve the segment names by pointing to them in the visualization window and pressing "i". Then, the name is printed on the console. What do you observe, when controlling different end effectors, going upward the kinematic chain?

Aufgabe 9.2, Redundancy Control: Replace the current redundancy controller (`SubordinateController`) by one, which controls the shoulder position (Right.arm.sSeg3). Check the syntax of your xml with:

```
xmllint –schema /homes/rhaschke/src/cbf/schemas/schemas.xsd ¡your xml file¿
```

In the neo circuit, connect the second input of _init:set_target to a new `use_method` unit referencing your new sub controller (something like `cbf:NS:<name>:set`, lookup the name beforehand). Hint: you can dive into a container unit by clicking the 'o'. You escape from a container by right-clicking on the background.

Aufgabe 9.3, Parallel Control: Instead of using a `SubordinateController`, you could also run both controllers with same priority level, side by side. To this end, use `CompositeReference`, `CompositePotential`, and `CompositeSensorTransform` elements. They always expect elements of like type as children. Remember to adapt the `TaskDimension` of the `EffectorTransform`. Which difference in behaviour you observe?