Skills for mobile manipulation

—

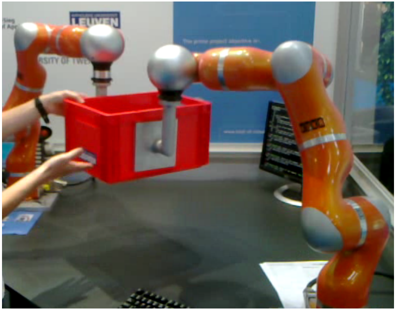
(To ROS or not to ROS)

Herman Bruyninckx
K.U.Leuven, Mechanical Engineering, Belgium
`http://people.mech.kuleuven.be/~bruyninc/`

München, November 6, 2010

Overview

- My profile
- The Task – Skill – Motion trinity
  (a.k.a.: *How to do this manipulation in ROS?*)



- Motion & Skill specification, sensing and control

My profile

- Mathematical Physics, CS, Mechatronics

My profile

- Mathematical Physics, CS, Mechatronics
- 1998: decided to make *open source software engineering* 50% of my academic profile

My profile

- Mathematical Physics, CS, Mechatronics
- 1998: decided to make *open source software engineering* 50% of my academic profile
- was lured by three machine tool vendors into conceiving Orocos, on New Year's Eve 2000

My profile

- Mathematical Physics, CS, Mechatronics
- 1998: decided to make *open source software engineering* 50% of my academic profile
- was lured by three machine tool vendors into conceiving Orocos, on New Year's Eve 2000
- excited about PR2 + ROS, but...

My profile

- Mathematical Physics, CS, Mechatronics
- 1998: decided to make *open source software engineering* 50% of my academic profile
- was lured by three machine tool vendors into conceiving Orocos, on New Year's Eve 2000
- excited about PR2 + ROS, but. . . I'm old enough to have made all of ROS's mistakes!

My profile

- Mathematical Physics, CS, Mechatronics
- 1998: decided to make *open source software engineering* 50% of my academic profile
- was lured by three machine tool vendors into conceiving Orocos, on New Year's Eve 2000
- excited about PR2 + ROS, but... I'm old enough to have made all of ROS's mistakes!
- read enough to know *prior art* for everything

My profile

- Mathematical Physics, CS, Mechatronics
- 1998: decided to make *open source software engineering* 50% of my academic profile
- was lured by three machine tool vendors into conceiving Orocos, on New Year's Eve 2000
- excited about PR2 + ROS, but… I'm old enough to have made all of ROS's mistakes!
- read enough to know *prior art* for everything
- fit enough to keep on fighting reinvented wheels

My profile

- Mathematical Physics, CS, Mechatronics
- 1998: decided to make *open source software engineering* 50% of my academic profile
- was lured by three machine tool vendors into conceiving Orocos, on New Year's Eve 2000
- excited about PR2 + ROS, but. . . I'm old enough to have made all of ROS's mistakes!
- read enough to know *prior art* for everything
- fit enough to keep on fighting reinvented wheels
- Zen enough to search for vision & critical mass in ±100 open source projects a year. . .

My profile

- Mathematical Physics, CS, Mechatronics
- 1998: decided to make *open source software engineering* 50% of my academic profile
- was lured by three machine tool vendors into conceiving Orocos, on New Year's Eve 2000
- excited about PR2 + ROS, but... I'm old enough to have made all of ROS's mistakes!
- read enough to know *prior art* for everything
- fit enough to keep on fighting reinvented wheels
- Zen enough to search for vision & critical mass in ±100 open source projects a year...
- ...and smart enough to look *outside* of robotics!

My profile (2)

- stubborn enough to have a (strong) vision

My profile (2)

- stubborn enough to have a (strong) vision
- philosophical enough to recognize *paradigm mismatches* (a.k.a., “religious wars”)

My profile (2)

- stubborn enough to have a (strong) vision
- philosophical enough to recognize *paradigm mismatches* (a.k.a., "religious wars")
- modest enough to realise that my visions have had to change every 3–4 years...

My profile (2)

- stubborn enough to have a (strong) vision
- philosophical enough to recognize *paradigm mismatches* (a.k.a., “religious wars”)
- modest enough to realise that my visions have had to change every 3–4 years...
- connected enough to defend *multi-vendor system building* as highest exploitation priority

My profile (2)

- stubborn enough to have a (strong) vision
- philosophical enough to recognize *paradigm mismatches* (a.k.a., "religious wars")
- modest enough to realise that my visions have had to change every 3–4 years. . .
- connected enough to defend *multi-vendor system building* as highest exploitation priority
- my three worst *déja vus* in robotics:
  1. Shakey (SRI, 1980s) → PR2 on joystick control (2010)

My profile (2)

- stubborn enough to have a (strong) vision
- philosophical enough to recognize *paradigm mismatches* (a.k.a., "religious wars")
- modest enough to realise that my visions have had to change every 3–4 years...
- connected enough to defend *multi-vendor system building* as highest exploitation priority
- my three worst *déja vus* in robotics:
  1. Shakey (SRI, 1980s) → PR2 on joystick control (2010)
  2. Transputers (1990) → ROS (2010)

My profile (2)

- stubborn enough to have a (strong) vision
- philosophical enough to recognize *paradigm mismatches* (a.k.a., "religious wars")
- modest enough to realise that my visions have had to change every 3–4 years...
- connected enough to defend *multi-vendor system building* as highest exploitation priority
- my three worst *déja vus* in robotics:
  1. Shakey (SRI, 1980s) → PR2 on joystick control (2010)
  2. Transputers (1990) → ROS (2010)
  3. *Proceedings of the International Symposium on Industrial Robotics* (1980s) → current robotics conferences and journals

My profile (3)

- personal *best paper* award: Frederick P., Brooks. *No silver bullet. Essence and accidents of software engineering*, IEEE Computer, 1987.

  See also: *The mythical man month*
- Why. . . ?

My profile (3)

- personal *best paper* award: Frederick P., Brooks. *No silver bullet. Essence and accidents of software engineering*, IEEE Computer, 1987.

  See also: *The mythical man month*
- Why. . . ? *"Putting more developers on a* **late** *software project will only make it later."*

  Replace "*late*" by any other quality measure for software. . .

My profile (3)

- personal *best paper* award: Frederick P., Brooks. *No silver bullet. Essence and accidents of software engineering*, IEEE Computer, 1987.

  See also: *The mythical man month*
- Why...? *"Putting more developers on a* **late** *software project will only make it later."*

  Replace "*late*" by any other quality measure for software...
- personal *role models* in open source projects:
  - Linux kernel (+ *Linux Weekly News*, *KernelNewbies*)
  - Apache
  - Eclipse + OSGi

My profile (4)

- personal *best practice* favourite: *three levels of abstraction* as architecture for all robotic systems

My profile (4)

- personal *best practice* favourite: *three levels of abstraction* as architecture for all robotic systems
- research credo: *every robotics solution is a (softly) constrained optimization, in need of (cognitive) coordination*

My profile (4)

- personal *best practice* favourite: *three levels of abstraction* as architecture for all robotic systems
- research credo: *every robotics solution is a (softly) constrained optimization, in need of (cognitive) coordination*
- favourite *software benchmark*: bi-directional, inter-project reusability and composability

My profile (4)

- personal *best practice* favourite: *three levels of abstraction* as architecture for all robotic systems
- research credo: *every robotics solution is a (softly) constrained optimization, in need of (cognitive) coordination*
- favourite *software benchmark*: bi-directional, inter-project reusability and composability
- carreer ambitions:
  - to make open source the best *professional* choice in robotics
  - *paradigm shift* in motion *specification*
  - *citation index* for robotics software
  - stimulate fair citation *attitude*

My profile (6)

- best recent robotics quote:

  Don't abstract the physics away!
  Christoph Borst (DLR), October 2010

My profile (6)

- best recent robotics quote:

  Don't abstract the physics away!
  Christoph Borst (DLR), October 2010

- most short-sighted ROS quotes:
  ROS is supported by the whole community!
  ROS accelerates robotics research!
  If we are not in ROS, we do not exist!
  Every One (Soitseems), 2009–2010

My profile (6)

- best recent robotics quote:

  Don't abstract the physics away!
  Christoph Borst (DLR), October 2010

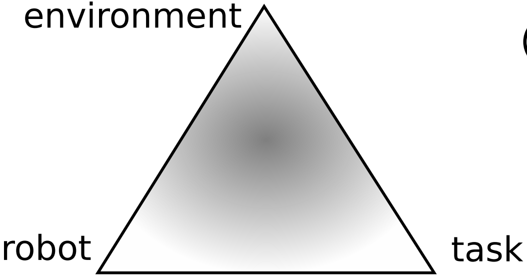
- most short-sighted ROS quotes:
  ROS is supported by the whole community!
  ROS accelerates robotics research!
  If we are not in ROS, we do not exist!
  Every One (Soitseems), 2009–2010

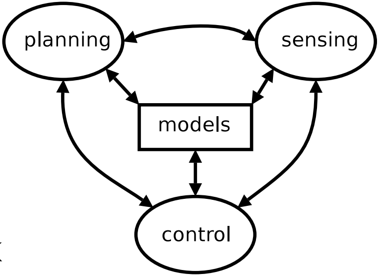
- my (short-sighted) ROS quote:

  ROS: I love it, but I hate it. . . !
  Herman Bruyninckx (Munich), 2010

My profile (4)

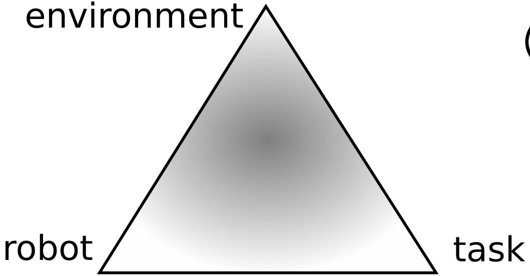
- most useful figures in my robotics classes:

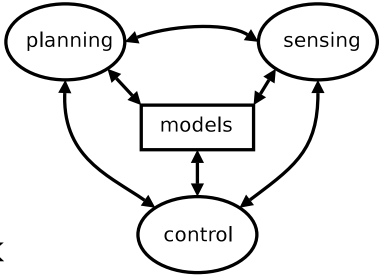




My profile (4)

- most useful figures in my robotics classes:





- my benchmark of intelligence & cognition:

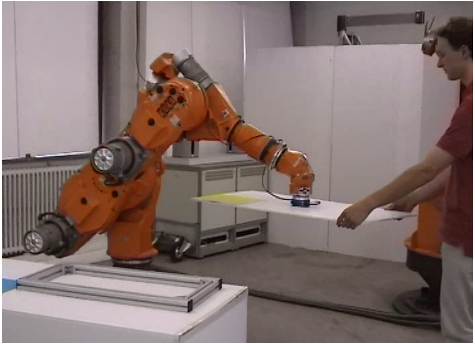
  to explain more with less concepts

- my current research hypothesis:

  cognitive models are the best software models

Mobile manipulation

Another example of *"How to do this in ROS?"*



How to extend this to **real mobile manipulation** on the PR2? And other similar robots!

ROS = Sense–Plan–Act

- **very few** good "control" functionalities
  (⇒ not grounded in traditional robotics. . .)
- emphasis on (only) **geometric, static planning**
- **uni-directional, input–output, hard set-point** "stack" hierarchies
- poor **"4C" separation of concerns**:

  Computation, Communication, Configuration, Coordination

ROS = Sense–Plan–Act

- **very few** good "control" functionalities
  (⇒ not grounded in traditional robotics...)
- emphasis on (only) **geometric, static planning**
- **uni-directional, input–output, hard set-point** "stack" hierarchies
- poor **"4C" separation of concerns**:

  Computation, Communication, Configuration, Coordination

⇒ PR2 moves like a robot...

4C: separation of concerns

C1 **Computation**

C2 **Communication**

C3 **Configuration**

C4 **Coordination**

4C: separation of concerns

C1 **Computation**: the useful functionality within the components that (hopefully) *pays the bill*

C2 **Communication**: overhead of supporting components/nodes to exchange data

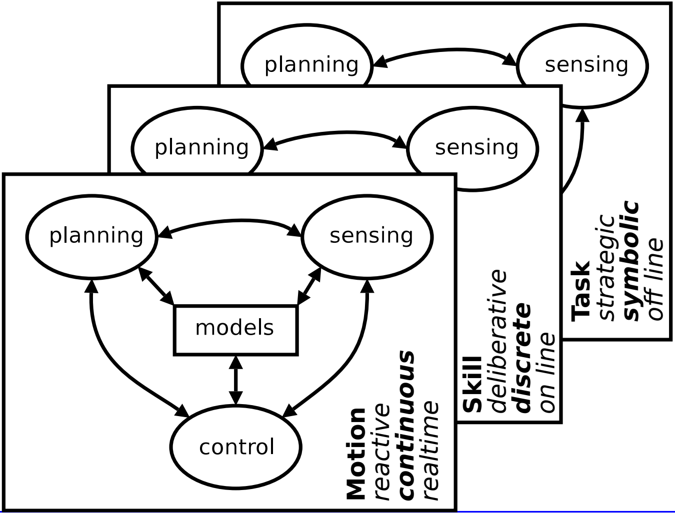
C3 **Configuration**: to give each component appropriate (task-dependent!) parameters

C4 **Coordination**: to help components in switching their functional behaviour in a coordinated way

Holds for hardware, algorithms, middleware,. . . !

Task—Skill—Motion

Traditional **three levels** of robot "architectures"



Three-level (meta) architecture

- first(?) **explicit** description: Saridis 1977
  - *Organization, Coordination, Direct control*
  - *Increasing order of intelligence, decreasing order of precision*
- is known under **various other names** (e.g., *strategic, deliberative, reactive*,...)
- research challenge for coming decade: to weave **more reasoning/intelligence** through all levels

My definitions

- **Motion**:
- **Skill**:
- **Task**:

My definitions

- **Motion**: a **continuous** **time/space activity** of a robot, moving its **joints and/or tool(s)** in a specified way, **until some constraint is violated** that can be **checked by sensors**.
- **Skill**:
- **Task**:

My definitions

- **Motion**: a **continuous time/space activity** of a robot, moving its **joints and/or tool(s)** in a specified way, **until some constraint is violated** that can be **checked by sensors**.
  (Extremely simple) examples:
  - a **force-controlled peg-in-hole** motion, terminated by reaching a force threshold in the insertion direction
  - a **force-guarded approach motion** in free space, terminated by sensing a non-zero approach force.
- **Skill**:
- **Task**:

My definitions

- **Motion**:
- **Skill**: a **discrete** state automaton (FSM), in which each State **runs** one single **Motion**, and each violation of a motion constraint (can) give rise to a **transition** **event**.
- **Task**:

My definitions

- **Motion**:
- **Skill**: a **discrete** state automaton (FSM), in which each State **runs** one single **Motion**, and each violation of a motion constraint (can) give rise to a **transition event**.
  (Extremely simple) examples:
  - **assemble a peg into a hole**: approach, find hole, align, insert
  - **opening a door**: locating the handle, reaching out to grasp it, grasping it, opening the door
- **Task**:

My definitions

- **Motion**:
- **Skill**:
- **Task**: **symbolic constraints** between sub-Tasks (= partial fulfilment of the whole Task), in which each transition between two such sub-Tasks (compatible with the constraints) is realised by one out of a set of appropriate Skills.

My definitions

- **Motion**:
- **Skill**:
- **Task**: **symbolic constraints** between sub-Tasks (= partial fulfilment of the whole Task), in which each transition between two such sub-Tasks (compatible with the constraints) is realised by one out of a set of appropriate Skills. (Not so extremely simple) example: **bring a bottle of beer from the fridge**

Intermediate reflections

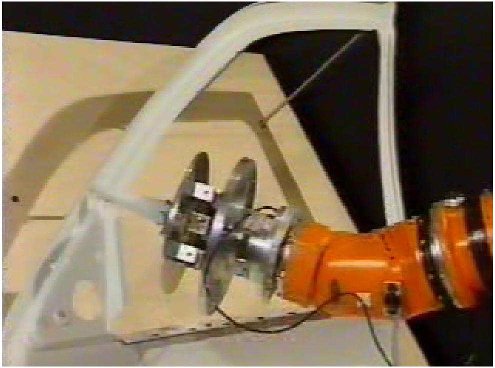
- Skills are the **“glue”** between the symbolic and the real worlds
- **reasoning** can take place at all levels
- **hierarchy** can exist at all levels.
- main & major & inevitable **research errors**: try to apply solutions fit for one level to problems at other levels.

Intermediate overview

Our research in Motion/Skill specification & execution:

- **Past**: (single) Task Frame Formalism
- **(Recent) Past**: (multiple) Feature Frame Formalism ("iTaSC")
- **Present**: Skills

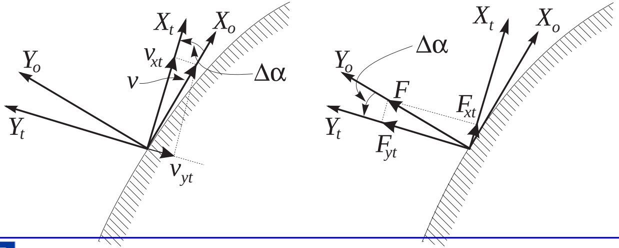
Past (1985–2000)
—Task Frame Formalism—



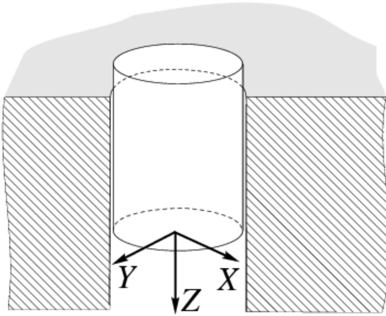
(Expected outcome of our PR2/WG contract!)

Task Frame Formalism (2)

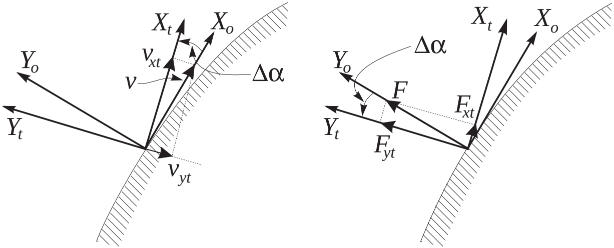
- Single frame with six DOFs.
- Explicit setpoints (= hard, uni-directional constraints)
- Velocity + Force.
- Only serial "skill" logic.
- Sensor-based tracking. (E.g., force, vision.)



```
move compliantly {
 with task frame directions
      xt:  force 0 N
      yt:  force 0 N
      zt:  velocity v mm/sec
      axt: force 0 Nmm
      ayt: force 0 Nmm
      axt velocity 0 rad/sec
 } until zt force < -f N
```



```
move compliantly {
  with task frame directions
       xt: velocity v mm/sec
       yt: force f N
       zt: velocity 0 mm/sec
       axt: velocity 0 rad/sec
       ayt: velocity 0 rad/sec
       azt: track (on velocities)
  } until until distance > d mm
```

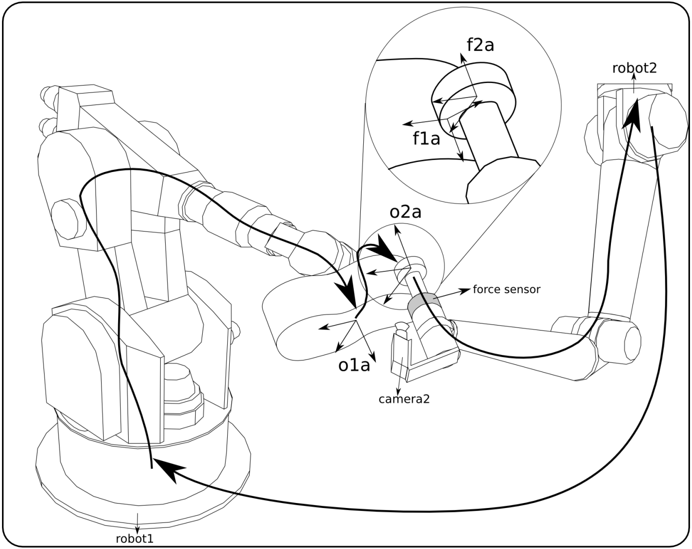


Past (2004–2009) —iTaSC—

“Instantaneous Task Specification with Constraints”

- **Multiple** frames. . .
- . . . with **partial** specification per frame. . .
- . . . and **constraint-based** i.s.o. setpoints.
- **Planning & Estimation** can be included.

Modelling primitives: kinematic loops



Step 1.: "Scene graph" model

- Relative geometric relationships between "feature frames" (⇒ ROS `tf2`?)
- Assign plan, estimation, uncertainty,... to each feature.

Step 1.: "Scene graph" model

- Relative geometric relationships between "feature frames" (⇒ ROS `tf2`?)
- Assign plan, estimation, uncertainty,. . . to each feature.

Step 2a.: Global objective function(s)

Step 2b.: Constraint specification per feature

Step 1.: “Scene graph” model

- Relative geometric relationships between “feature frames” (⇒ ROS `tf2`?)
- Assign plan, estimation, uncertainty,. . . to each feature.

Step 2a.: Global objective function(s)

Step 2b.: Constraint specification per feature

Step 3.: Solve the resulting constrained optimization problem

Step 1.: "Scene graph" model

- Relative geometric relationships between "feature frames" (⇒ ROS `tf2`?)
- Assign plan, estimation, uncertainty,... to each feature.

Step 2a.: Global objective function(s)

Step 2b.: Constraint specification per feature

Step 3.: Solve the resulting constrained optimization problem

Step 4.: Update the "scene graph" and iterate Step 3.

Present: Skills

- **Modelling**
- **Configuration**
- **Computation**
- **Coordination**

Present: Skills

- **Modelling** ("scene graph")
- **Configuration**
- **Computation**
- **Coordination**

Present: Skills

- **Modelling** ("scene graph")
- **Configuration** ("constraints"):
  - motion: instantaneous, trajectory primitives, interaction,...
    including weights between motion primitives and objective functions
  - learning: model parameter priors
- **Computation**
- **Coordination**

Present: Skills

- **Modelling** ("scene graph")
- **Configuration** ("constraints"):
  - motion: instantaneous, trajectory primitives, interaction,...
    including weights between motion primitives and objective functions
  - learning: model parameter priors
- **Computation** ("weighted constrained optimization")
  - instantaneous motion solver
  - estimation/learning/reasoning calculations

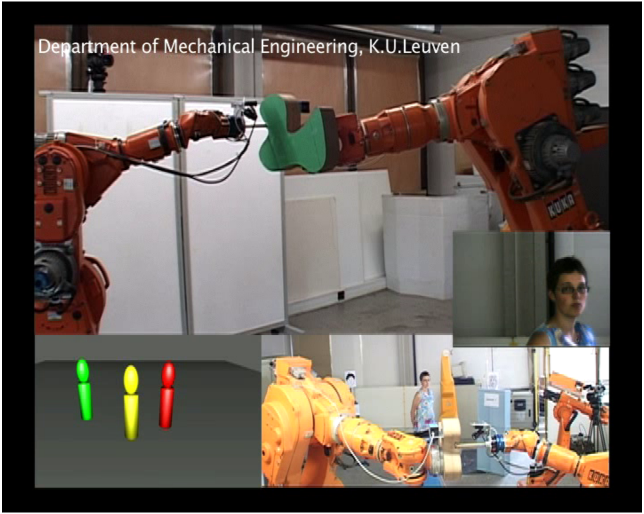
  includes monitoring of constraint **violations**!
- **Coordination**

Present: Skills

- **Modelling** ("scene graph")
- **Configuration** ("constraints"):
  - motion: instantaneous, trajectory primitives, interaction,...
    including weights between motion primitives and objective functions
  - learning: model parameter priors
- **Computation** ("weighted constrained optimization")
  - instantaneous motion solver
  - estimation/learning/reasoning calculations

  includes monitoring of constraint **violations**!
- **Coordination** (constraint violation event-driven FSM)
  includes "discrete scheduling" of **Bayesian network**!

Example experiment
—Human-aware dual-arm Skill—



Summary of presented paradigm

- our paradigm: a **methodological** way of **specifying** Skills and Motions
- Methodology = **4C** + constrained optimiz.
- **constraint-based**:
  - (**soft**) constraints are composable & bi-directional
  - constraints = **knowledge relationships**
  - allow **single-concept** **integration** of cognition and reasoning at all three **levels of abstraction** (Task, Skill, Motion)
- **multi-frame**, **partial** specification
- **scene graph** is central **shared** **resource**
- **traditional** *Sense-Plan-Act* is **smooth** limit case

Summary of presented paradigm (2)

- our competitive research advantage: an excellent partner for **symbol grounding** in cognitive robotics…
- …including professional/**commercially supported** implementation inside Orocos/ROS ecosystem

Summary of presented paradigm (2)

- our competitive research advantage: an excellent partner for **symbol grounding** in cognitive robotics...
- ...including professional/**commercially supported** implementation inside Orocos/ROS ecosystem
- *"To ROS or not to ROS?"*

Summary of presented paradigm (2)

- our competitive research advantage: an excellent partner for **symbol grounding** in cognitive robotics. . .
- . . . including professional/**commercially supported** implementation inside Orocos/ROS ecosystem
- *"To ROS or not to ROS?"*
  Wrong question!

⇒ *"Let's professionalize open source robotics!"*

Summary of presented paradigm (3)

- currently *best practice* and most impressive[a] implementations of our paradigm: **DLR Justin**…
- …using Simulink/RTW, and no open source…

[a] *Coffee making* video does *not* need "$\times$10" annotation…