Task Constrained Motion Planning for Robot Manipulators

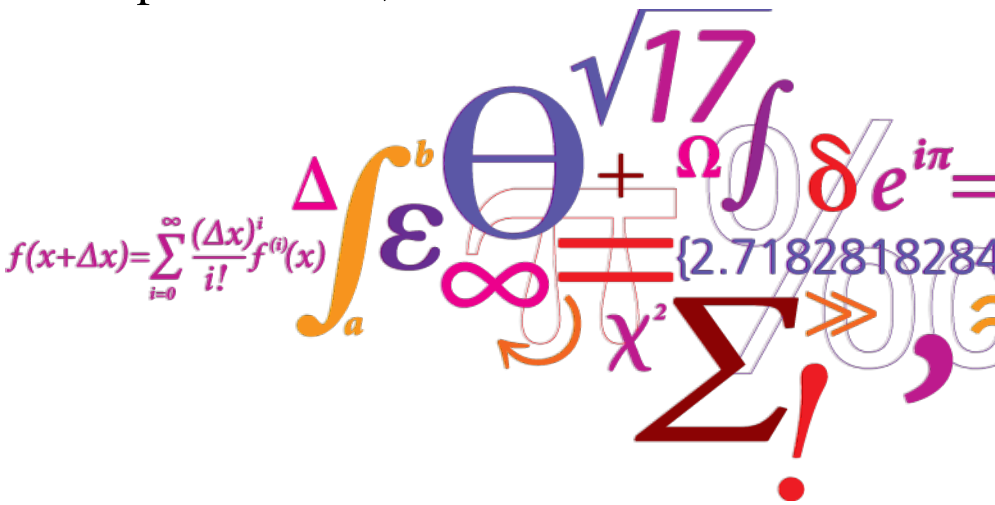
Marie Claire Capolei, Haiyan Wu, Adrian Llopart Maurin,

Nils Axel Andersen, Ole Ravn

Automation and Control

Technical University of Denmark

s151188@student.dtu.dk



Automation and Control

Department of Electrical Engineering

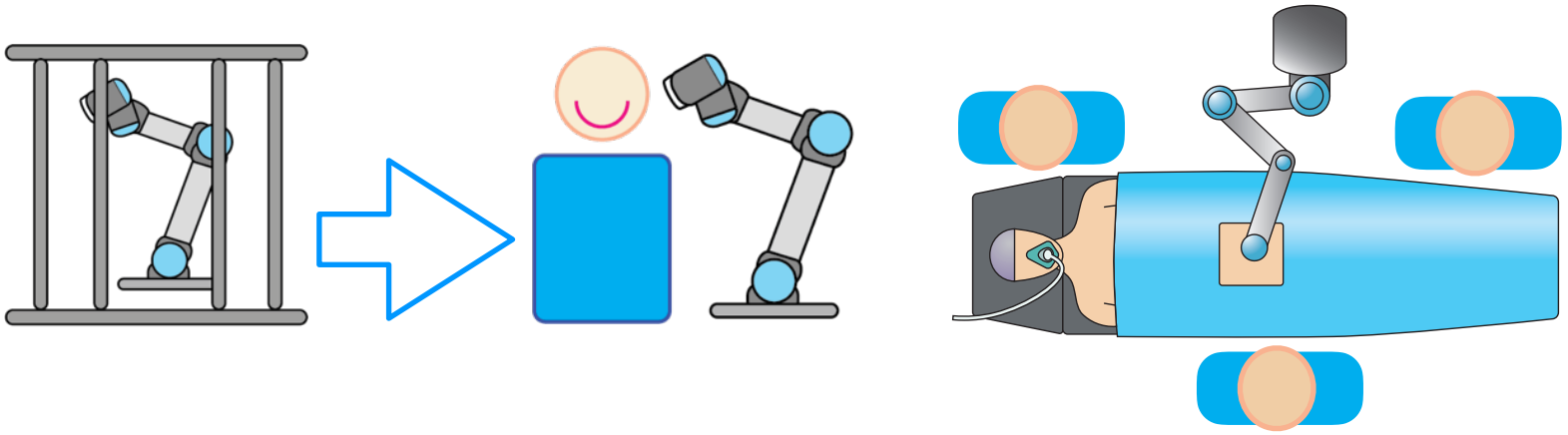
Outline

- Background and challenges
- Control architecture
- Method
- Experiment setup and results
- Conclusion

(work in progress)

General objective

- Human-aware motion system capable of elaborating safe robot motion while executing constrained task



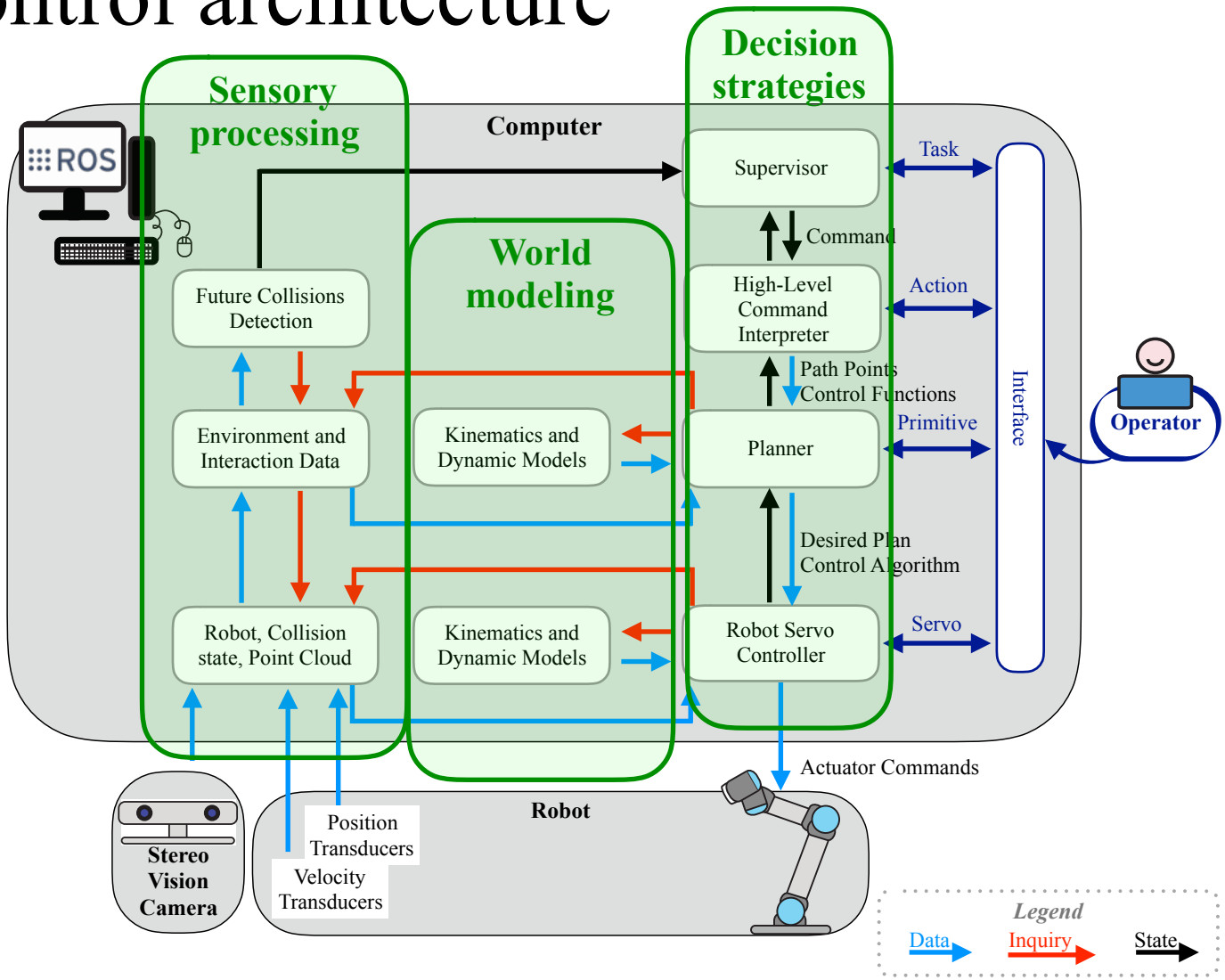
Constrained Motion Problem

- Connecting the start to the goal configuration
- Finding a collision-free path for a robot in an environment containing other rigid bodies
- Respecting other constraints, for example task constraints
- Possibly optimizing certain criteria

Challenges

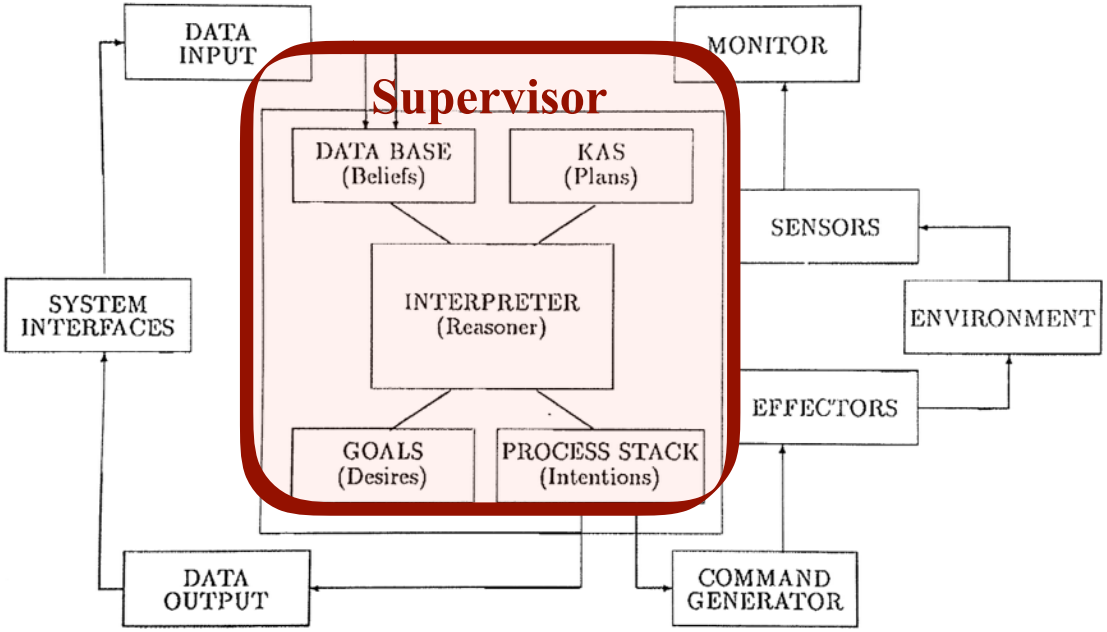
- Autonomous industrial robot arm
- Respecting predetermined task constraint
- Safe Human-Robot **collaboration**
- Flexible and modular software, suitable for future improvements, extension and integration on different industrial robot arm

Control architecture



The reasoning system

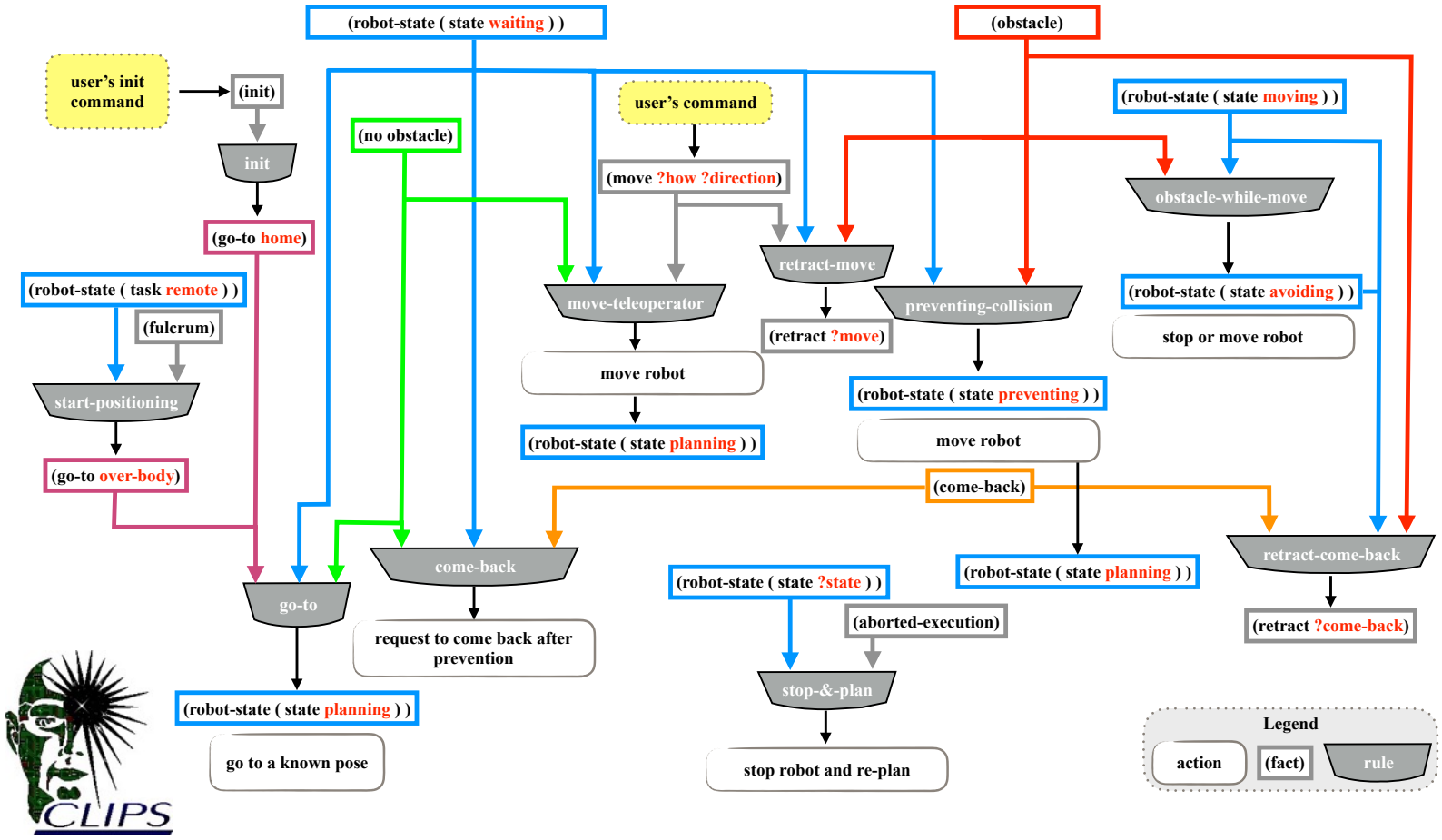
- System capable of generating appropriate actions that are function of some current set of **beliefs**, **desires** and **intention**



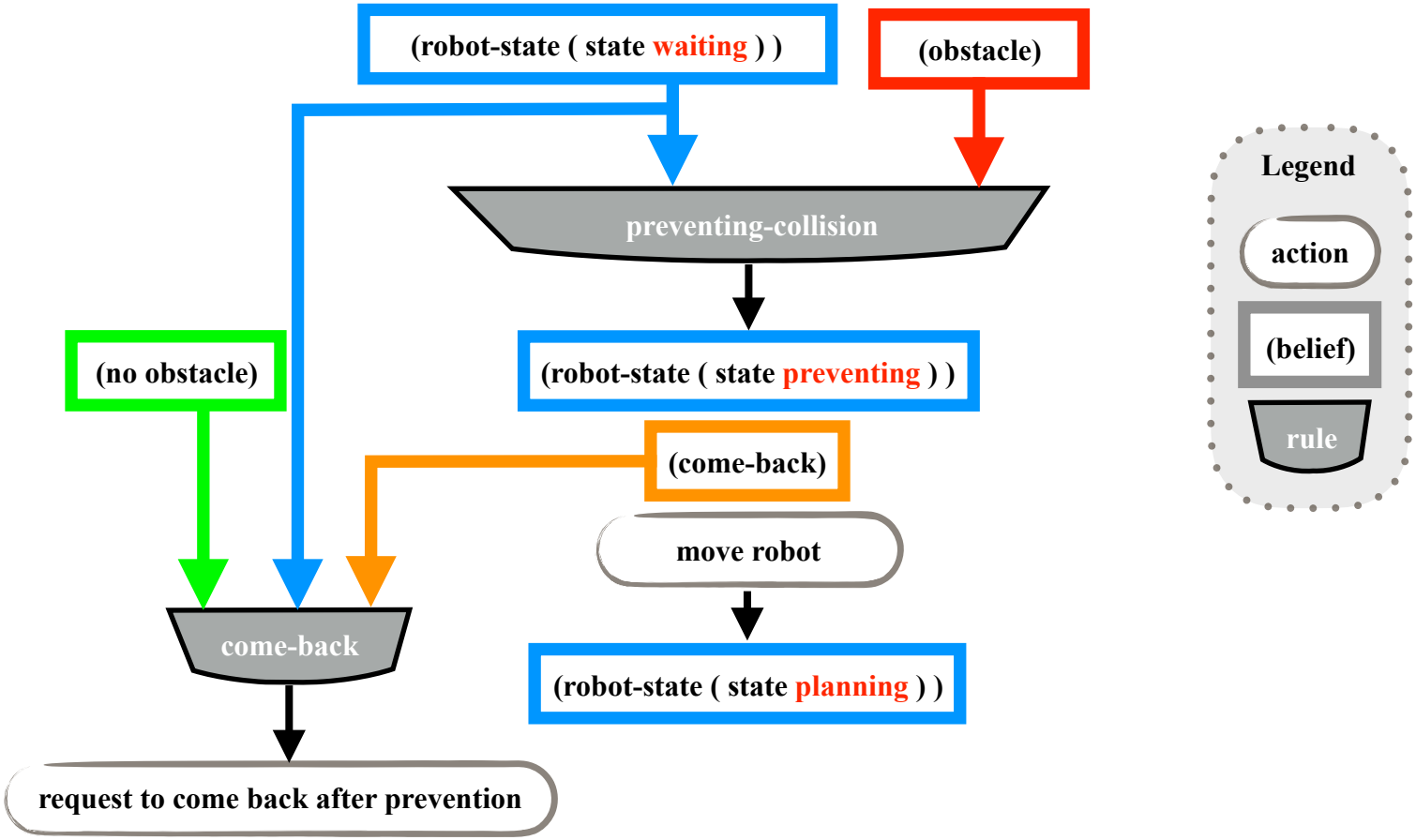
Procedural reasoning system (PRS)

Reasoning and planning in dynamic domain: an experiment with a mobile robot
Georgeff, M. P.; Lansky, A. L.; Schoppers, M. J.

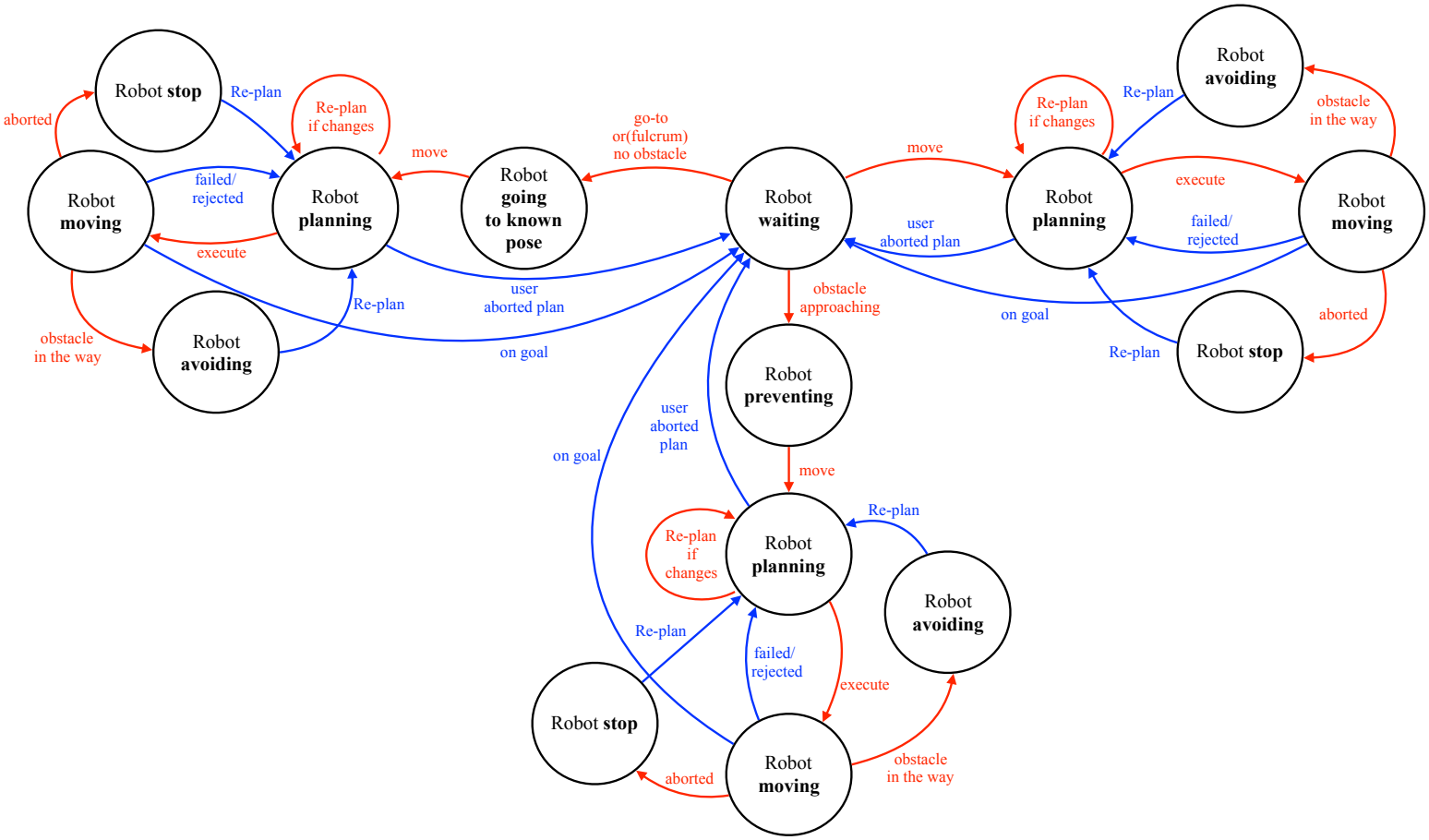
The interpreter logic



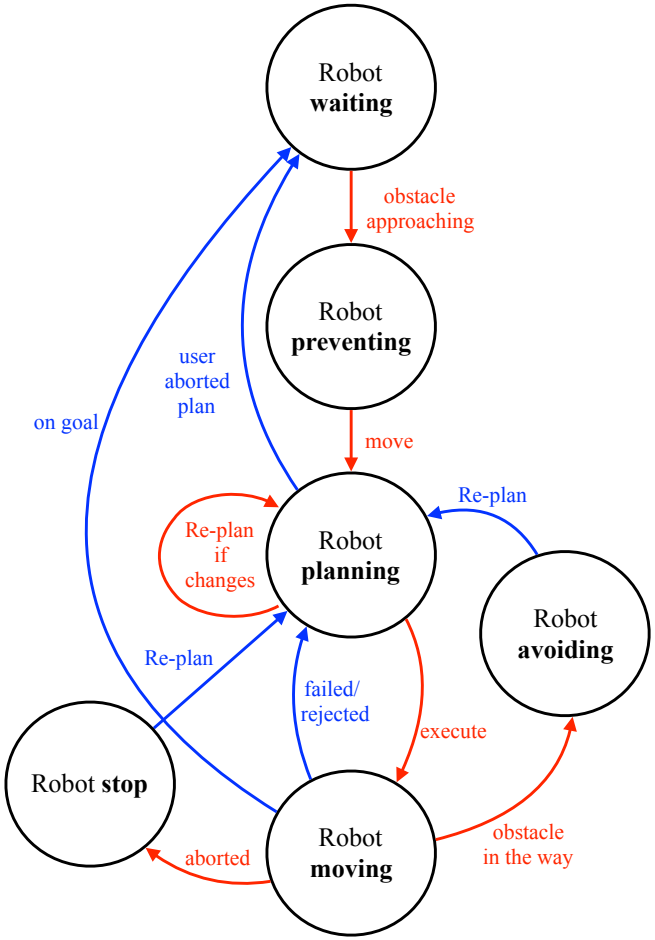
How the interpreter operates



State machine



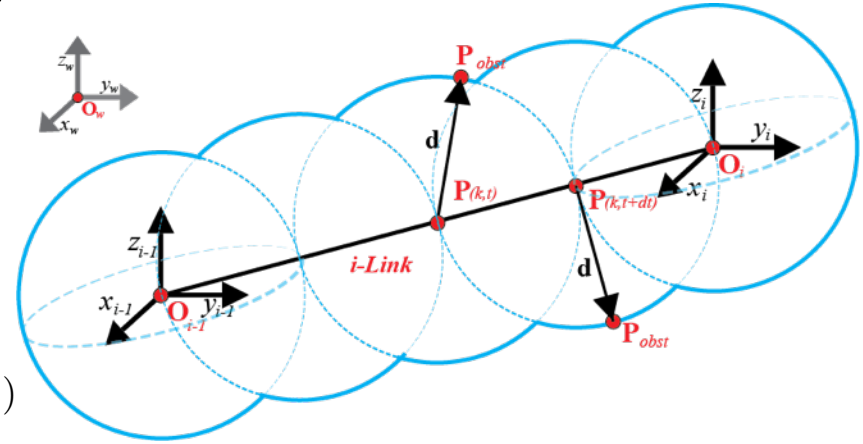
State machine: example



Obstacle detection algorithm

- The i-link is described by two frames:

$o_{i+1}x_{i-1}y_{i-1}z_{i-1}$, $o_i x_i y_i z_i$



- The axis of the link

$$r_{link_i}: \begin{cases} x_{link_i}(k,t) = x_{i-1}(k) + t \cdot (x_i(k) - x_{i-1}(k)) \\ y_{link_i}(k,t) = y_{i-1}(k) + t \cdot (y_i(k) - y_{i-1}(k)) \\ z_{link_i}(k,t) = z_{i-1}(k) + t \cdot (z_i(k) - z_{i-1}(k)) \end{cases}$$

- A rigid body is classified as *obstacle* when it is detected within a certain distance from the *i-link*

for $(t = 0, t <= 1)$ **do**

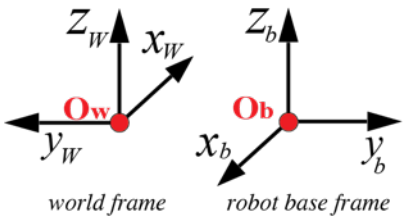
$\mathbf{P}_{link_i}(k,t) = (x_{link_i}(k,t),\ y_{link_i}(k,t),\ z_{link_i}(k,t))$

if $(distance(\mathbf{P}_{octomap},\ \mathbf{P}_{link_i}(k,t)) \leq safe\_distance)$ **then**

$prevent/avoid\_collision()$

$t \mathrel{+}= \Delta_t$

Positioning: algorithm



- Current end-effector pose respect to the robot base frame

$$\mathbf{T}^b_{ee,i} = \begin{bmatrix} \mathbf{R}^b_{ee,i} & \mathbf{o}^b_{ee,i} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} = \mathbf{O}^b_{ee,i}(\mathbf{o}^b_{ee,1}, \mathbf{R}^b_{ee,i})$$

- Current end-effector pose respect to the world frame

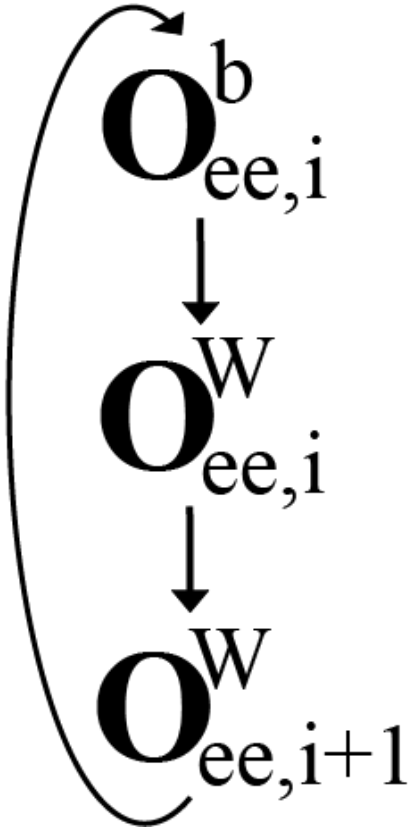
$$\mathbf{T}^W_{ee,i} = \mathbf{A}^W_b \cdot \mathbf{T}^b_{ee,i} = \mathbf{O}^W_{ee,i}(\mathbf{o}^W_{ee,1}, \mathbf{R}^W_{ee,i})$$

- End-effector **translation** respect to the world frame

$$\mathbf{O}^W_{ee,i+1} : \begin{cases} \mathbf{o}^W_{ee,i+1} = \mathbf{o}^W_{ee,i} + \delta \\ \mathbf{R}^W_{ee,i+1} = \mathbf{R}^W_{ee,i} \end{cases}$$

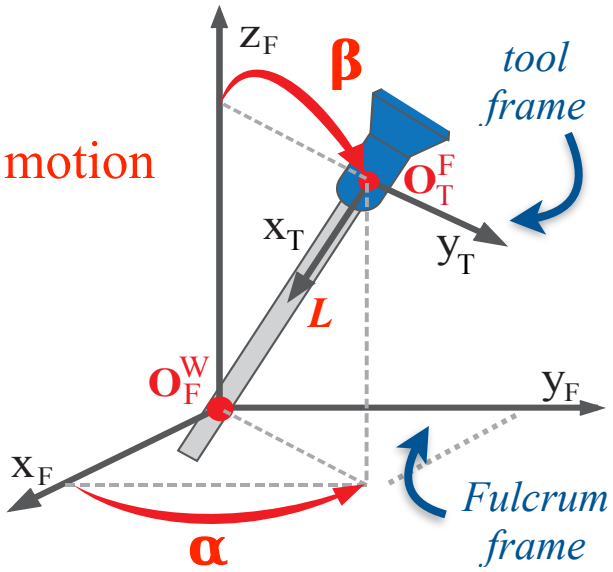
- End-effector **rotation** respect to the world frame

$$\mathbf{O}^W_{ee,i+1} : \begin{cases} \mathbf{R}_x(\psi) \cdot \mathbf{T}^W_{ee,i} & \text{rotate cw/ccw} \\ \mathbf{R}_y(\theta) \cdot \mathbf{T}^W_{ee,i} & \text{rotate forward/backward} \\ \mathbf{R}_z(\varphi) \cdot \mathbf{T}^W_{ee,i} & \text{rotate left/right} \end{cases}$$



Positioning: algorithm

Positioning the tool around a remote center of motion



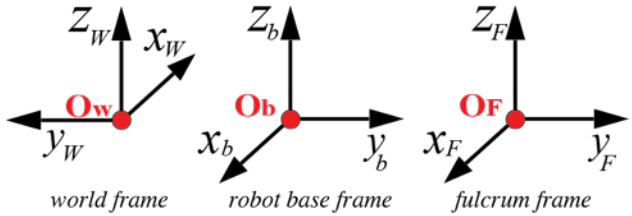
- Current tool pose in *world frame*

$$\mathbf{O}^{W}_{tool,i}(\mathbf{o}^{W}_{tool,i}, \mathbf{R}^{W}_{tool,i}) = \mathbf{T}^{W}_{tool,i} = \mathbf{T}^{W}_{ee,i} \cdot \mathbf{A}^{ee}_{tool}$$

- Current tool pose in *fulcrum frame*

$$\mathbf{O}^{F}_{tool,i}(\mathbf{o}^{F}_{tool,i}, \mathbf{R}^{F}_{tool,i}) = \mathbf{T}^{F}_{tool,i} = (\mathbf{T}^{W}_{F,i})^{-1} \cdot \mathbf{T}^{W}_{tool,i}$$

$$\mathbf{O}^{F}_{tool,i} : \begin{cases} \mathbf{o}^{F}_{tool,i}(\alpha_i, \beta_i, L_i) \\ \mathbf{R}^{F}_{tool,i} = \mathbf{R}_z(\alpha_i) \cdot \mathbf{R}_y(\beta_i) \end{cases}$$



- Next tool pose in fulcrum frame then in world frame

$$\mathbf{O}^{F}_{tool,i+1} : \begin{cases} \mathbf{o}^{F}_{tool,i} + \delta \\ \mathbf{R}_z(\alpha_{i+1}) \cdot \mathbf{R}_y(\beta_{i+1}) \end{cases} \Longrightarrow \mathbf{O}^{W}_{ee,i+1} = \mathbf{T}^{W}_{F,i} \cdot \mathbf{T}^{F}_{tool,i+1} \cdot (\mathbf{A}^{ee}_{tool})^{-1}$$

Positioning: algorithm

Positioning the end-effector in case of collision prevention

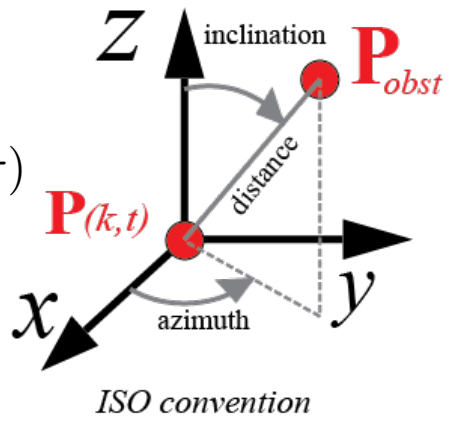
- The obstacle spheric coordinates respect to the link

$$inclination_{obst} = acos(\frac{z_{obst} - z_{link_i}(k,t)}{distance(\mathbf{P}_{obst},\ \mathbf{P}_{link_i}(k,t)})$$

$$azimut_{obst} = atan2(y_{obst} - y_{link_i}(k,t), x_{obst} - x_{link_i}(k,t))$$

- The desired end-effector position is function of the obstacle pose

$$\mathbf{o}^W_{ee,i+1}: \begin{cases} x^W_{ee,i+1} = x^W_{ee,i} + d \cdot cos(azimut_{obst} + \pi) \\ y^W_{ee,i+1} = y^W_{ee,i} + d \cdot sin(azimut_{obst} + \pi) \\ z^W_{ee,i+1} = z^W_{ee,i} + d \cdot cos(inclination_{obst} + \pi) \end{cases}$$

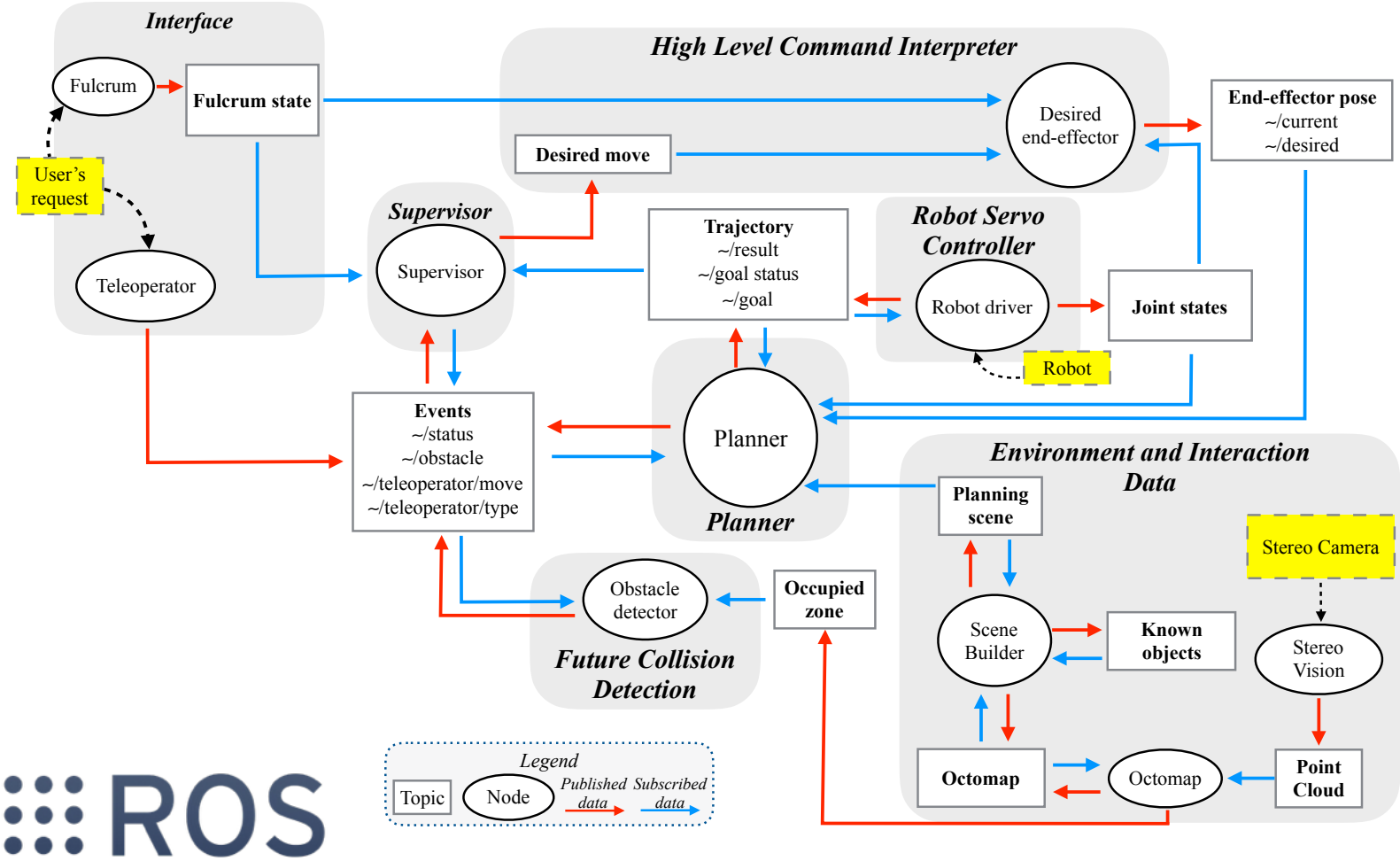


ISO convention

- The end-effector orientation is kept constant

$$\mathbf{R}^W_{ee,i+1} = \mathbf{R}^W_{ee,i}$$

Software components




Thanks!