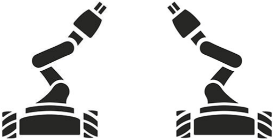
# Cooperative Object Transportation

## through an Unstructured Environment

Davide Torielli & Fabio Fusaro

September 10, 2018



# Outline

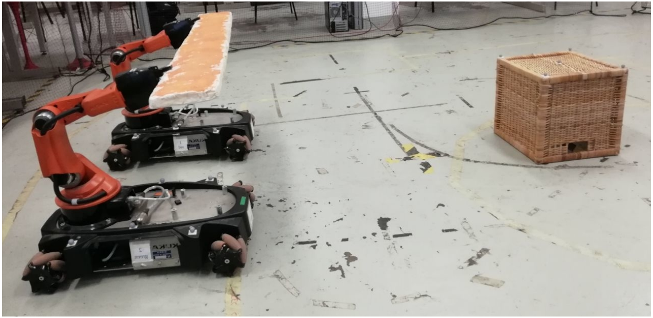
# Intoduction
## KUKA youBot


- **KUKA youBot**: a omnidirectional mobile platform with 4 wheels and a 5 DOF arm with a two-finger gripper.

# Introduction

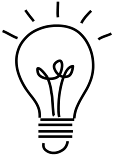
This work presents a solution to control mobile manipulators in difficult environments.

In particular, it involves cooperation between two KUKA youBots that must avoid an obstacle while carrying a common load.



# Outline

# The theory
## TPIK: the idea



Our work is based on the TPIK (Task Priority Inverse Kinematic) theory. It concerns different tasks with different priorities (e.g. considering joint limits, **avoiding obstacles**, reaching a desired configuration...) that the robot must satisfy.

The goal is to find a control which makes the robot achieve the wanted tasks.

# The theory
## TPIK: the math

In math terms, we have to find the system velocity reference vector $\dot{\boldsymbol{\bar{y}}}$ that satisfies **at best** the requirements (i.e. all the tasks).

$$S_k \triangleq \left\{ arg\ R\text{-} \min_{\dot{\boldsymbol{\bar{y}}} \in S_{k-1}} \left\| \boldsymbol{A}_k (\dot{\boldsymbol{\bar{x}}}_k - \boldsymbol{J}_k \dot{\boldsymbol{\bar{y}}}) \right\|^2 \right\}$$

- $\dot{\boldsymbol{\bar{x}}}_k$ is the stacked vector of all the desired output velocities.
- $\boldsymbol{J}_k$ is the Jacobian relationship expressing the current rate of change of the $k$-th task vector $[\dot{x}_{1,k}, ..., \dot{x}_{m,k}]^T$ with respect to the system velocity vector $\dot{\boldsymbol{y}}$.
- $\boldsymbol{A}_k$ is the diagonal matrix of all the activation functions of the $k$-th task.
- $S_{k-1}$ is to consider that the task must be satisfied only **after** satisfying all other **higher** priority tasks.

# The theory
## TPKI: cooperation

Things become a bit complicated because we must consider that we have two robots carrying a common load.

First, each robot acts as if it were alone. So we found the Cartesian **non-cooperative** tool-frame velocities for each agent:

$$\dot{\boldsymbol{x}}_{t,i} = \boldsymbol{J}_{t,i}\dot{\boldsymbol{y}}_i, \quad i = a, b$$

# The theory
## TPKI: cooperation

The idea is to give more way of action to the robot in more trouble finding a Cartesian **cooperative** tool-frame velocity that is a **weighted compromise** between the two Cartesian non-cooperative tool-frame velocities:

$$\dot{\hat{\mathbf{x}}}_t = \frac{1}{\mu_a + \mu_b}(\mu_a \dot{\mathbf{x}}_{t,a} + \mu_b \dot{\mathbf{x}}_{t,b}), \quad \mu_a, \mu_b > 0$$

In the general case, the result might not lay in the space of feasible object velocities; therefore it must be projected on this subspace to obtain a feasible solution.

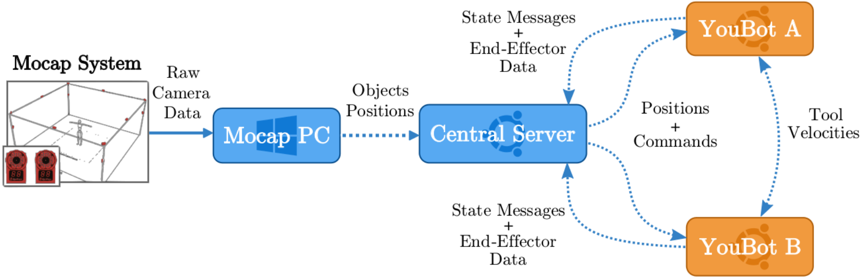
# Outline

1. Introduction
2. The theory
3. The practice
4. Our work
   - Obstacle avoidance with a single youBot
   - Obstacle avoidance with both youBots
5. Results

# The practice

## System architecture: MoCap & Central Server

- **MoCap System**: 8 OptiTrack Flex3 cameras which detect markers and a Motive software to send the robot positions.
- **Central Server**: a unified console to control the two youBots.



# Outline

# Outline

# Obstacle avoidance with a single youBot
## Reference Rate

We implement the task adding to the TPIK list the formula

$$\dot{\bar{x}}_{ja} = \gamma_{ja}((r + \Delta_{ja}) - \|\boldsymbol{d}\|)$$

which finds the reference rate which makes the joint go away from the obstacle.

With:

- $\gamma_{ja}$ the control gain.
- $r$ the radius of the obstacle.
- $\Delta_{ja}$ a constant that defines a safety distance from the obstacle.
- $\|\boldsymbol{d}\|$ the norm of the distance between the joint and the obstacle.

# Obstacle avoidance with a single youBot

## Task Induced Jacobian & Activation Function

Also the joint avoidance Jacobian must be added to find the suitable solution:

$$\boldsymbol{J}_{ja} = \boldsymbol{n}^T \boldsymbol{J}(\boldsymbol{c})$$

With:

- $\boldsymbol{n}^T = \frac{\boldsymbol{d}}{\|\boldsymbol{d}\|}$ the distance versor.
- $\boldsymbol{J}(\boldsymbol{c})$ the joint Jacobian, in function of the robot configuration vector $\boldsymbol{c}$.

And, as last, a suitable activation matrix $\boldsymbol{A}_{ja}$ is built to activate the task in the $\Delta_{ja}$ transition zone.

Initially this was implemented only for a single joint, then we added both formulas for each arm joint.

# Obstacle avoidance with a single youBot

## MoCap & UDP

The odometry is not accurate in general.
Also, we need a world reference frame for when the two youBot will cooperate.
So:

- We use the cameras to understand the position of the youBot through MoCap bridge.
- Through UDP infrastructure, the central Server controls the robot.

# Outline

# Obstacle avoidance with both youBots

For the **cooperation**, the most important thing is to not make the tool fall or break while performing the **obstacle avoidance** and the other tasks:

- The two robots agree on a common frame for the carried tool, i.e. maintaining a certain fixed distance from the frame object.
- They collaborate thanks to the data shared with the central Server through the UDP.

# Outline

# Results

The video is also visible HERE.

# Thank you for the attention!