Robotic Systems I

Lecture 9: Contacts and Optimization-based Control

Konstantinos Chatzilygeroudis - costashatz@upatras.gr

Department of Electrical and Computer Engineering
University of Patras

Template made by Panagiotis Papagiannopoulos

Manipulator Equation Recap v3

Manipulator Equation Reminder:

$$\underbrace{\boldsymbol{M}(\boldsymbol{q})}_{\text{"Mass Matrix"}}\dot{\boldsymbol{v}} + \underbrace{\boldsymbol{C}(\boldsymbol{q},\boldsymbol{v})}_{\text{"Coriolis/Gravity Forces"}} = \underbrace{\boldsymbol{u}}_{\text{"Usually } \boldsymbol{\tau}\text{"}} + \underbrace{\boldsymbol{F}_{\text{ext}}}_{\text{"External forces"}}$$

Velocity Kinematics:

$$\dot{\boldsymbol{q}} = \boldsymbol{G}(\boldsymbol{q})\boldsymbol{v}$$

Forward Dynamics:

$$\dot{\boldsymbol{v}} = \boldsymbol{M}^{-1}(\boldsymbol{q})\Big(\boldsymbol{u} + \boldsymbol{F}_{\text{ext}} - \boldsymbol{C}(\boldsymbol{q},\boldsymbol{v})\Big)$$

Inverse Dynamics:

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{q},\boldsymbol{v}) - \boldsymbol{F}_{\text{ext}}$$

Manipulator Equation (again!):

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}) = \boldsymbol{u} + \boldsymbol{F}_{\text{ext}}$$

Let's see some dimensions:

- $\boldsymbol{M} \in \mathbb{R}$

Manipulator Equation (again!):

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}) = \boldsymbol{u} + \boldsymbol{F}_{\text{ext}}$$

Let's see some dimensions:

- $\boldsymbol{M} \in \mathbb{R}^{n \times n}$
- $\boldsymbol{C} \in \mathbb{R}$

Manipulator Equation (again!):

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}) = \boldsymbol{u} + \boldsymbol{F}_{\text{ext}}$$

Let's see some dimensions:

- $\boldsymbol{M} \in \mathbb{R}^{n \times n}$
- $\boldsymbol{C} \in \mathbb{R}^{n \times 1}$
- $\boldsymbol{u} \in \mathbb{R}$

Manipulator Equation for Floating-Base Robots

Manipulator Equation (again!):

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}) = \boldsymbol{u} + \boldsymbol{F}_{\text{ext}}$$

Let's see some dimensions:

- $\boldsymbol{M} \in \mathbb{R}^{n \times n}$
- $\boldsymbol{C} \in \mathbb{R}^{n \times 1}$
- $\boldsymbol{u} \in \mathbb{R}^{n \times 1}$
- $\boldsymbol{F}_{\text{ext}} \in \mathbb{R}$

Manipulator Equation (again!):

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}) = \boldsymbol{u} + \boldsymbol{F}_{\text{ext}}$$

Let's see some dimensions:

- $\boldsymbol{M} \in \mathbb{R}^{n \times n}$
- $\boldsymbol{C} \in \mathbb{R}^{n \times 1}$
- $\boldsymbol{u} \in \mathbb{R}^{n \times 1}$
- $\boldsymbol{F}_{\text{ext}} \in \mathbb{R}^{n \times 1}$
- $n = n_{\text{base}} + n_{\text{joints}}$

Manipulator Equation for Floating-Base Robots

Manipulator Equation (again!):

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}) = \boldsymbol{u} + \boldsymbol{F}_{\text{ext}}$$

Let's see some dimensions:

- $\boldsymbol{M} \in \mathbb{R}^{n\times n}$
- $\boldsymbol{C} \in \mathbb{R}^{n\times 1}$
- $\boldsymbol{u} \in \mathbb{R}^{n\times 1}$
- $\boldsymbol{F}_{\text{ext}} \in \mathbb{R}^{n\times 1}$
- $n = n_{\text{base}} + n_{\text{joints}} = 6 + n_{\text{joints}}$

Manipulator Equation (again!):

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}) = \boldsymbol{u} + \boldsymbol{F}_{\text{ext}}$$

Let's see some dimensions:

- $\boldsymbol{M} \in \mathbb{R}^{n \times n}$
- $\boldsymbol{C} \in \mathbb{R}^{n \times 1}$
- $\boldsymbol{u} \in \mathbb{R}^{n \times 1}$
- $\boldsymbol{F}_{\text{ext}} \in \mathbb{R}^{n \times 1}$
- $n = n_{\text{base}} + n_{\text{joints}} = 6 + n_{\text{joints}}$

Manipulator Equation (floating-base):

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}) = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \boldsymbol{F}_{\text{ext}}$$

Manipulator Equation (floating-base):

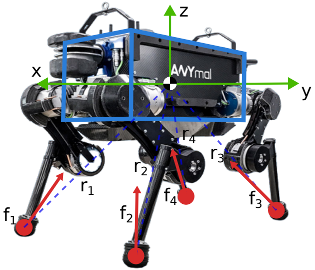
$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}) = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \boldsymbol{F}_{\text{ext}}$$

- **We have NO direct control over the base!**
- How can we control the robot?

Manipulator Equation (floating-base):

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}) = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \boldsymbol{F}_{\text{ext}}$$

- **We have NO direct control over the base!**
- How can we control the robot?
- Through contact forces: $\boldsymbol{F}_{\text{ext}}$!
- $\boldsymbol{F}_{\text{ext}} = \sum_{i=1}^{N_{\text{contact}}} \boldsymbol{J}_i(\boldsymbol{q})^T \boldsymbol{f}_i$
- $\boldsymbol{f}_i \in \mathbb{R}^3$



We assume that:

- $\dot{\boldsymbol{v}} \approx 0$
- $\boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}) \approx \boldsymbol{g}(\boldsymbol{q})$

The manipulator equation becomes:

$$\boldsymbol{g}(\boldsymbol{q}) = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \sum_{i=1}^{N_{\text{contact}}} \boldsymbol{J}_i(\boldsymbol{q})^T \boldsymbol{f}_i$$

We assume that:

- $\dot{\boldsymbol{v}} \approx 0$
- $\boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}) \approx \boldsymbol{g}(\boldsymbol{q})$

The manipulator equation becomes:

$$\boldsymbol{g}(\boldsymbol{q}) = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \sum_{i=1}^{N_{\text{contact}}} \boldsymbol{J}_i(\boldsymbol{q})^T \boldsymbol{f}_i$$

- **We have a robot with one leg!** How can we find the torques $\boldsymbol{\tau}$ to apply such that we *cancel the gravity*?
- $\begin{bmatrix} \boldsymbol{g}_u(\boldsymbol{q}) \\ \boldsymbol{g}_a(\boldsymbol{q}) \end{bmatrix} = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \begin{bmatrix} \boldsymbol{J}_u(\boldsymbol{q})^T \boldsymbol{f} \\ \boldsymbol{J}_a(\boldsymbol{q})^T \boldsymbol{f} \end{bmatrix}$

https://www.umass.edu/robotics/projects/staccatoe-single-legged-hopping-robot

https://www.umass.edu/robotics/projects/staccatoe-single-legged-hopping-robot

- **We have a robot with one leg!** How can we find the torques $\boldsymbol{\tau}$ to apply such that we *cancel the gravity*?
- $\begin{bmatrix} \boldsymbol{g}_u(\boldsymbol{q}) \\ \boldsymbol{g}_a(\boldsymbol{q}) \end{bmatrix} = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \begin{bmatrix} \boldsymbol{J}_u(\boldsymbol{q})^T\boldsymbol{f} \\ \boldsymbol{J}_a(\boldsymbol{q})^T\boldsymbol{f} \end{bmatrix}$

Monopod Balancing (2)

https://www.umass.edu/robotics/projects/staccatoe-single-legged-hopping-robot

- **We have a robot with one leg!** How can we find the torques $\boldsymbol{\tau}$ to apply such that we *cancel the gravity*?
- $\begin{bmatrix}\boldsymbol{g}_u(\boldsymbol{q})\\ \boldsymbol{g}_a(\boldsymbol{q})\end{bmatrix} = \begin{bmatrix}\boldsymbol{0}_6\\ \boldsymbol{\tau}\end{bmatrix} + \begin{bmatrix}\boldsymbol{J}_u(\boldsymbol{q})^T\boldsymbol{f}\\ \boldsymbol{J}_a(\boldsymbol{q})^T\boldsymbol{f}\end{bmatrix}$
- $\boldsymbol{g}_u(\boldsymbol{q}) = \boldsymbol{J}_u(\boldsymbol{q})^T\boldsymbol{f}$

Monopod Balancing (2)

https://www.umass.edu/robotics/projects/staccatoe-single-legged-hopping-robot

- **We have a robot with one leg!** How can we find the torques $\boldsymbol{\tau}$ to apply such that we *cancel the gravity*?
- $\begin{bmatrix} \boldsymbol{g}_u(\boldsymbol{q}) \\ \boldsymbol{g}_a(\boldsymbol{q}) \end{bmatrix} = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \begin{bmatrix} \boldsymbol{J}_u(\boldsymbol{q})^T \boldsymbol{f} \\ \boldsymbol{J}_a(\boldsymbol{q})^T \boldsymbol{f} \end{bmatrix}$
- $\boldsymbol{g}_u(\boldsymbol{q}) = \boldsymbol{J}_u(\boldsymbol{q})^T \boldsymbol{f}$
- $\boldsymbol{f} = \boldsymbol{J}_u(\boldsymbol{q})^{-T} \boldsymbol{g}_u(\boldsymbol{q})$

https://www.umass.edu/robotics/projects/staccatoe-single-legged-hopping-robot

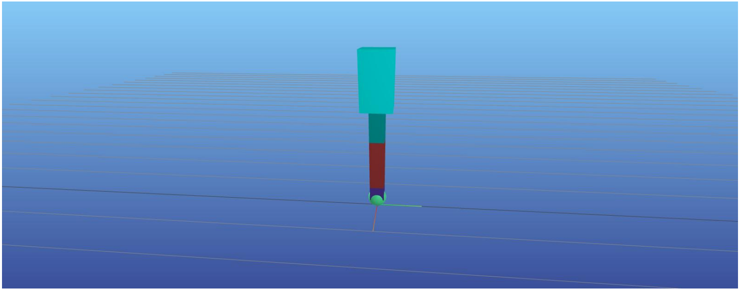
- **We have a robot with one leg!** How can we find the torques $\boldsymbol{\tau}$ to apply such that we *cancel the gravity*?
- $\begin{bmatrix} \boldsymbol{g}_u(\boldsymbol{q}) \\ \boldsymbol{g}_a(\boldsymbol{q}) \end{bmatrix} = \begin{bmatrix} \boldsymbol{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \begin{bmatrix} \boldsymbol{J}_u(\boldsymbol{q})^T \boldsymbol{f} \\ \boldsymbol{J}_a(\boldsymbol{q})^T \boldsymbol{f} \end{bmatrix}$
- $\boldsymbol{g}_u(\boldsymbol{q}) = \boldsymbol{J}_u(\boldsymbol{q})^T \boldsymbol{f}$
- $\boldsymbol{f} = \boldsymbol{J}_u(\boldsymbol{q})^{-T} \boldsymbol{g}_u(\boldsymbol{q})$
- Thus, $\boldsymbol{\tau} = \boldsymbol{g}_a(\boldsymbol{q}) - \boldsymbol{J}_a(\boldsymbol{q})^T \boldsymbol{f}$

Monopod Balancing (2)

https://www.umass.edu/robotics/projects/staccatoe-single-legged-hopping-robot

- **We have a robot with one leg!** How can we find the torques $\boldsymbol{\tau}$ to apply such that we *cancel the gravity*?
- $\begin{bmatrix} \boldsymbol{g}_u(\boldsymbol{q}) \\ \boldsymbol{g}_a(\boldsymbol{q}) \end{bmatrix} = \begin{bmatrix} \boldsymbol{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \begin{bmatrix} \boldsymbol{J}_u(\boldsymbol{q})^T \boldsymbol{f} \\ \boldsymbol{J}_a(\boldsymbol{q})^T \boldsymbol{f} \end{bmatrix}$
- $\boldsymbol{g}_u(\boldsymbol{q}) = \boldsymbol{J}_u(\boldsymbol{q})^T \boldsymbol{f}$
- $\boldsymbol{f} = \boldsymbol{J}_u(\boldsymbol{q})^{-T} \boldsymbol{g}_u(\boldsymbol{q})$
- Thus, $\boldsymbol{\tau} = \boldsymbol{g}_a(\boldsymbol{q}) - \boldsymbol{J}_a(\boldsymbol{q})^T \boldsymbol{f}$
- Is $\boldsymbol{J}_u(\boldsymbol{q})^T$ always invertible?

Monopod Balancing - Code Example



Biped Balancing

We assume that:

- $\dot{\boldsymbol{v}} \approx 0$
- $\boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}) \approx \boldsymbol{g}(\boldsymbol{q})$

Biped Manipulator Equation:

$$\boldsymbol{g}(\boldsymbol{q}) = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \boldsymbol{J}_l(\boldsymbol{q})^T \boldsymbol{f}_l + \boldsymbol{J}_r(\boldsymbol{q})^T \boldsymbol{f}_r$$

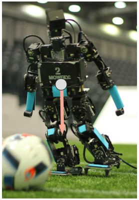
We assume that:

- $\dot{\boldsymbol{v}} \approx 0$
- $\boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}) \approx \boldsymbol{g}(\boldsymbol{q})$

Biped Manipulator Equation:

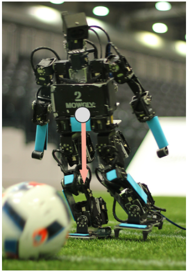
$$\boldsymbol{g}(\boldsymbol{q}) = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \boldsymbol{J}_l(\boldsymbol{q})^T \boldsymbol{f}_l + \boldsymbol{J}_r(\boldsymbol{q})^T \boldsymbol{f}_r$$

- How can we find the torques $\boldsymbol{\tau}$ to apply such that we *cancel the gravity*?
- $\begin{bmatrix} \boldsymbol{g}_u(\boldsymbol{q}) \\ \boldsymbol{g}_a(\boldsymbol{q}) \end{bmatrix} = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \begin{bmatrix} (\boldsymbol{J}_l(\boldsymbol{q})^T)_u \boldsymbol{f}_l + (\boldsymbol{J}_r(\boldsymbol{q})^T)_u \boldsymbol{f}_r \\ (\boldsymbol{J}_l(\boldsymbol{q})^T)_a \boldsymbol{f}_l + (\boldsymbol{J}_r(\boldsymbol{q})^T)_a \boldsymbol{f}_r \end{bmatrix}$



Source

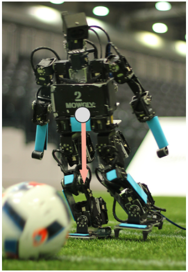
Biped Balancing (2)



Source

- $$\begin{bmatrix} \boldsymbol{g}_u(\boldsymbol{q}) \\ \boldsymbol{g}_a(\boldsymbol{q}) \end{bmatrix} = \begin{bmatrix} \boldsymbol{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \begin{bmatrix} (\boldsymbol{J}_l(\boldsymbol{q})^T)_u \boldsymbol{f}_l + (\boldsymbol{J}_r(\boldsymbol{q})^T)_u \boldsymbol{f}_r \\ (\boldsymbol{J}_l(\boldsymbol{q})^T)_a \boldsymbol{f}_l + (\boldsymbol{J}_r(\boldsymbol{q})^T)_a \boldsymbol{f}_r \end{bmatrix}$$

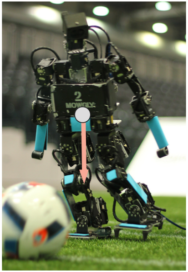
Biped Balancing (2)



Source

- $$\begin{bmatrix}\boldsymbol{g}_u(\boldsymbol{q})\\ \boldsymbol{g}_a(\boldsymbol{q})\end{bmatrix} = \begin{bmatrix}\boldsymbol{0}_6\\ \boldsymbol{\tau}\end{bmatrix} + \begin{bmatrix}(\boldsymbol{J}_l(\boldsymbol{q})^T)_u\boldsymbol{f}_l + (\boldsymbol{J}_r(\boldsymbol{q})^T)_u\boldsymbol{f}_r\\ (\boldsymbol{J}_l(\boldsymbol{q})^T)_a\boldsymbol{f}_l + (\boldsymbol{J}_r(\boldsymbol{q})^T)_a\boldsymbol{f}_r\end{bmatrix}$$
- $$\begin{bmatrix}\boldsymbol{f}_l\\ \boldsymbol{f}_r\end{bmatrix} = \begin{bmatrix}(\boldsymbol{J}_l(\boldsymbol{q})^T)_u & (\boldsymbol{J}_r(\boldsymbol{q})^T)_u\end{bmatrix}^\dagger \boldsymbol{g}_u(\boldsymbol{q})$$

Biped Balancing (2)



Source

- $\begin{bmatrix} \boldsymbol{g}_u(\boldsymbol{q}) \\ \boldsymbol{g}_a(\boldsymbol{q}) \end{bmatrix} = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \begin{bmatrix} (\boldsymbol{J}_l(\boldsymbol{q})^T)_u \boldsymbol{f}_l + (\boldsymbol{J}_r(\boldsymbol{q})^T)_u \boldsymbol{f}_r \\ (\boldsymbol{J}_l(\boldsymbol{q})^T)_a \boldsymbol{f}_l + (\boldsymbol{J}_r(\boldsymbol{q})^T)_a \boldsymbol{f}_r \end{bmatrix}$
- $\begin{bmatrix} \boldsymbol{f}_l \\ \boldsymbol{f}_r \end{bmatrix} = \begin{bmatrix} (\boldsymbol{J}_l(\boldsymbol{q})^T)_u & (\boldsymbol{J}_r(\boldsymbol{q})^T)_u \end{bmatrix}^\dagger \boldsymbol{g}_u(\boldsymbol{q})$
- This is minimizing $\boldsymbol{f}_{l/r}$! We need an optimization scheme to minimize $\boldsymbol{\tau}$!
- Is it balancing?

What if we have no assumptions?

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}) = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \boldsymbol{J}(\boldsymbol{q})^T \boldsymbol{f}$$

What if we have no assumptions?

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}) = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \boldsymbol{J}(\boldsymbol{q})^T \boldsymbol{f}$$

$$\begin{bmatrix} \boldsymbol{M}_u(\boldsymbol{q})\dot{\boldsymbol{v}} \\ \boldsymbol{M}_a(\boldsymbol{q})\dot{\boldsymbol{v}} \end{bmatrix} + \begin{bmatrix} \boldsymbol{C}_u(\boldsymbol{q}, \boldsymbol{v}) \\ \boldsymbol{C}_a(\boldsymbol{q}, \boldsymbol{v}) \end{bmatrix} = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \begin{bmatrix} \boldsymbol{J}_u(\boldsymbol{q})^T \boldsymbol{f} \\ \boldsymbol{J}_a(\boldsymbol{q})^T \boldsymbol{f} \end{bmatrix}$$

What if we have no assumptions?

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}) = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \boldsymbol{J}(\boldsymbol{q})^T \boldsymbol{f}$$

$$\begin{bmatrix} \boldsymbol{M}_u(\boldsymbol{q})\dot{\boldsymbol{v}} \\ \boldsymbol{M}_a(\boldsymbol{q})\dot{\boldsymbol{v}} \end{bmatrix} + \begin{bmatrix} \boldsymbol{C}_u(\boldsymbol{q}, \boldsymbol{v}) \\ \boldsymbol{C}_a(\boldsymbol{q}, \boldsymbol{v}) \end{bmatrix} = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \begin{bmatrix} \boldsymbol{J}_u(\boldsymbol{q})^T \boldsymbol{f} \\ \boldsymbol{J}_a(\boldsymbol{q})^T \boldsymbol{f} \end{bmatrix}$$

- Let's now solve for both $\dot{\boldsymbol{v}}$ and $\boldsymbol{f}$. What does this mean?

What if we have no assumptions?

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}) = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \boldsymbol{J}(\boldsymbol{q})^T \boldsymbol{f}$$

$$\begin{bmatrix} \boldsymbol{M}_u(\boldsymbol{q})\dot{\boldsymbol{v}} \\ \boldsymbol{M}_a(\boldsymbol{q})\dot{\boldsymbol{v}} \end{bmatrix} + \begin{bmatrix} \boldsymbol{C}_u(\boldsymbol{q}, \boldsymbol{v}) \\ \boldsymbol{C}_a(\boldsymbol{q}, \boldsymbol{v}) \end{bmatrix} = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \begin{bmatrix} \boldsymbol{J}_u(\boldsymbol{q})^T \boldsymbol{f} \\ \boldsymbol{J}_a(\boldsymbol{q})^T \boldsymbol{f} \end{bmatrix}$$

- Let's now solve for both $\dot{\boldsymbol{v}}$ and $\boldsymbol{f}$. What does this mean?
- $\begin{bmatrix} \boldsymbol{M}_u(\boldsymbol{q}) & -\boldsymbol{J}_u(\boldsymbol{q})^T \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{v}} \\ \boldsymbol{f} \end{bmatrix} = -\boldsymbol{C}_u(\boldsymbol{q}, \boldsymbol{v})$

Monopod Balancing v2

What if we have no assumptions?

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{q},\boldsymbol{v}) = \begin{bmatrix}\boldsymbol{0}_6\\ \boldsymbol{\tau}\end{bmatrix} + \boldsymbol{J}(\boldsymbol{q})^T\boldsymbol{f}$$

$$\begin{bmatrix}\boldsymbol{M}_u(\boldsymbol{q})\dot{\boldsymbol{v}}\\ \boldsymbol{M}_a(\boldsymbol{q})\dot{\boldsymbol{v}}\end{bmatrix} + \begin{bmatrix}\boldsymbol{C}_u(\boldsymbol{q},\boldsymbol{v})\\ \boldsymbol{C}_a(\boldsymbol{q},\boldsymbol{v})\end{bmatrix} = \begin{bmatrix}\boldsymbol{0}_6\\ \boldsymbol{\tau}\end{bmatrix} + \begin{bmatrix}\boldsymbol{J}_u(\boldsymbol{q})^T\boldsymbol{f}\\ \boldsymbol{J}_a(\boldsymbol{q})^T\boldsymbol{f}\end{bmatrix}$$

- Let's now solve for both $\dot{\boldsymbol{v}}$ and $\boldsymbol{f}$. What does this mean?
- $\begin{bmatrix}\boldsymbol{M}_u(\boldsymbol{q}) & -\boldsymbol{J}_u(\boldsymbol{q})^T\end{bmatrix}\begin{bmatrix}\dot{\boldsymbol{v}}\\ \boldsymbol{f}\end{bmatrix} = -\boldsymbol{C}_u(\boldsymbol{q},\boldsymbol{v})$
- $\begin{bmatrix}\dot{\boldsymbol{v}}\\ \boldsymbol{f}\end{bmatrix} = -\begin{bmatrix}\boldsymbol{M}_u(\boldsymbol{q}) & -\boldsymbol{J}_u(\boldsymbol{q})^T\end{bmatrix}^{\dagger}\boldsymbol{C}_u(\boldsymbol{q},\boldsymbol{v})$

Monopod Balancing v2

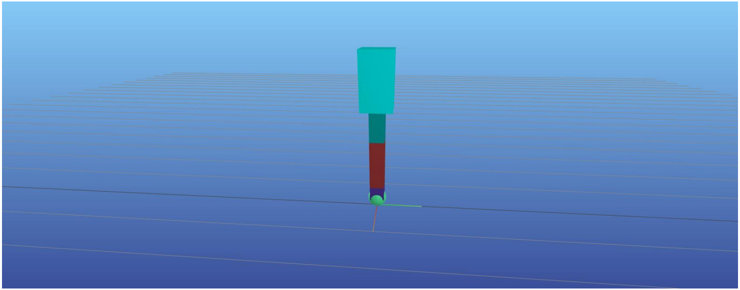
What if we have no assumptions?

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{q},\boldsymbol{v}) = \begin{bmatrix}\boldsymbol{0}_6\\ \boldsymbol{\tau}\end{bmatrix} + \boldsymbol{J}(\boldsymbol{q})^T\boldsymbol{f}$$

$$\begin{bmatrix}\boldsymbol{M}_u(\boldsymbol{q})\dot{\boldsymbol{v}}\\ \boldsymbol{M}_a(\boldsymbol{q})\dot{\boldsymbol{v}}\end{bmatrix} + \begin{bmatrix}\boldsymbol{C}_u(\boldsymbol{q},\boldsymbol{v})\\ \boldsymbol{C}_a(\boldsymbol{q},\boldsymbol{v})\end{bmatrix} = \begin{bmatrix}\boldsymbol{0}_6\\ \boldsymbol{\tau}\end{bmatrix} + \begin{bmatrix}\boldsymbol{J}_u(\boldsymbol{q})^T\boldsymbol{f}\\ \boldsymbol{J}_a(\boldsymbol{q})^T\boldsymbol{f}\end{bmatrix}$$

- Let's now solve for both $\dot{\boldsymbol{v}}$ and $\boldsymbol{f}$. What does this mean?
- $\begin{bmatrix}\boldsymbol{M}_u(\boldsymbol{q}) & -\boldsymbol{J}_u(\boldsymbol{q})^T\end{bmatrix}\begin{bmatrix}\dot{\boldsymbol{v}}\\ \boldsymbol{f}\end{bmatrix} = -\boldsymbol{C}_u(\boldsymbol{q},\boldsymbol{v})$
- $\begin{bmatrix}\dot{\boldsymbol{v}}\\ \boldsymbol{f}\end{bmatrix} = -\begin{bmatrix}\boldsymbol{M}_u(\boldsymbol{q}) & -\boldsymbol{J}_u(\boldsymbol{q})^T\end{bmatrix}^{\dagger}\boldsymbol{C}_u(\boldsymbol{q},\boldsymbol{v})$
- And $\boldsymbol{\tau} = \boldsymbol{M}_a(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}_a(\boldsymbol{q},\boldsymbol{v}) - \boldsymbol{J}_a(\boldsymbol{q})^T\boldsymbol{f}$
- Is it balancing?

Monopod Balancing v2 - Code Example



Manipulator Equation (again!):

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{q},\boldsymbol{v}) = \begin{bmatrix}\mathbf{0}_6\\ \boldsymbol{\tau}\end{bmatrix} + \sum_{i=1}^{N_{\text{contact}}} \boldsymbol{J}_i(\boldsymbol{q})^T\boldsymbol{f}_i$$

- Assuming we are at state $(\boldsymbol{q},\boldsymbol{v})$,

$$\text{the dynamics are linear with respect to } \begin{bmatrix}\dot{\boldsymbol{v}}\\ \boldsymbol{\tau}\\ \boldsymbol{f}_1\\ \vdots\\ \boldsymbol{f}_{N_{\text{contact}}}\end{bmatrix}$$

Manipulator Equation (again!):

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{q},\boldsymbol{v}) = \begin{bmatrix}\boldsymbol{0}_6\\ \boldsymbol{\tau}\end{bmatrix} + \sum_{i=1}^{N_{\text{contact}}} \boldsymbol{J}_i(\boldsymbol{q})^T\boldsymbol{f}_i$$

- Assuming we are at state $(\boldsymbol{q},\boldsymbol{v})$,

  the dynamics are linear with respect to $\begin{bmatrix}\dot{\boldsymbol{v}}\\ \boldsymbol{\tau}\\ \boldsymbol{f}_1\\ \vdots\\ \boldsymbol{f}_{N_{\text{contact}}}\end{bmatrix}$

- **Why?**

Multi-limb Robots

Manipulator Equation (again!):

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}) = \begin{bmatrix} \boldsymbol{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \sum_{i=1}^{N_{\text{contact}}} \boldsymbol{J}_i(\boldsymbol{q})^T \boldsymbol{f}_i$$

- Assuming we are at state $(\boldsymbol{q}, \boldsymbol{v})$,

  the dynamics are linear with respect to $\begin{bmatrix} \dot{\boldsymbol{v}} \\ \boldsymbol{\tau} \\ \boldsymbol{f}_1 \\ \vdots \\ \boldsymbol{f}_{N_{\text{contact}}} \end{bmatrix}$

- **Why?**

$$\begin{bmatrix} \boldsymbol{M}(\boldsymbol{q}) & -\boldsymbol{I}_{6+n_{\text{joints}}} & -\boldsymbol{J}_1(\boldsymbol{q})^T & \cdots & -\boldsymbol{J}_{N_{\text{contact}}}(\boldsymbol{q})^T \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{v}} \\ \boldsymbol{0}_6 \\ \boldsymbol{\tau} \\ \boldsymbol{f}_1 \\ \vdots \\ \boldsymbol{f}_{N_{\text{contact}}} \end{bmatrix} = -\boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v})$$

Multi-limb Robots (2)

$$\begin{bmatrix} \boldsymbol{M}(\boldsymbol{q}) \\ -\boldsymbol{S} \\ -\boldsymbol{J}_1(\boldsymbol{q})^T \\ \vdots \\ -\boldsymbol{J}_{N_{\text{contact}}}(\boldsymbol{q})^T \end{bmatrix}^T \begin{bmatrix} \dot{\boldsymbol{v}} \\ \boldsymbol{u} \\ \boldsymbol{f}_1 \\ \vdots \\ \boldsymbol{f}_{N_{\text{contact}}} \end{bmatrix} = -\boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}), \text{ where } \boldsymbol{S} = \begin{bmatrix} \mathbf{0}_{6\times 6} \\ \boldsymbol{I}_{n_{\text{joints}}} \end{bmatrix}$$

Multi-limb Robots (2)

$$\begin{bmatrix} \boldsymbol{M}(\boldsymbol{q}) \\ -\boldsymbol{S} \\ -\boldsymbol{J}_1(\boldsymbol{q})^T \\ \vdots \\ -\boldsymbol{J}_{N_{\text{contact}}}(\boldsymbol{q})^T \end{bmatrix}^T \begin{bmatrix} \dot{\boldsymbol{v}} \\ \boldsymbol{u} \\ \boldsymbol{f}_1 \\ \vdots \\ \boldsymbol{f}_{N_{\text{contact}}} \end{bmatrix} = -\boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v}), \text{ where } \boldsymbol{S} = \begin{bmatrix} \mathbf{0}_{6\times 6} \\ \boldsymbol{I}_{n_{\text{joints}}} \end{bmatrix}$$

- We set $\boldsymbol{x} = \begin{bmatrix} \dot{\boldsymbol{v}} \\ \boldsymbol{u} \\ \boldsymbol{f}_1 \\ \vdots \\ \boldsymbol{f}_{N_{\text{contact}}} \end{bmatrix}$ as the optimization variables and we can use **any optimizer that supports at least linear constraints!**
- We usually use a QP solver:
  - Flexible enough (quadratic objectives)
  - Supports linear constraints
  - Fast enough for real-time control!

Whole Body Control

- We call this type of controllers **"Whole-Body Controllers"**
- We also refer to this type of control as **"Whole-Body Control"** (WBC)
- We can add many tasks and let the optimizer find the best solution
- We can add many constraints (*linear*)
- QPs are fast! We can also do hierarchies of solvers!
- Generic way of defining controllers

Whole Body Control via QP

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{q}^T\boldsymbol{x}$$
$$\text{s.t. } \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} = \boldsymbol{0}$$
$$\boldsymbol{C}\boldsymbol{x} - \boldsymbol{d} \leqslant \boldsymbol{0}$$

where $\boldsymbol{x}, \boldsymbol{q} \in \mathbb{R}^N$, $\boldsymbol{Q} \succ 0 \in \mathbb{R}^{N\times N}$.

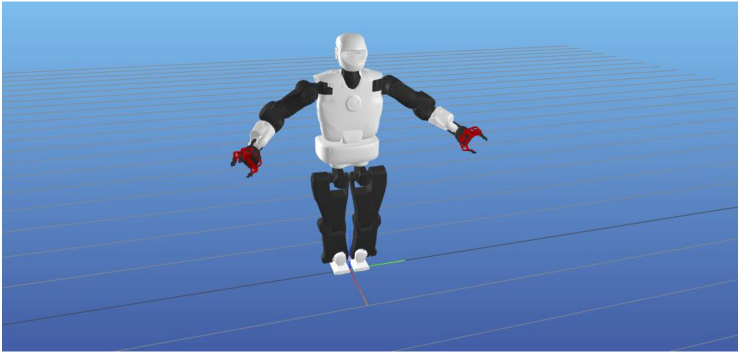
- $\boldsymbol{x} = \begin{bmatrix} \dot{\boldsymbol{v}} \\ \boldsymbol{u} \\ \boldsymbol{f}_1 \\ \vdots \\ \boldsymbol{f}_{N_{\text{contact}}} \end{bmatrix}$, $\boldsymbol{A} = \begin{bmatrix} \boldsymbol{M}(\boldsymbol{q}) & -\boldsymbol{S} & -\boldsymbol{J}_1(\boldsymbol{q})^T & \cdots & -\boldsymbol{J}_{N_{\text{contact}}}(\boldsymbol{q})^T \end{bmatrix}$, $\boldsymbol{b} = -\boldsymbol{C}(\boldsymbol{q}, \boldsymbol{v})$
- Multiple tasks are combined with a weighted sum: $\sum_{i=0}^{N_{\text{tasks}}} w_i \|\boldsymbol{W}_i\boldsymbol{x} - \boldsymbol{t}_i\|^2$
- Practically, we define $\boldsymbol{W} = \begin{bmatrix} w_1\boldsymbol{W}_1 \\ \vdots \\ w_{N_{\text{tasks}}}\boldsymbol{W}_{N_{\text{tasks}}} \end{bmatrix}$ and $\boldsymbol{t} = \begin{bmatrix} w_1\boldsymbol{t}_1 \\ \vdots \\ w_{N_{\text{tasks}}}\boldsymbol{t}_{N_{\text{tasks}}} \end{bmatrix}$
- We set $\boldsymbol{Q} = \boldsymbol{W}^T\boldsymbol{W}$ and $\boldsymbol{q} = -\boldsymbol{W}^T\boldsymbol{t}$
- We set $\boldsymbol{C}, \boldsymbol{d}$ according to any other constraints (like joint limits, etc.)

Whole Body Control via QP (2)

End-effector desired acceleration as a QP task:

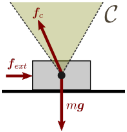
- We have a desired acceleration $\ddot{\boldsymbol{p}}_d$ for an end-effector.
- $\ddot{\boldsymbol{p}}_d$ usually comes from a P(ID) task-space controller
- How can we make this a QP task?
- $\dot{\boldsymbol{p}} = \boldsymbol{J}(\boldsymbol{q})\boldsymbol{v} \implies \ddot{\boldsymbol{p}} = \dot{\boldsymbol{J}}(\boldsymbol{q})\boldsymbol{v} + \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{v}}$ *(product rule)*
- Thus, we get a task like $\|\boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{v}} - (\ddot{\boldsymbol{p}}_d - \dot{\boldsymbol{J}}(\boldsymbol{q})\boldsymbol{v})\|^2$
- Putting this in all the optimization variables, we get $\boldsymbol{W}_i = \begin{bmatrix}\boldsymbol{J}(\boldsymbol{q}) & \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}$ and $\boldsymbol{t}_i = \ddot{\boldsymbol{p}}_d - \dot{\boldsymbol{J}}(\boldsymbol{q})\boldsymbol{v}$

Whole Body Control - Code Example



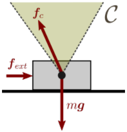
Contact Modelling

- We have been ignoring something!
  Contacts do not allow for any force!
- Let's consider the 2D contact



From `https://scaron.info/`

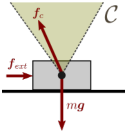
Contact Modelling



From `https://scaron.info/`

- We have been ignoring something!
  Contacts do not allow for any force!
- Let's consider the 2D contact
- This contact is fixed while the force $\boldsymbol{f}_c = m\boldsymbol{g} - \boldsymbol{f}_{\text{ext}}$ remains inside the *friction cone* $\mathcal{C}$
- We say the contact is in *fixed mode*

Contact Modelling



From https://scaron.info/

- We have been ignoring something!
  Contacts do not allow for any force!
- Let's consider the 2D contact
- This contact is fixed while the force $\boldsymbol{f}_c = m\boldsymbol{g} - \boldsymbol{f}_{\text{ext}}$ remains inside the *friction cone* $\mathcal{C}$
- We say the contact is in *fixed mode*
- When $\boldsymbol{f}_c \notin \mathcal{C}$, it switches to the *sliding mode*
- *Rolling mode* when the body rotates over the ground/other body
- *Broken/free mode* when there is no contact force $\boldsymbol{f}_c$

Controlling with Contacts

- We usually want the contacts to be in **fixed mode**!
- Sliding contacts are hard to control
- Rolling contacts are also most of the time undesirable
- Free mode is what we already know!

Friction Cones

- For each contact we have a force being applied at the contact position: $\boldsymbol{f}$
- We can split the force into the *normal* component, $\boldsymbol{f}_n$ and the *tangential* component, $\boldsymbol{f}_t$
- In 3D space we have 2 tangential components: $\boldsymbol{f}_t$ and $\boldsymbol{f}_b$
- For the contact to be fixed we need to have the following:

$$
\begin{aligned}
(\boldsymbol{f} \cdot \boldsymbol{n}) &> 0, \text{ the force cannot pull/grasp!} \\
\|\boldsymbol{f} \cdot \boldsymbol{t}\|_2 &\leqslant \mu(\boldsymbol{f} \cdot \boldsymbol{n}) \\
\|\boldsymbol{f} \cdot \boldsymbol{b}\|_2 &\leqslant \mu(\boldsymbol{f} \cdot \boldsymbol{n})
\end{aligned}
$$

where μ is the Coulomb *static friction coefficient* at the contact, $\boldsymbol{n}$ is the normal direction and $\boldsymbol{t}, \boldsymbol{b}$ are the tangential directions of the contact.

Friction Cones in QP

Friction Cone Constraints:

$$(\boldsymbol{f} \cdot \boldsymbol{n}) > 0, \text{ the force cannot pull/grasp!}$$
$$\|\boldsymbol{f} \cdot \boldsymbol{t}\|_2 \leqslant \mu(\boldsymbol{f} \cdot \boldsymbol{n})$$
$$\|\boldsymbol{f} \cdot \boldsymbol{b}\|_2 \leqslant \mu(\boldsymbol{f} \cdot \boldsymbol{n})$$

These are "Conic" constraints!

Friction Cones in QP

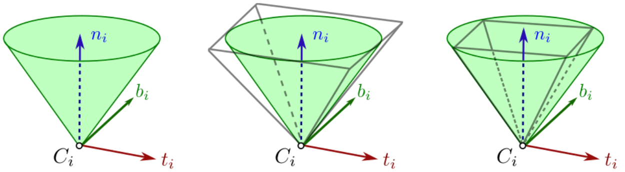
Friction Cone Constraints:

$$(\boldsymbol{f} \cdot \boldsymbol{n}) > 0, \text{ the force cannot pull/grasp!}$$

$$\|\boldsymbol{f} \cdot \boldsymbol{t}\|_2 \leqslant \mu(\boldsymbol{f} \cdot \boldsymbol{n})$$

$$\|\boldsymbol{f} \cdot \boldsymbol{b}\|_2 \leqslant \mu(\boldsymbol{f} \cdot \boldsymbol{n})$$

These are "Conic" constraints! We need to linearize them for the QP:



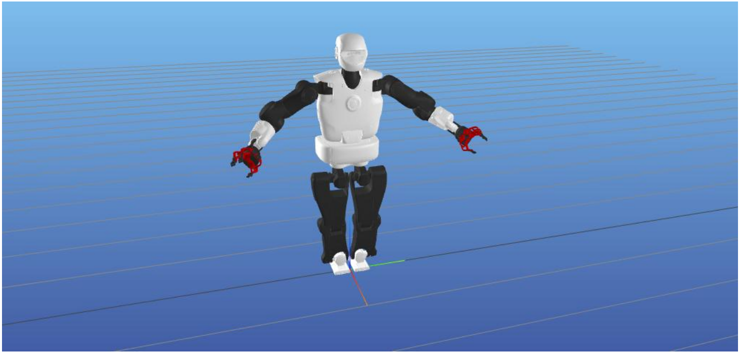
From `https://scaron.info/`

$$(\boldsymbol{f} \cdot \boldsymbol{n}) > 0, \text{ the force cannot pull/grasp!}$$

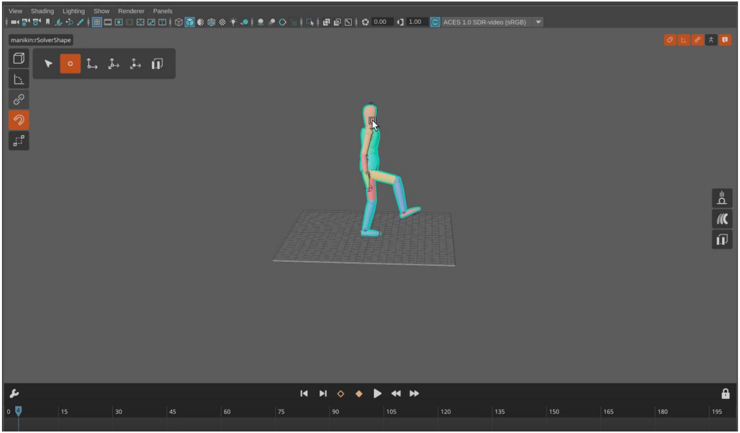
$$|\boldsymbol{f} \cdot \boldsymbol{t}| \leqslant \mu(\boldsymbol{f} \cdot \boldsymbol{n}) \text{ or } |\boldsymbol{f} \cdot \boldsymbol{t}| \leqslant \frac{\mu}{\sqrt{2}}(\boldsymbol{f} \cdot \boldsymbol{n})$$

$$|\boldsymbol{f} \cdot \boldsymbol{b}| \leqslant \mu(\boldsymbol{f} \cdot \boldsymbol{n}) \text{ or } |\boldsymbol{f} \cdot \boldsymbol{b}| \leqslant \frac{\mu}{\sqrt{2}}(\boldsymbol{f} \cdot \boldsymbol{n})$$

WBC with Friction Cones - Code Example



WBC in the wild!



Bibliography

Optimization Based Full Body Control for the Atlas Robot, *Feng, S., Whitman, E., Xinjilefu, X. and Atkeson, C.G.*, 2014, Humanoids. paper

Balancing experiments on a torque-controlled humanoid with hierarchical inverse dynamics, *Herzog, A., Righetti, L., Grimminger, F., Pastor, P. and Schaal, S.*, 2014, IROS. paper

Thank you

- **Any Questions?**
- **Office Hours**:
  - **Tue-Wed** *(10:00-12:00)*
  - 24/7 by email (`costashatz@upatras.gr`, subject: *ECE_RSI_AM*)
- **Material and Announcements**

Laboratory of Automation & Robotics