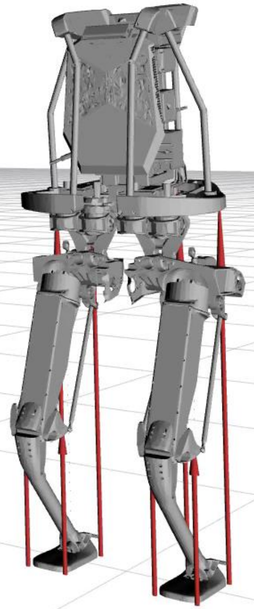
INNOVATIVE ROBOTICS FOR AGILE PRODUCTION

# Trajectory Optimization and Model Predictive Control for Agile Bipedal Locomotion



**Enrico Mingo Hoffman**

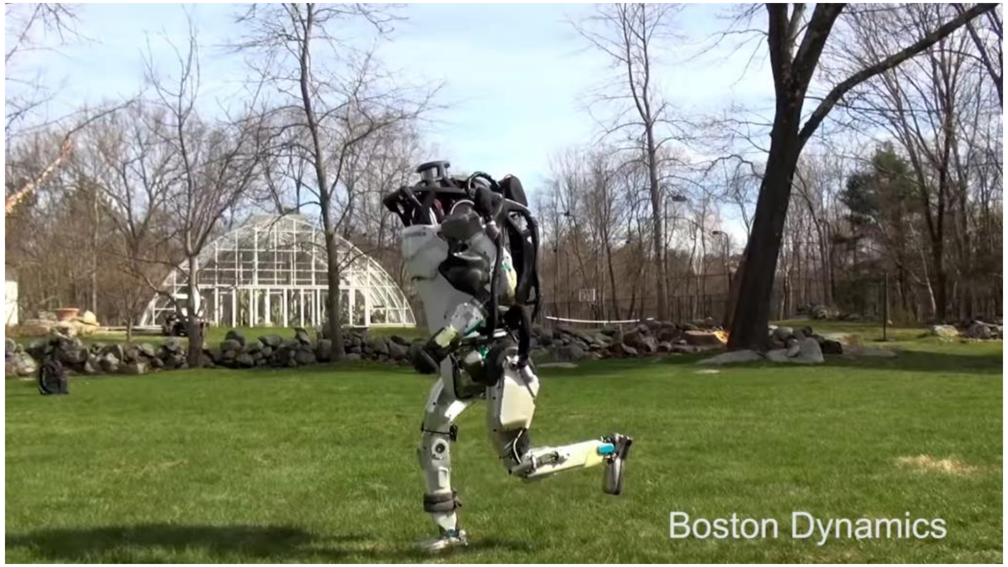
Humanoids 2022, Ginowan, Okinawa, Japan

Humanoids 2022

# Agile and Dynamic Locomotion

## Fundamental motor skills for deployment of humanoid bipeds in real applications

→ Efficiently and effectively traverse environments




**Challenges:**

→ Large contact forces (& momentum)

→ Multiple impacts

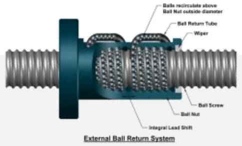
→ Aerial phases (limited control action)

→ Adaptability

→ Whole-Body motions

# PAL Robotics

## Development of technologies for agile and dynamic locomotion




### 01.

### Electric Linear Actuators

High efficiency and robustness, with high-power output, back-drivable.

### 02.

### Serial-Parallel Hybrid Chains

Resilience to impact and improved mass/inertia distribution.

### 03.

### Optimal Control

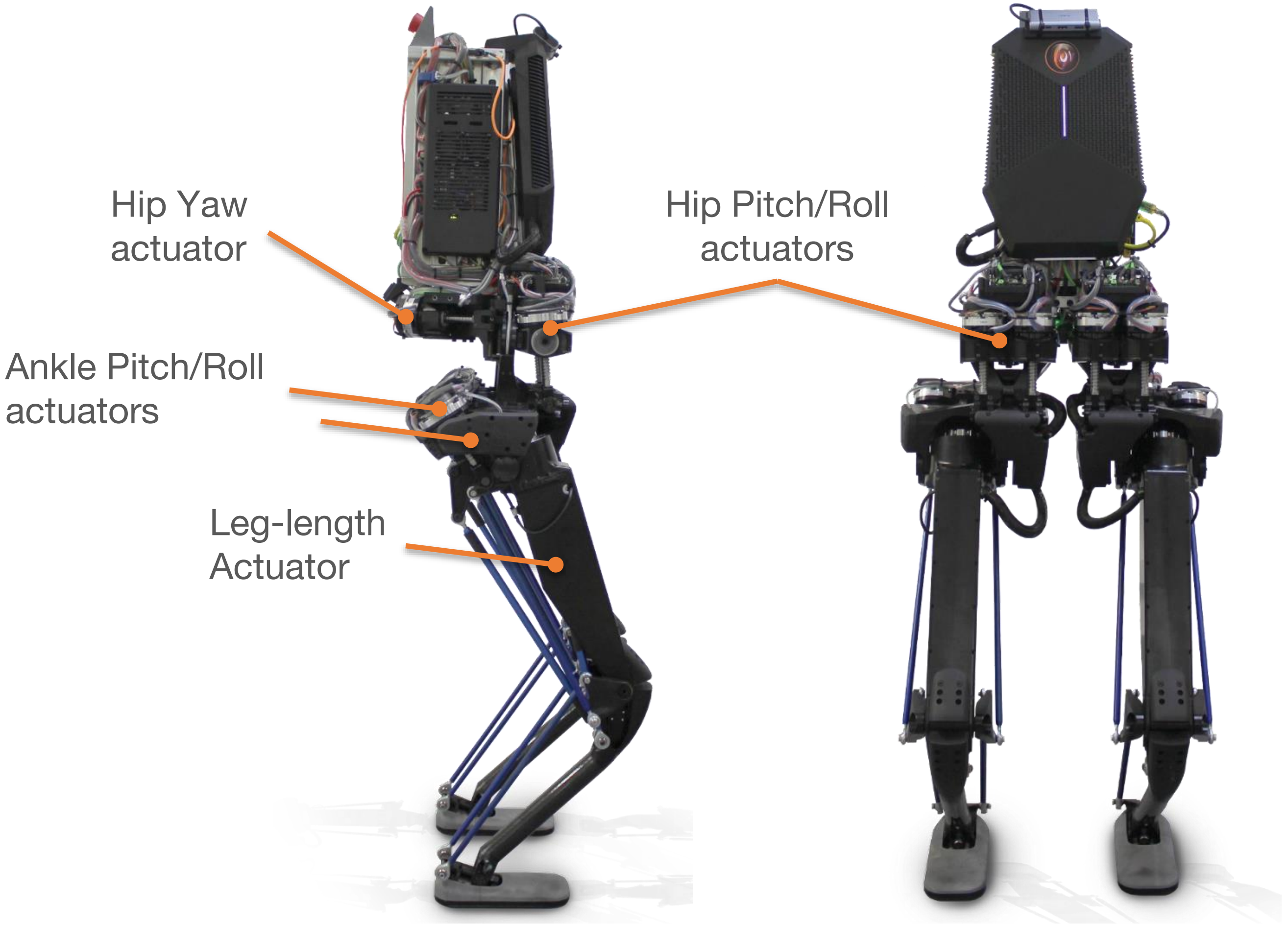
Planning and control of energy-efficient dynamic motions.

# Kangaroo

- 12 DoFs
- 2 types/sizes linear actuators
- 12 serial-parallel hybrid mechanisms
- Non-linear transmission
- Low inertia/mass legs
- High impact resilience

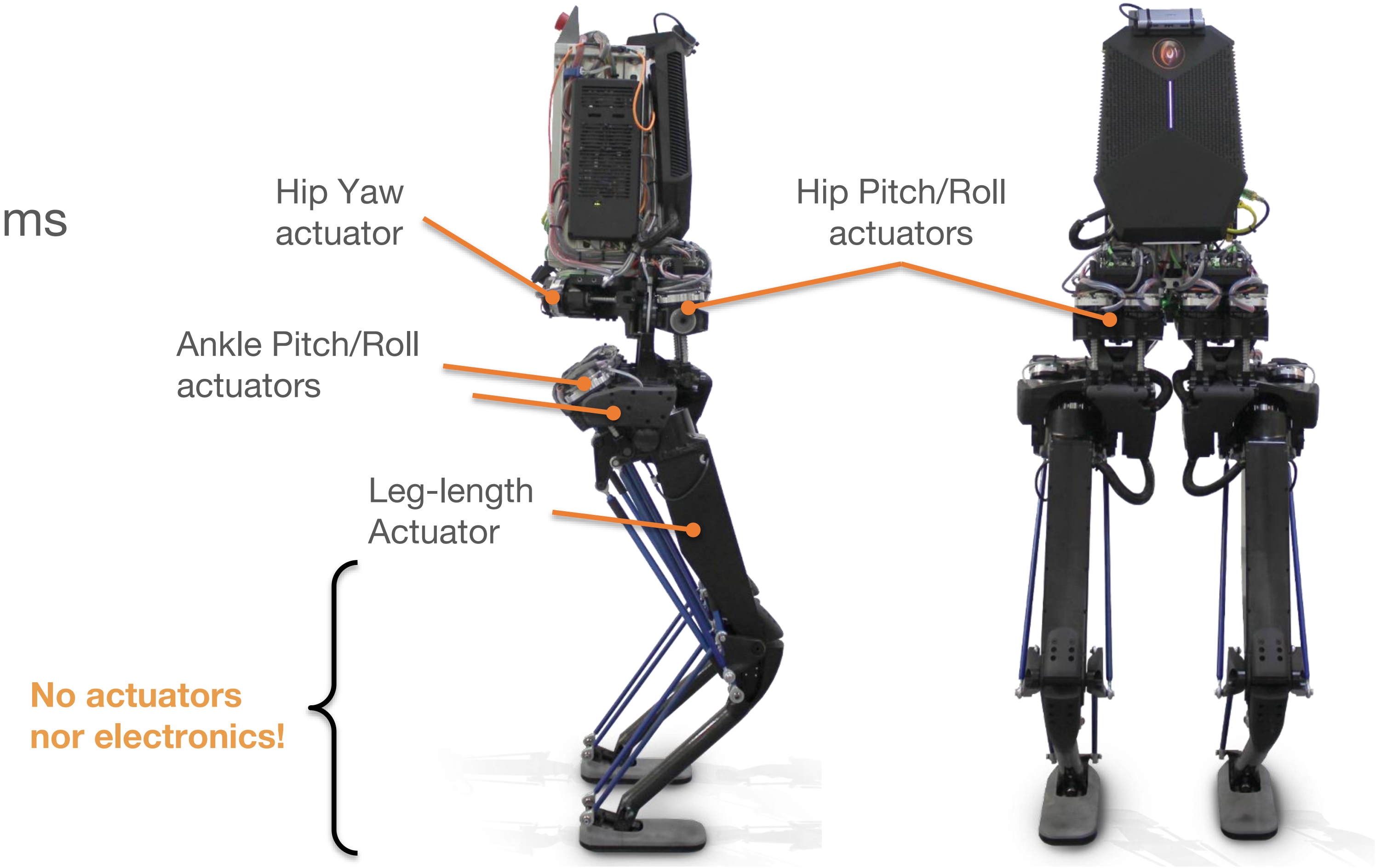
# Kangaroo

- 12 DoFs
- 2 types/sizes linear actuators
- 12 serial-parallel hybrid mechanisms
- Non-linear transmission
- Low inertia/mass legs
- High impact resilience



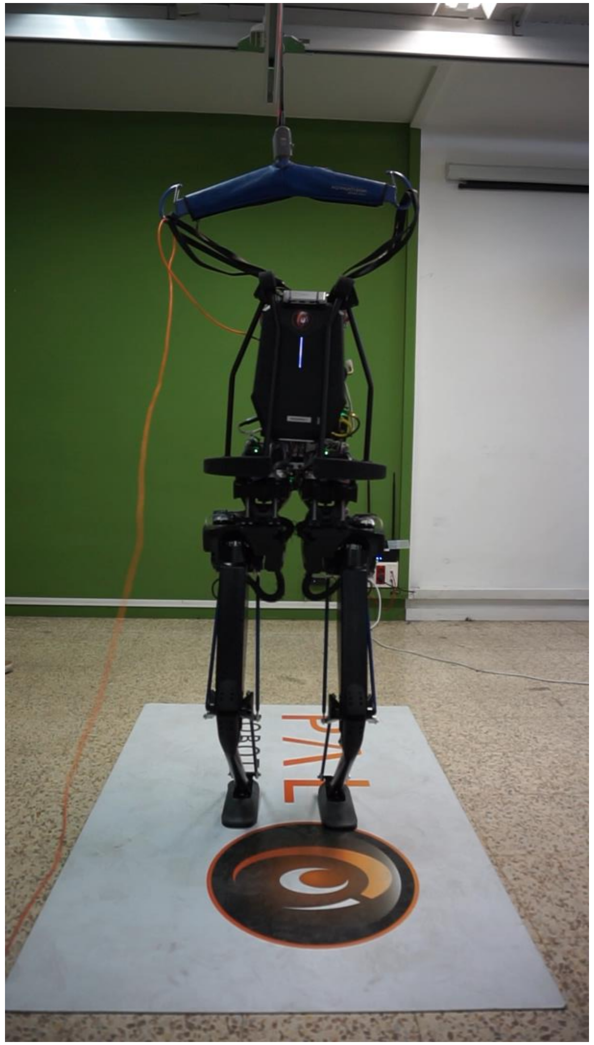
# Kangaroo

- 12 DoFs
- 2 types/sizes linear actuators
- 12 serial-parallel hybrid mechanisms
- Non-linear transmission
- Low inertia/mass legs
- High impact resilience



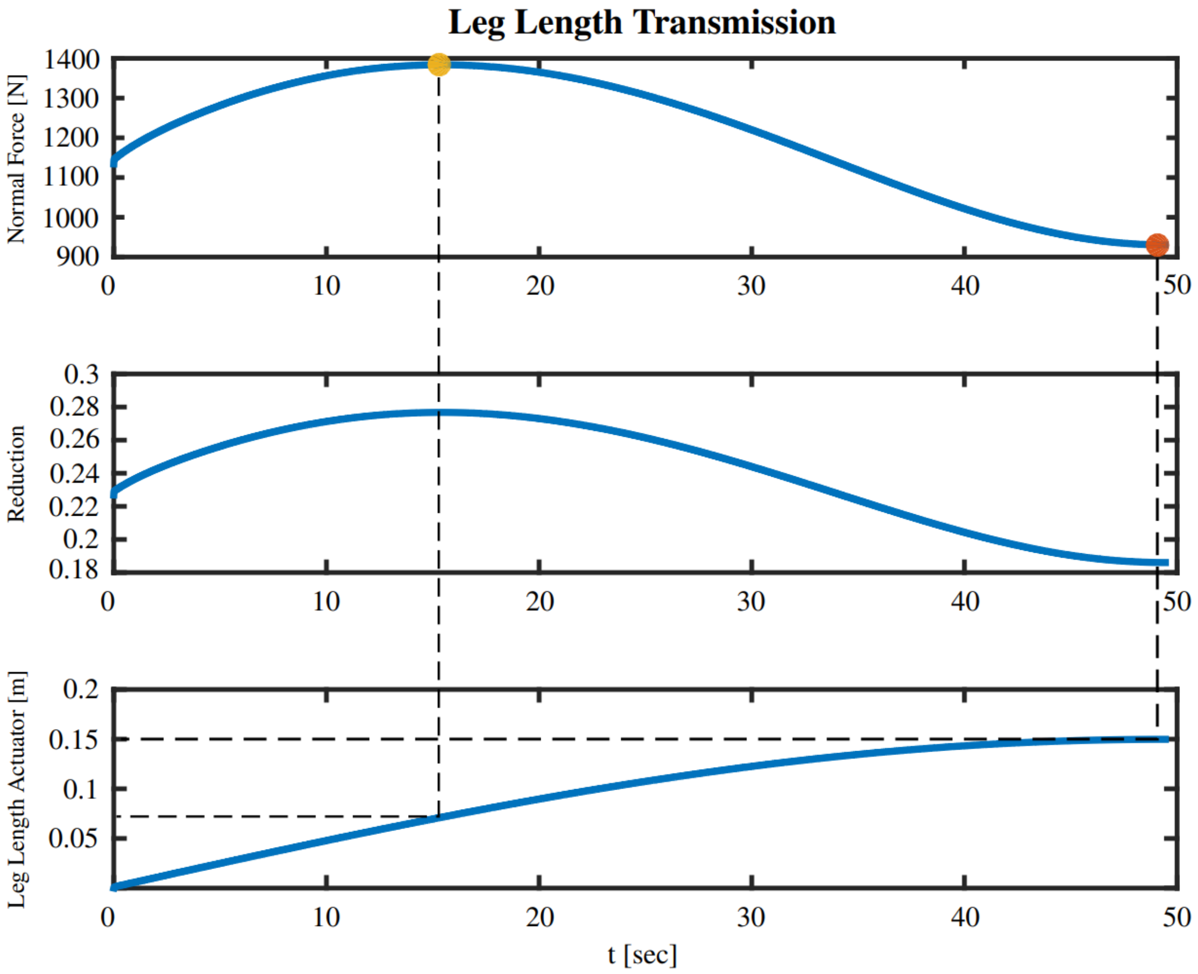
# Kangaroo

- 12 DoFs
- 2 types/sizes linear actuators
- 12 serial-parallel hybrid mechanisms
- Non-linear transmission
- Low inertia/mass legs
- High impact resilience



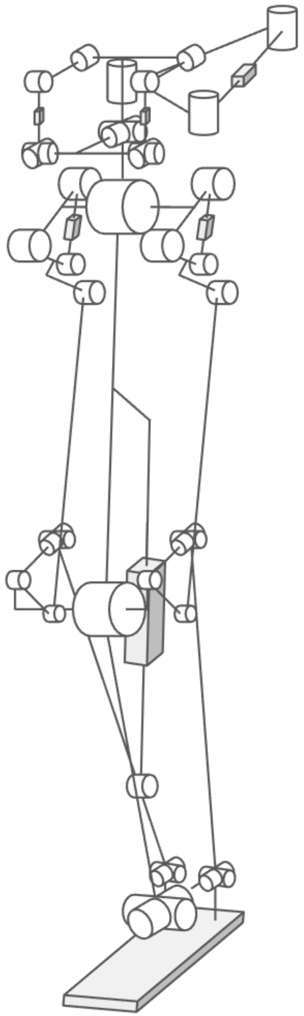
"handcrafted" jump

# Non-linear transmission

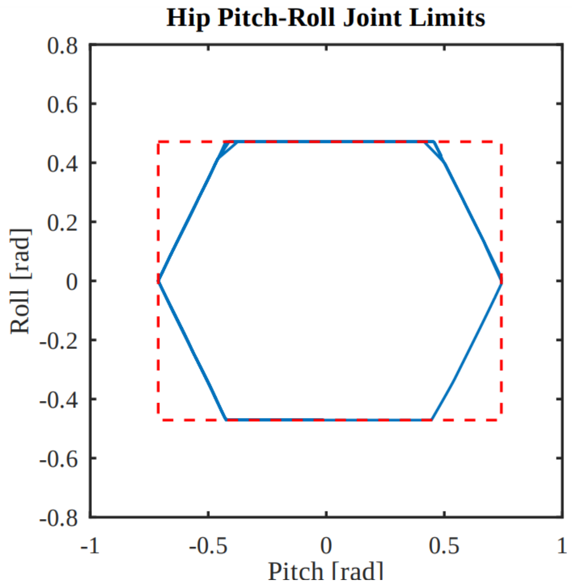



# Modeling

**Full-Model**

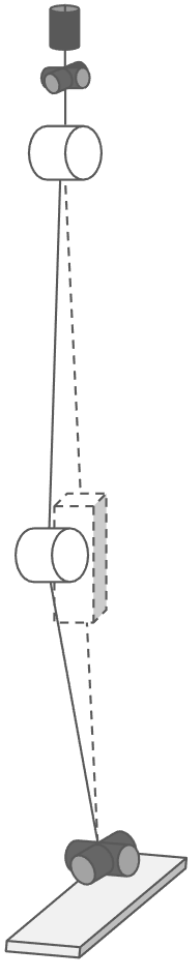


- 6 serial-parallel hybrid chains
- 38 DoFs
  - 6 actuated DoFs
  - 32 passive (constrained) DoFs
- Closed Linkage Library (CLL) for IK/FK and ID w/ floating-base and serial-parallel hybrid chains (multi-body constraint based)
- URDF-based model + GAZEBO simulation



# Modeling

**Simple-Model**

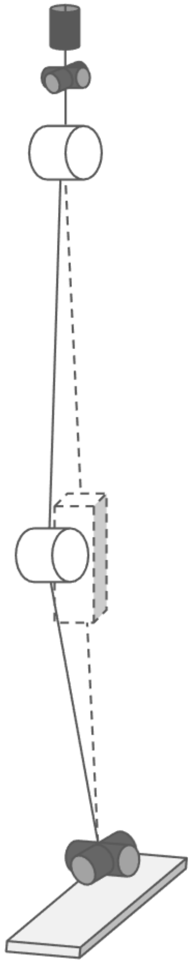


- 1 serial-parallel hybrid chains
- 8 DoFs
  - 5 actuated DoFs
  - 1 *virtual* actuated DoF
  - 2 passive (constrained) DoFs
- Simple constraint in IK and ID
- URDF-based model + GAZEBO simulation

# Modeling

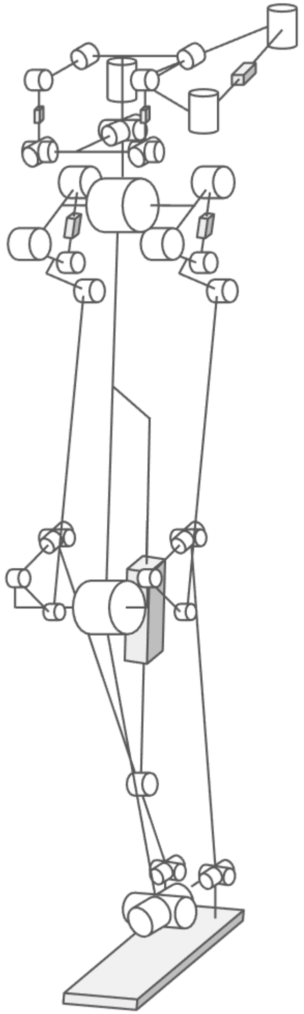
**Simple-Model**

More suitable for *planning* and *control,* especially considering a preview horizon.



**Full-Model**

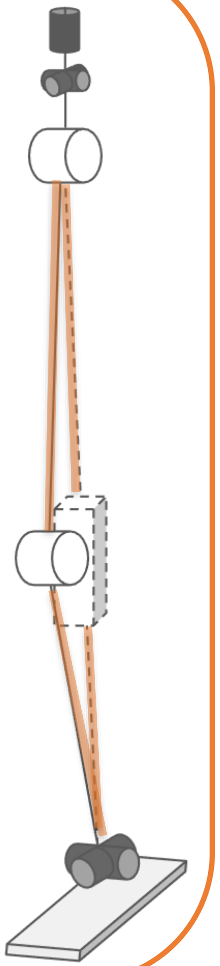
More suitable for *instantaneous* mapping from/to actuators considering all the non-linearities of the series-parallel hybrid chains.



# Modeling

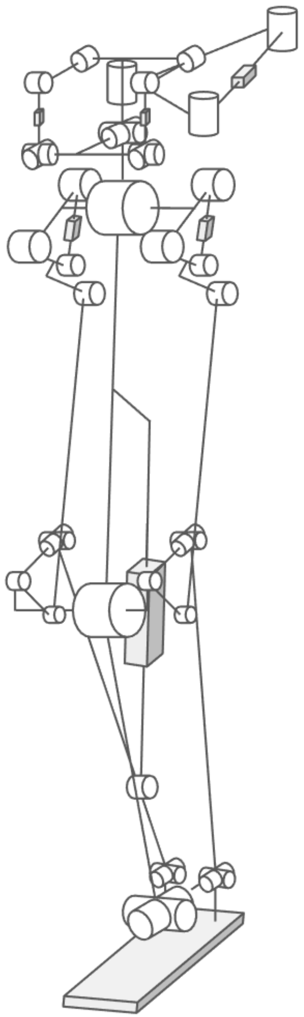
**Simple-Model**

More suitable for *planning* and *control,* especially considering a preview horizon.



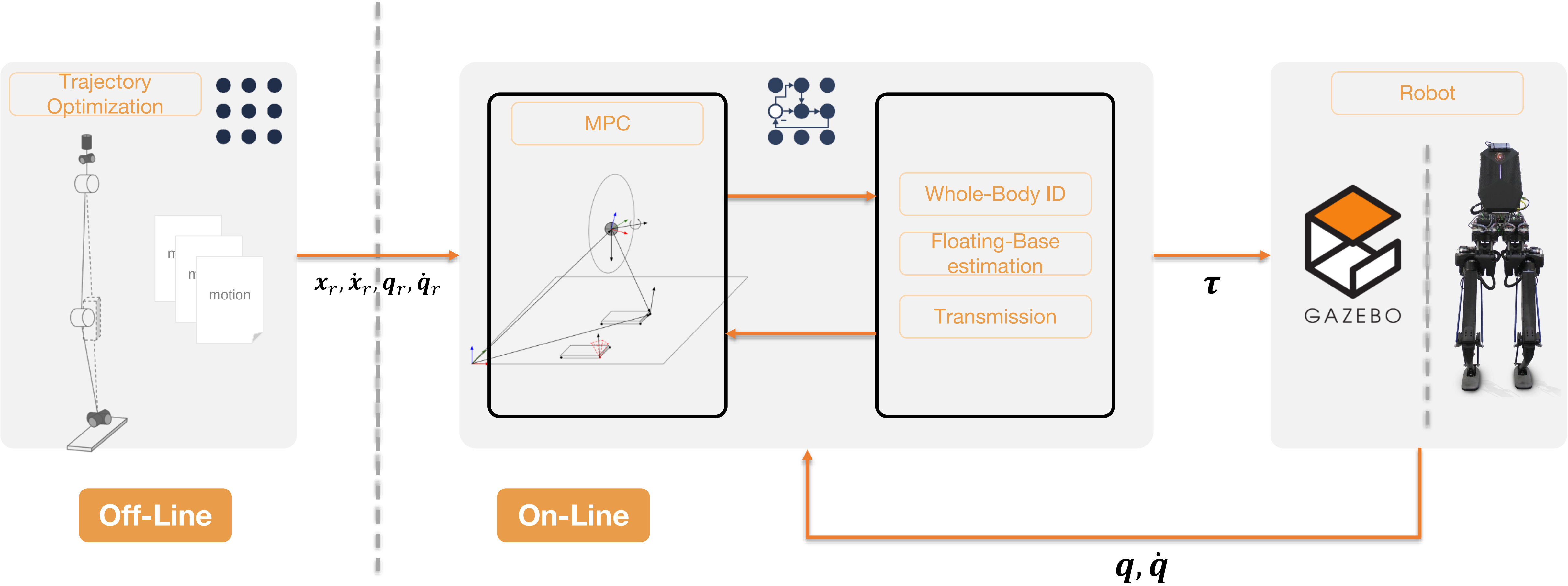
**Full-Model**

More suitable for *instantaneous* mapping from/to actuators considering all the non-linearities of the series-parallel hybrid chains.



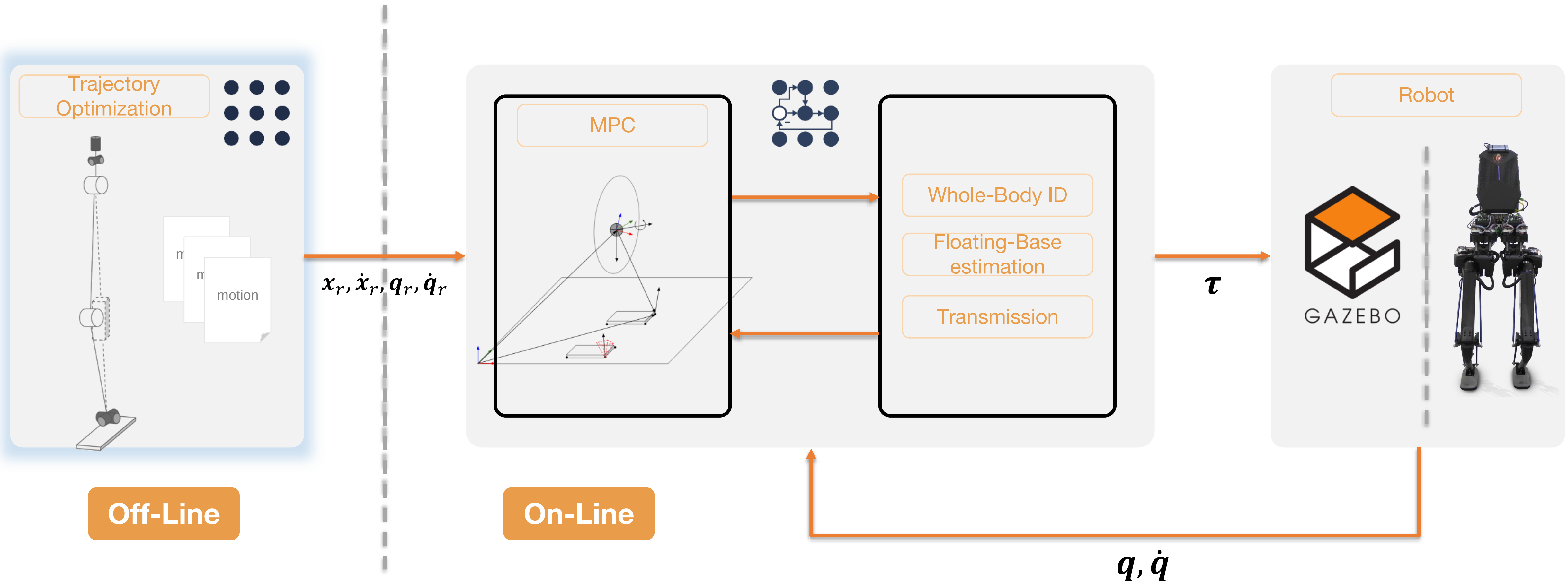
# Agile and Dynamic Locomotion

## Planning and Control Pipeline



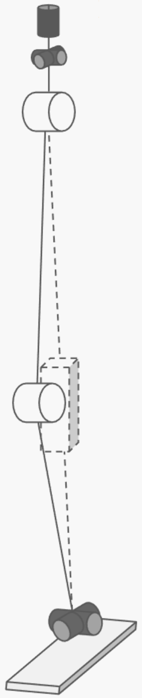
# Agile and Dynamic Locomotion

## Planning and Control Pipeline



# Trajectory Optimization

## Full dynamics of Simple-Model

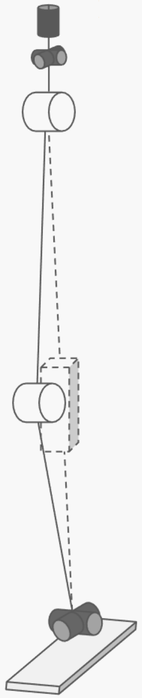


$$\begin{cases} \boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{v}} + \boldsymbol{h}(\boldsymbol{q},\boldsymbol{v}) = \boldsymbol{S\tau} + \boldsymbol{J}_c^T(\boldsymbol{q})\boldsymbol{f}_c + \boldsymbol{J}_v^T(\boldsymbol{q})\boldsymbol{\lambda} \\ \boldsymbol{J}_c(\boldsymbol{q})\dot{\boldsymbol{v}} + \dot{\boldsymbol{J}}_c(\boldsymbol{q},\boldsymbol{v})\boldsymbol{v} = \boldsymbol{0} \\ \boldsymbol{J}_v(\boldsymbol{q})\dot{\boldsymbol{v}} + \dot{\boldsymbol{J}}_v(\boldsymbol{q},\boldsymbol{v})\boldsymbol{v} = \boldsymbol{0} \end{cases}$$

**Contacts + Serial-Parallel Hybrid chains**

# Trajectory Optimization

## Full dynamics of Simple-Model



$$\begin{cases} M(q)\dot{v} + h(q,v) = S\tau + J_c^T(q)f_c + J_v^T(q)\lambda \\ J_c(q)\dot{v} + \dot{J}_c(q,v)v = 0 \\ J_v(q)\dot{v} + \dot{J}_v(q,v)v = 0 \end{cases}$$

**Contacts + Serial-Parallel Hybrid chains**

## OCP

$$x_k = \begin{bmatrix} q \\ v \end{bmatrix} \qquad u_k = \begin{bmatrix} \dot{v} \\ f_0 \\ \vdots \\ f_{c-1} \\ \lambda \end{bmatrix}$$

*N* states
*N-1* controls

$$x_{k+1} = f(x_k, u_k)$$

Multiple-shooting
Double integrator (inverse dynamics)

$$S\tau = M(q)\dot{v} + h(q,v) - J_c^T(q)f_c - J_v^T(q)\lambda$$
$$J_c(q)\dot{v} + \dot{J}_c(q,v)v = 0$$
$$J_v(q)\dot{v} + \dot{J}_v(q,v)v = 0$$

Explicit contacts scheduling

$$f_c \subset \mathcal{F}_c$$ Friction cones

$$\tau_m \le \tau \le \tau_M$$ Torque limits

PAL ROBOTICS

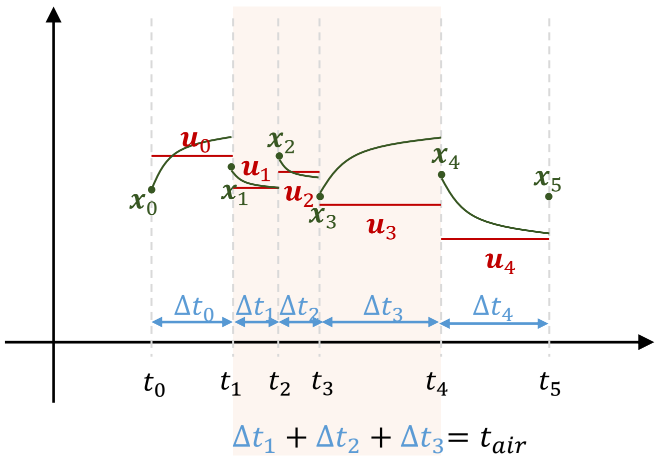
# Explicit contact scheduling

## Variable-Time Multiple-Shooting

- Time between nodes is a variable
- Less node
- Less accurate solution

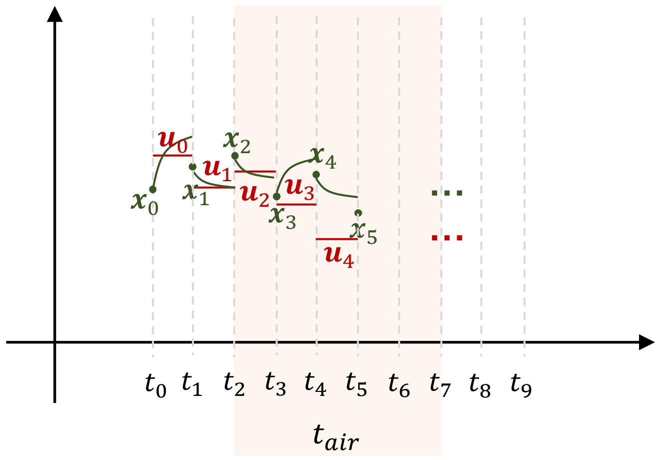
## Fixed-Time Multiple-Shooting

- Time between nodes is fixed
- More nodes
- More accurate solution



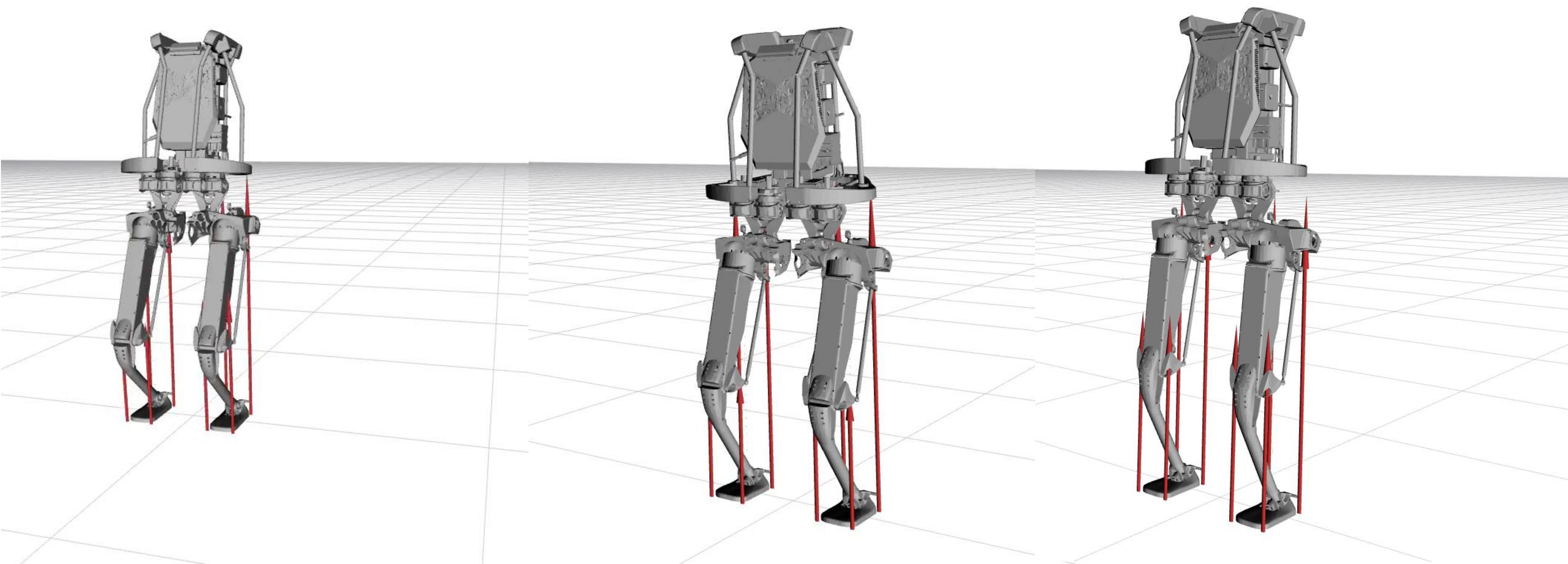

*N-1* Δtimes

Permits to find times of motion phases.

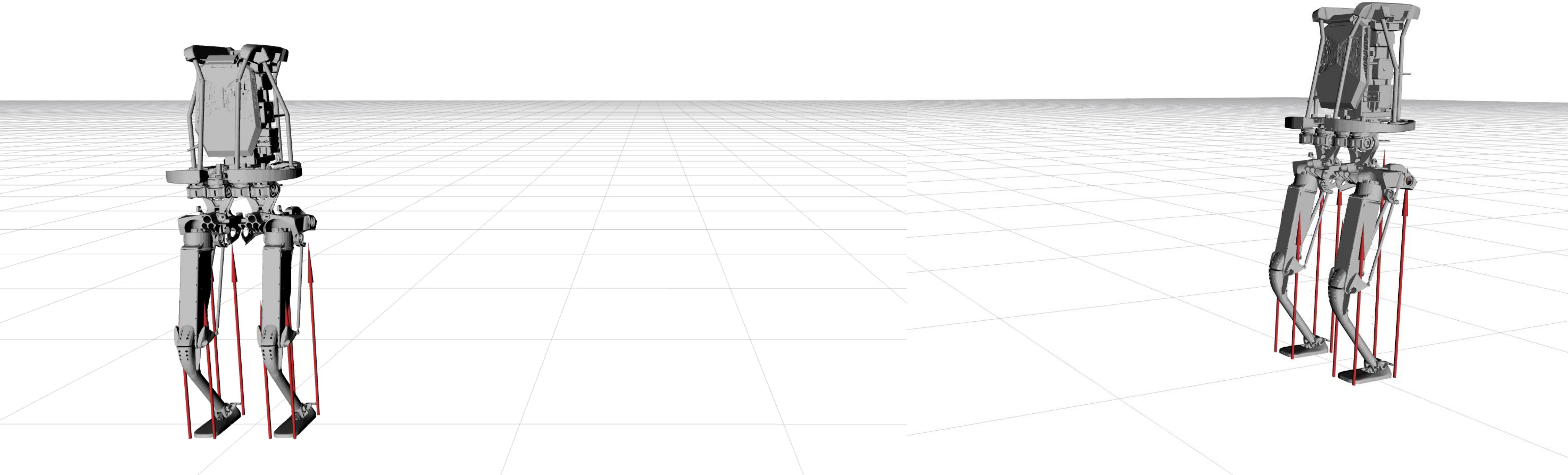



Discretized same as MPC.

# Library of Motions



# Library of Motions

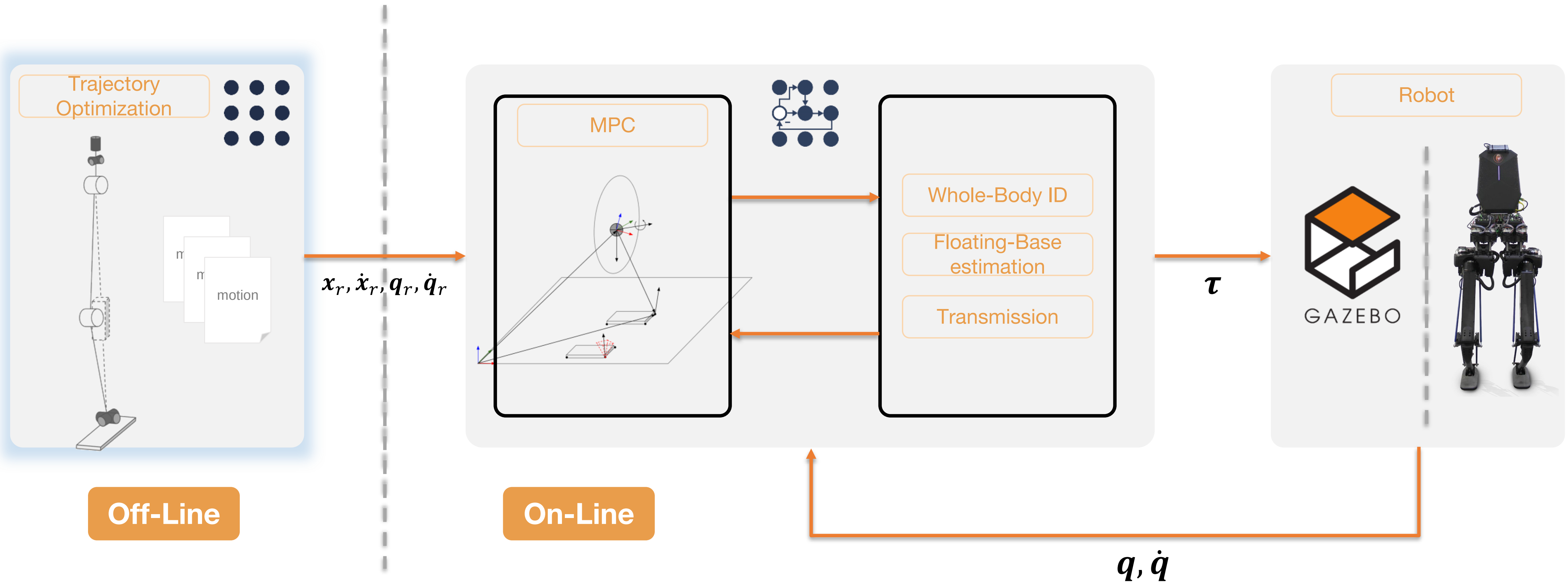


# Trajectory Optimization

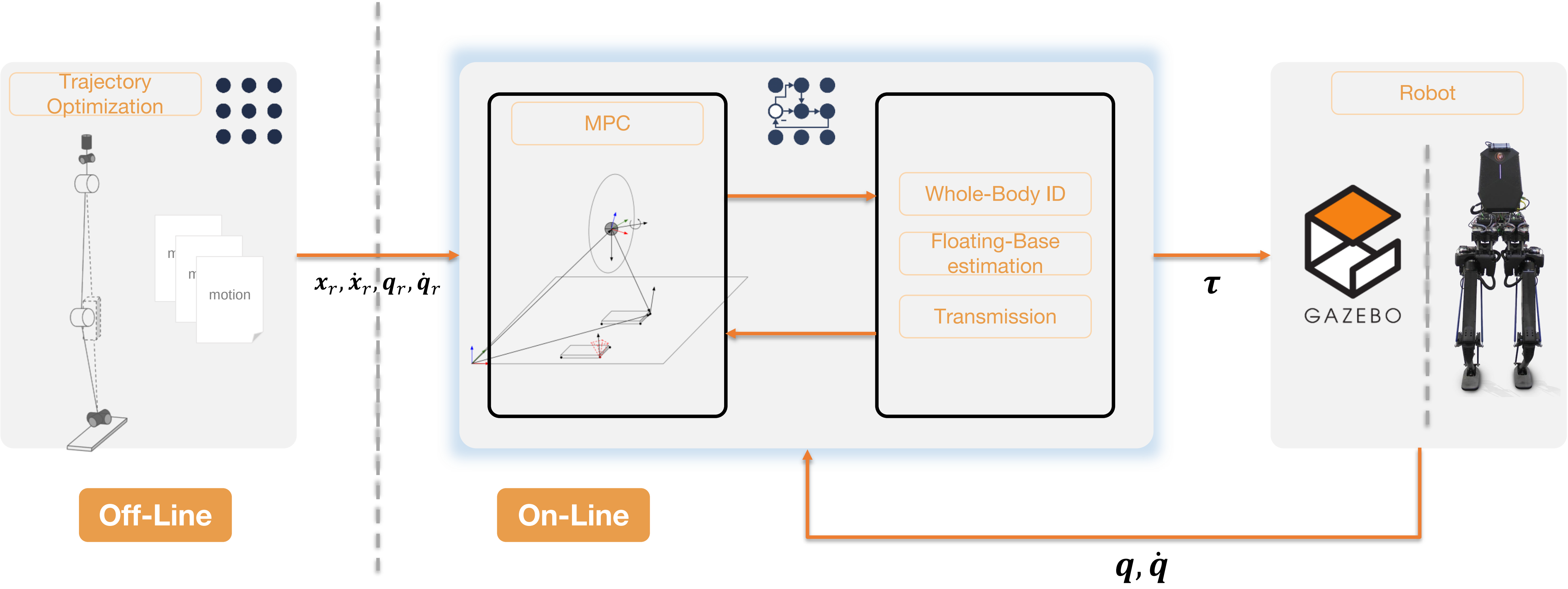
- Implementation based on Horizon framework [1] (CasADi + ipopt, in Python)
- Reuse of previous solutions as initial guess for new motions (eg: Jump, 50 nodes, 614 Vs 143 iterations, **~4.3x less iterations**)
- Floating-base dynamics constraint not always 100% satisfied
- Inverse Dynamics formulation *faster (less and faster iterations)* than Forward Dynamics

[1] Ruscelli F., Laurenzi A., Tsagarakis N.G., Mingo Hoffman E. *"Horizon: A Trajectory Optimization Framework for Robotic Systems."* Frontiers in Robotics & AI. 2022

# Agile and Dynamic Locomotion
## Planning and Control Pipeline

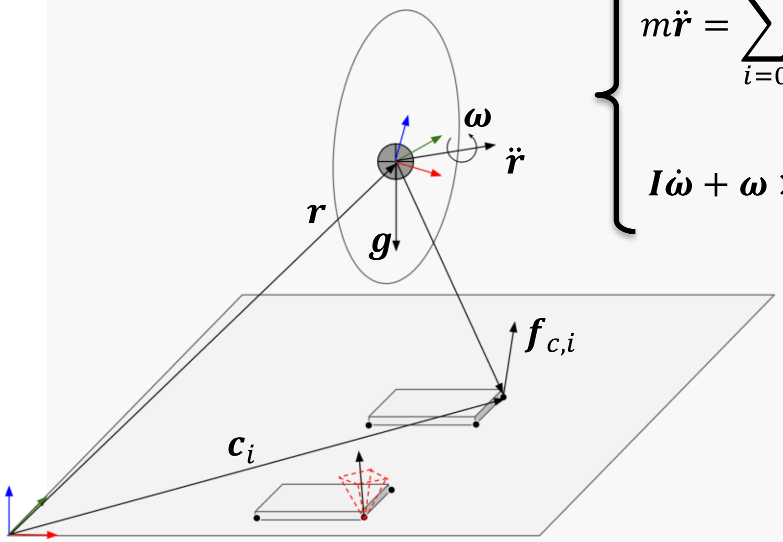


# Agile and Dynamic Locomotion
## Planning and Control Pipeline based on TO and MPC



# MPC

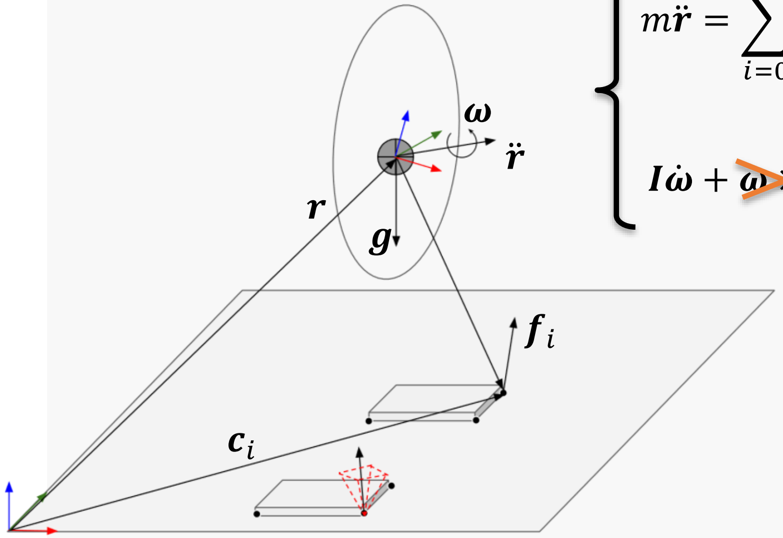
## SRBD




$$\begin{cases} m\ddot{\boldsymbol{r}} = \sum_{i=0}^{c-1} \boldsymbol{f}_i + m\boldsymbol{g} \\ \boldsymbol{I}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \boldsymbol{I}\boldsymbol{\omega} = \sum_{i=0}^{c-1} (\boldsymbol{c}_i - \boldsymbol{r}) \times \boldsymbol{f}_i \end{cases}$$

# MPC

## Linearized SRBD [2]



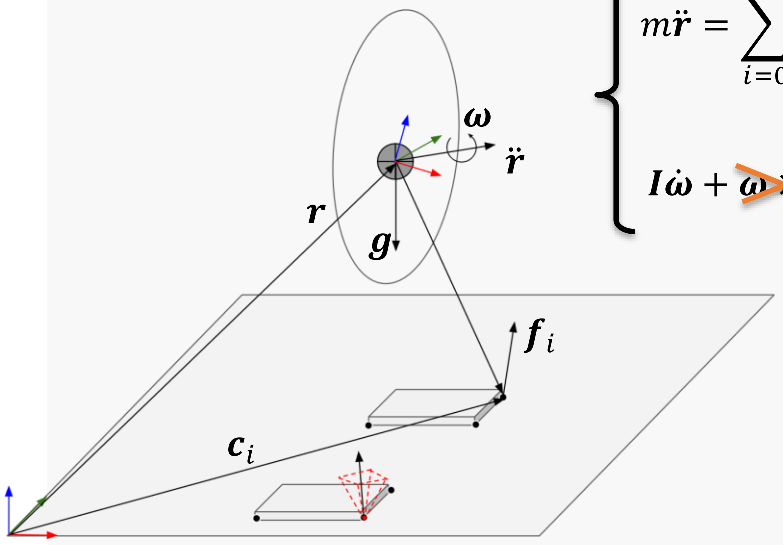

$$\begin{cases} m\ddot{\boldsymbol{r}} = \sum_{i=0}^{c-1} \boldsymbol{f}_i + m\boldsymbol{g} \\ \boldsymbol{I}\dot{\boldsymbol{\omega}} + \cancel{\boldsymbol{\omega} \times \boldsymbol{I}\boldsymbol{\omega}} = \sum_{i=0}^{c-1} (\boldsymbol{c}_i - \boldsymbol{r}) \times \boldsymbol{f}_i \end{cases}$$

$$\begin{cases} \ddot{\boldsymbol{r}} = \frac{1}{m}\left(\sum_{i=0}^{c-1} \boldsymbol{f}_i + m\boldsymbol{g}\right) \\ \dot{\boldsymbol{\omega}} = \boldsymbol{I}^{-1}\left(\sum_{i=0}^{c-1} (\boldsymbol{c}_i - \boldsymbol{r}) \times \boldsymbol{f}_{c,i}\right) \\ \boldsymbol{I} = \boldsymbol{R}_{\Theta} \boldsymbol{I}_{\mathcal{B}} \boldsymbol{R}_{\Theta}^{T} \end{cases}$$

[2] J. Di Carlo, P. M. Wensing, B. Katz, G. Bledt, and S. Kim, "*Dynamic locomotion in the MIT cheetah 3 through convex model-predictive control*," in IEEE RSJ International Conference on Intelligent Robots and Systems, 2018

# MPC

## Linearized SRBD [2]




$$
\begin{cases}
m\ddot{\boldsymbol{r}} = \sum_{i=0}^{c-1} \boldsymbol{f}_i + m\boldsymbol{g} \\
\boldsymbol{I}\dot{\boldsymbol{\omega}} + \cancel{\boldsymbol{\omega} \times \boldsymbol{I}\boldsymbol{\omega}} = \sum_{i=0}^{c-1} (\boldsymbol{c}_i - \boldsymbol{r}) \times \boldsymbol{f}_i
\end{cases}
$$

$$
\begin{cases}
\ddot{\boldsymbol{r}} = \frac{1}{m}\left(\sum_{i=0}^{c-1} \boldsymbol{f}_i + m\boldsymbol{g}\right) \\
\dot{\boldsymbol{\omega}} = \boldsymbol{I}^{-1}\left(\sum_{i=0}^{c-1} (\boldsymbol{c}_i - \boldsymbol{r}) \times \boldsymbol{f}_{c,i}\right) \\
\boldsymbol{I} = \boldsymbol{R}_{\boldsymbol{\Theta}} \boldsymbol{I}_{\mathcal{B}} \boldsymbol{R}_{\boldsymbol{\Theta}}^T \\
\boldsymbol{x}_k = \begin{bmatrix} \boldsymbol{\Theta} \\ \boldsymbol{r} \\ \boldsymbol{\omega} \\ \dot{\boldsymbol{r}} \end{bmatrix}, \quad \boldsymbol{u}_k = \begin{bmatrix} \boldsymbol{f}_0 \\ \vdots \\ \boldsymbol{f}_{c-1} \end{bmatrix} \\
\boldsymbol{x}_{k+1} = \boldsymbol{A}_k \boldsymbol{x}_k + \boldsymbol{B}_k \boldsymbol{u}_k \\
\boldsymbol{f}_c \subset \mathcal{F}_c
\end{cases}
$$

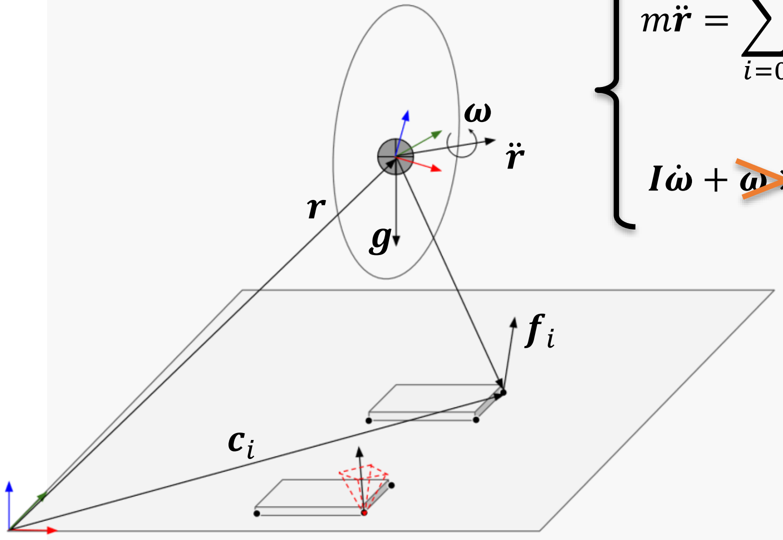
- k = 0, feedback from the robot
- k > 0 from desired state trajectory

Playing with liniearized friction cones is possible to add/remove contacts

[2] J. Di Carlo, P. M. Wensing, B. Katz, G. Bledt, and S. Kim, "*Dynamic locomotion in the MIT cheetah 3 through convex model-predictive control*," in IEEE RSJ International Conference on Intelligent Robots and Systems, 2018

# MPC

## Linearized SRBD [2]




$$\begin{cases} m\ddot{\boldsymbol{r}} = \sum_{i=0}^{c-1} \boldsymbol{f}_i + m\boldsymbol{g} \\ \boldsymbol{I}\dot{\boldsymbol{\omega}} + \cancel{\boldsymbol{\omega} \times \boldsymbol{I}}\boldsymbol{\omega} = \sum_{i=0}^{c-1} (\boldsymbol{c}_i - \boldsymbol{r}) \times \boldsymbol{f}_i \end{cases}$$

$$\begin{cases} \ddot{\boldsymbol{r}} = \frac{1}{m}\left(\sum_{i=0}^{c-1} \boldsymbol{f}_i + m\boldsymbol{g}\right) \\ \dot{\boldsymbol{\omega}} = \boldsymbol{I}^{-1}\left(\sum_{i=0}^{c-1} (\boldsymbol{c}_i - \boldsymbol{r}) \times \boldsymbol{f}_{c,i}\right) \\ \boldsymbol{I} = \boldsymbol{R}_{\boldsymbol{\Theta}} \boldsymbol{I}_{\mathcal{B}} \boldsymbol{R}_{\boldsymbol{\Theta}}^T \\ \boldsymbol{x}_k = \begin{bmatrix} \boldsymbol{\Theta} \\ \boldsymbol{r} \\ \boldsymbol{\omega} \\ \dot{\boldsymbol{r}} \end{bmatrix}, \quad \boldsymbol{u}_k = \begin{bmatrix} \boldsymbol{f}_0 \\ \vdots \\ \boldsymbol{f}_{c-1} \end{bmatrix} \\ \boldsymbol{x}_{k+1} = \boldsymbol{A}_k \boldsymbol{x}_k + \boldsymbol{B}_k \boldsymbol{u}_k \\ \boldsymbol{f}_c \subset \mathcal{F}_c \end{cases}$$

**Cost**

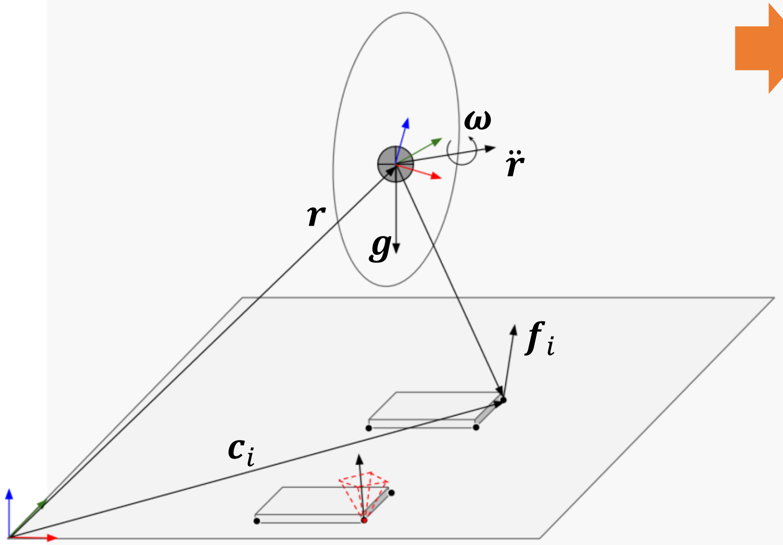
$$\min_{\boldsymbol{x},\boldsymbol{u}} \|\boldsymbol{x} - \boldsymbol{x}_r\|_{\boldsymbol{Q}} + \|\boldsymbol{u}\|_{\boldsymbol{R}}$$

Reference state along a trajectory

[2] J. Di Carlo, P. M. Wensing, B. Katz, G. Bledt, and S. Kim, "*Dynamic locomotion in the MIT cheetah 3 through convex model-predictive control*," in IEEE RSJ International Conference on Intelligent Robots and Systems, 2018

# MPC

## Linearized SRBD [2]



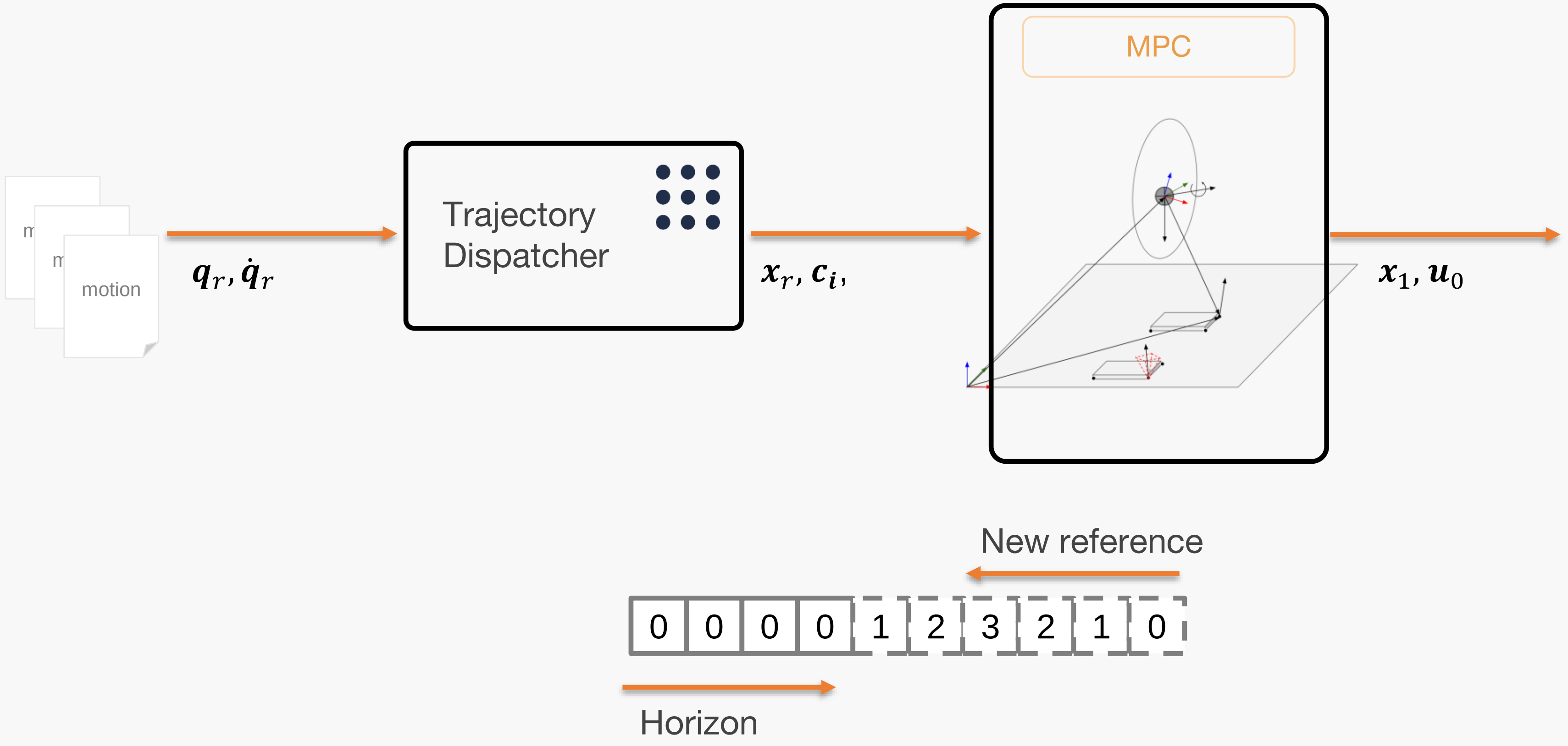

## QP

$$\min_{\boldsymbol{w}} \boldsymbol{w}^T \boldsymbol{H} \boldsymbol{w} + \boldsymbol{g}^T \boldsymbol{w}$$
$$s.t. \quad \boldsymbol{C}\boldsymbol{w} \leq \boldsymbol{c}$$

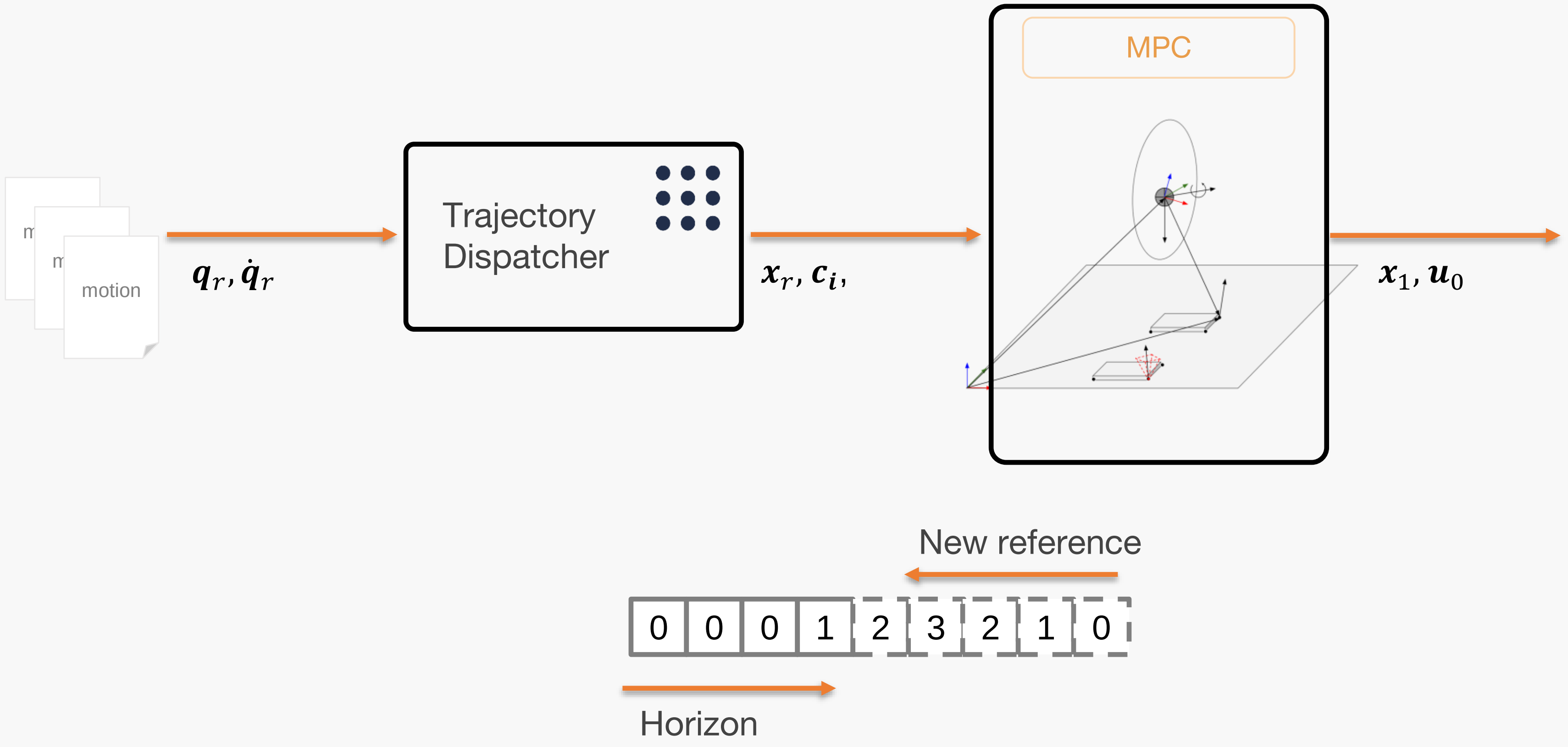
- Efficiently solved using OSQP [3]
- Sparse implementation using Eigen
- 40 nodes, dt = 30 ms
  - solution time 3 ms

[3] Stellato, B. and Banjac, G. and Goulart, P. and Bemporad, A. and Boyd, S. , "*OSQP: an operator splitting solver for quadratic programs* ," in Mathematical Programming Computation, 2020

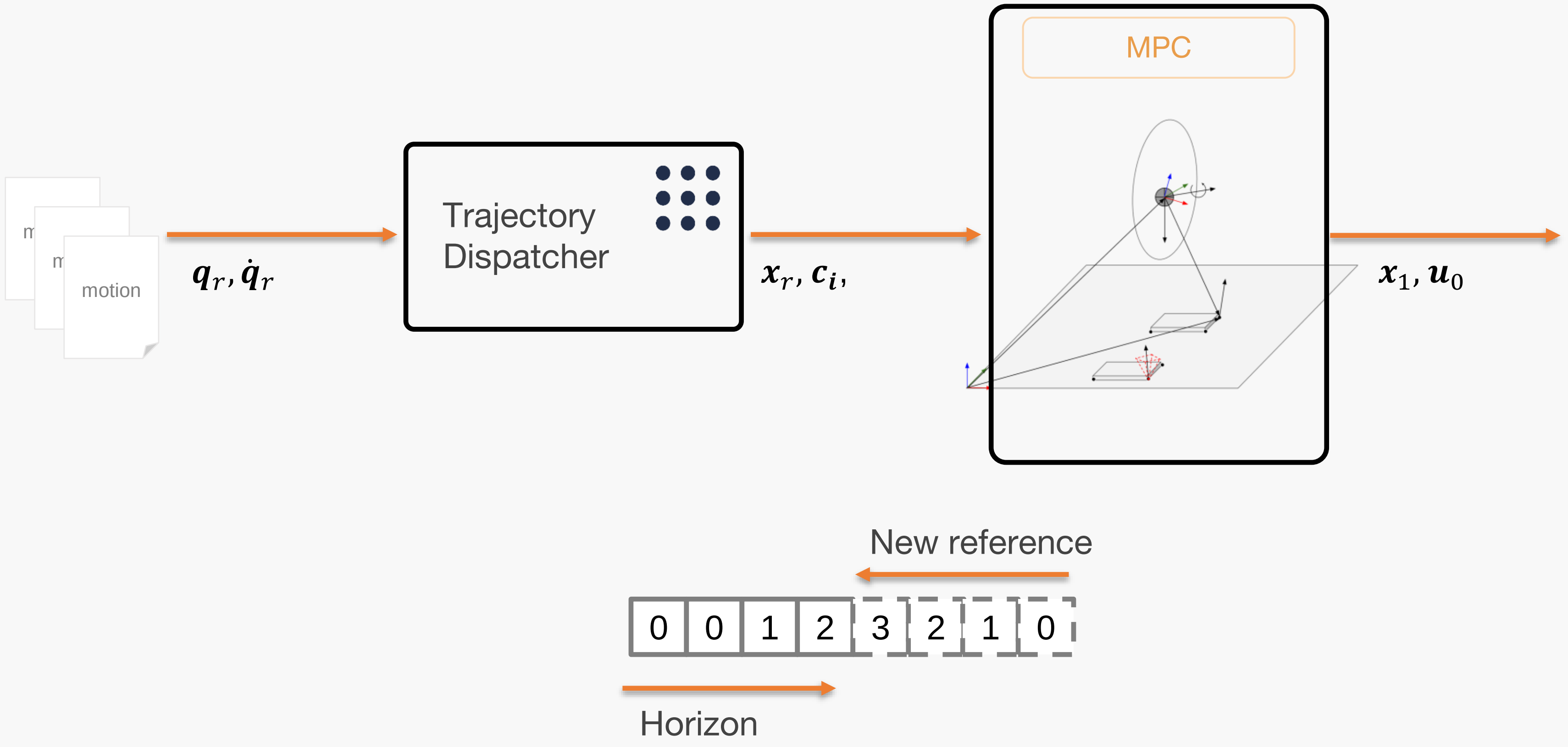
# TO ► MPC


motion
$\boldsymbol{q}_r, \dot{\boldsymbol{q}}_r$
Trajectory Dispatcher
$\boldsymbol{x}_r, \boldsymbol{c}_i,$
MPC
$\boldsymbol{x}_1, \boldsymbol{u}_0$
New reference
0 0 0 0 1 2 3 2 1 0
Horizon

# TO ► MPC

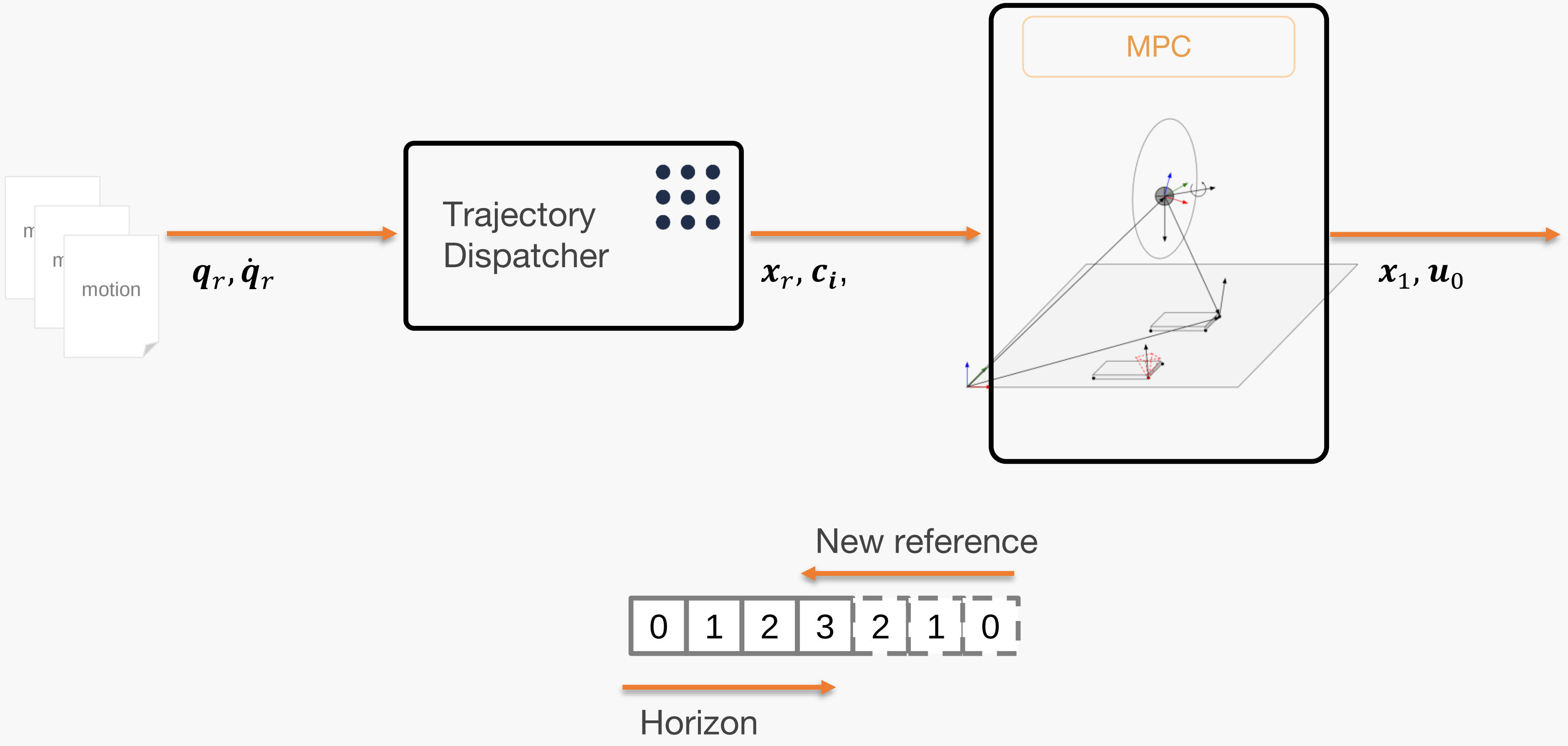

motion
$q_r, \dot{q}_r$
Trajectory Dispatcher
$x_r, c_i,$
MPC
$x_1, u_0$
New reference
0 0 0 1 2 3 2 1 0
Horizon

# TO ► MPC

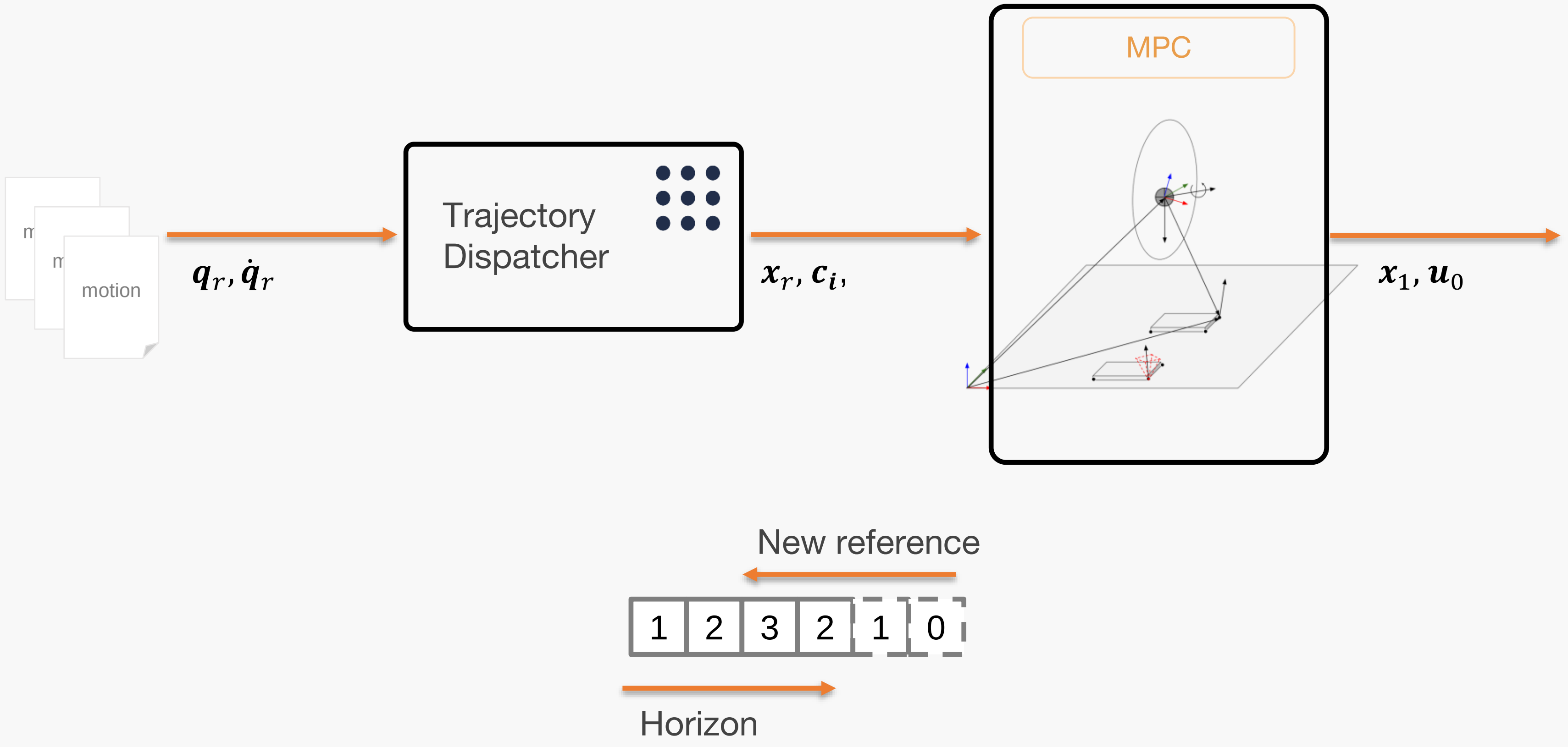


motion
$q_r, \dot{q}_r$
Trajectory Dispatcher
$x_r, c_i,$
MPC
$x_1, u_0$
New reference
0 0 1 2 3 2 1 0
Horizon

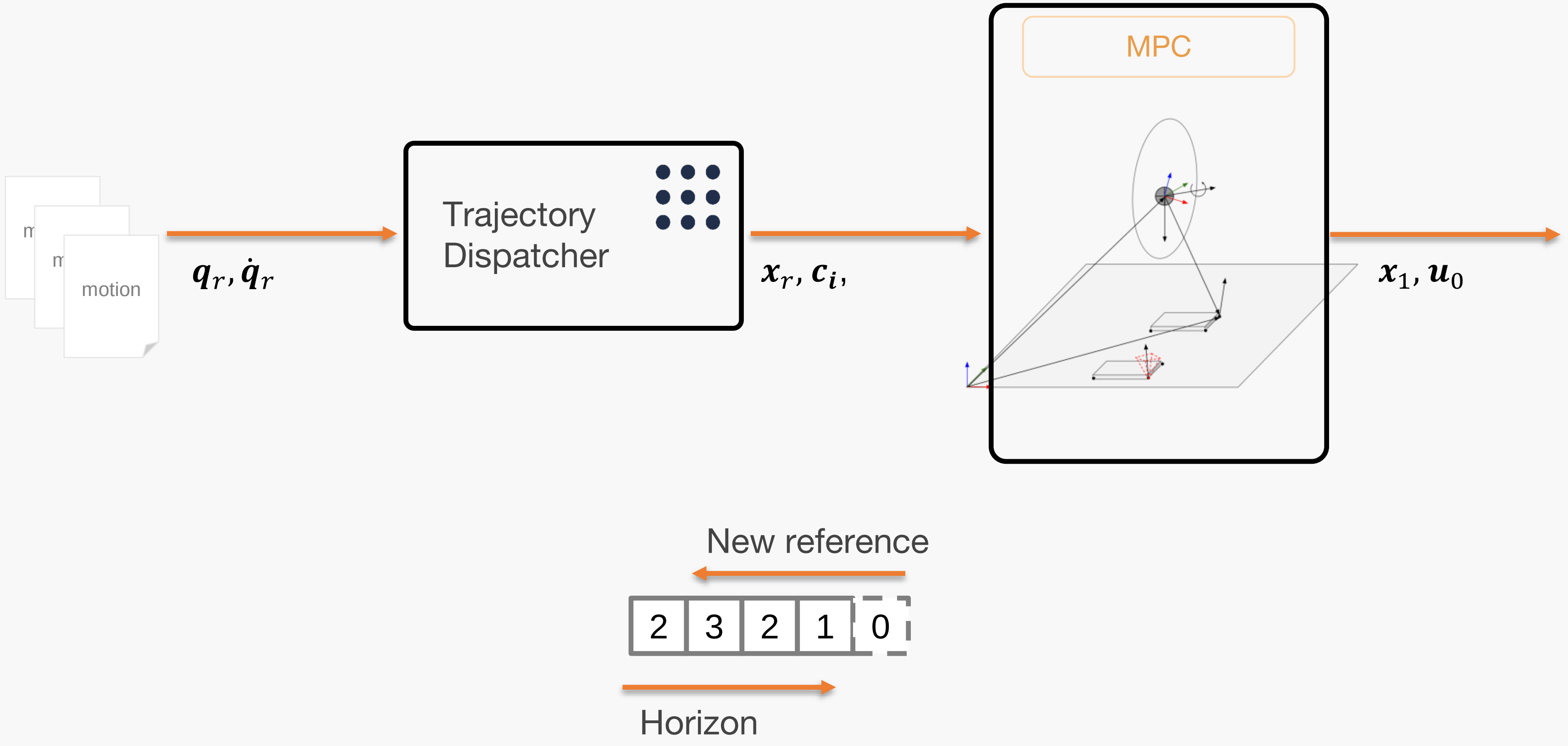
# TO ► MPC


motion
$q_r, \dot{q}_r$
Trajectory Dispatcher
$x_r, c_i,$
MPC
$x_1, u_0$
New reference
0 1 2 3 2 1 0
Horizon

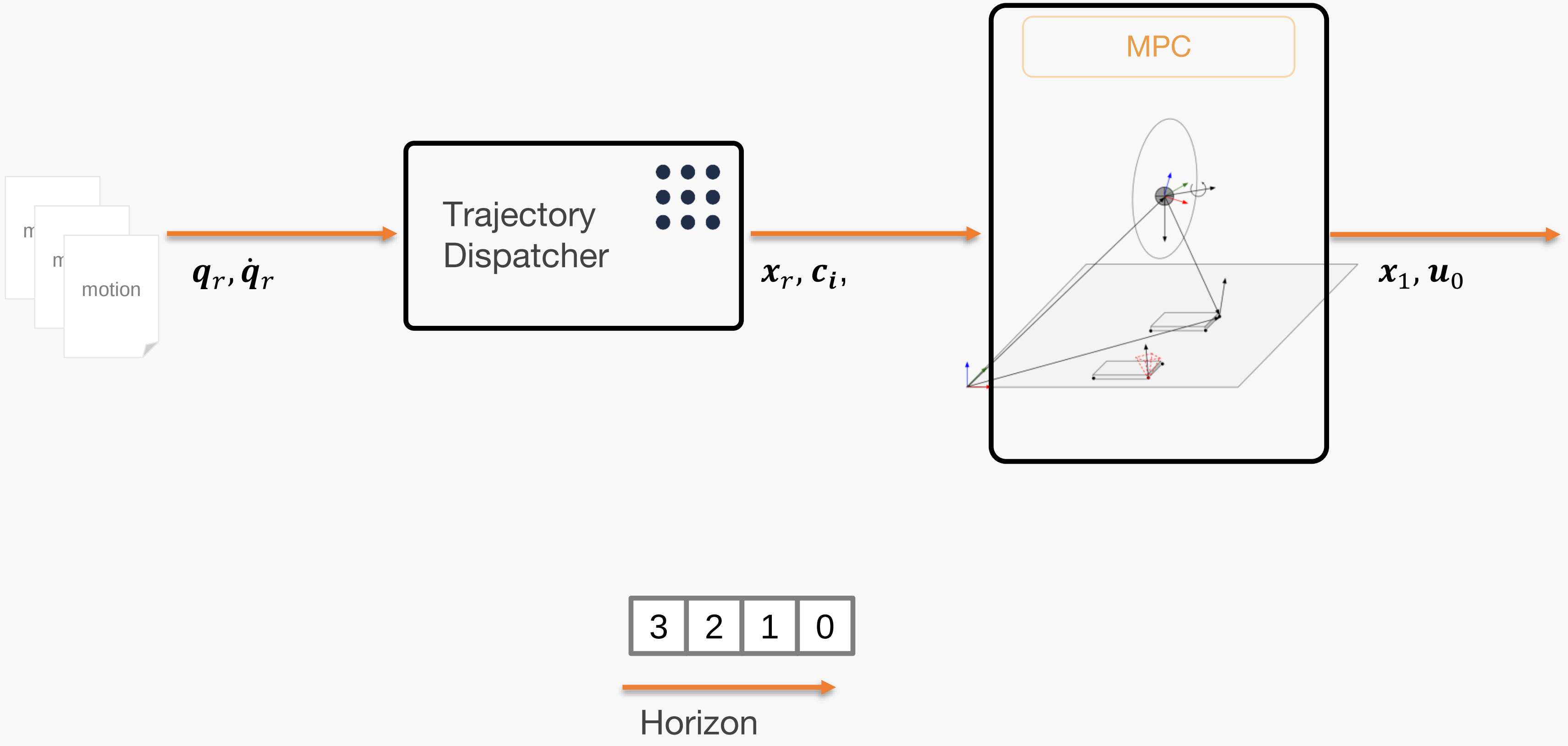
# TO ► MPC


motion
$q_r, \dot{q}_r$
Trajectory Dispatcher
$x_r, c_i,$
MPC
$x_1, u_0$
New reference
1 2 3 2 1 0
Horizon

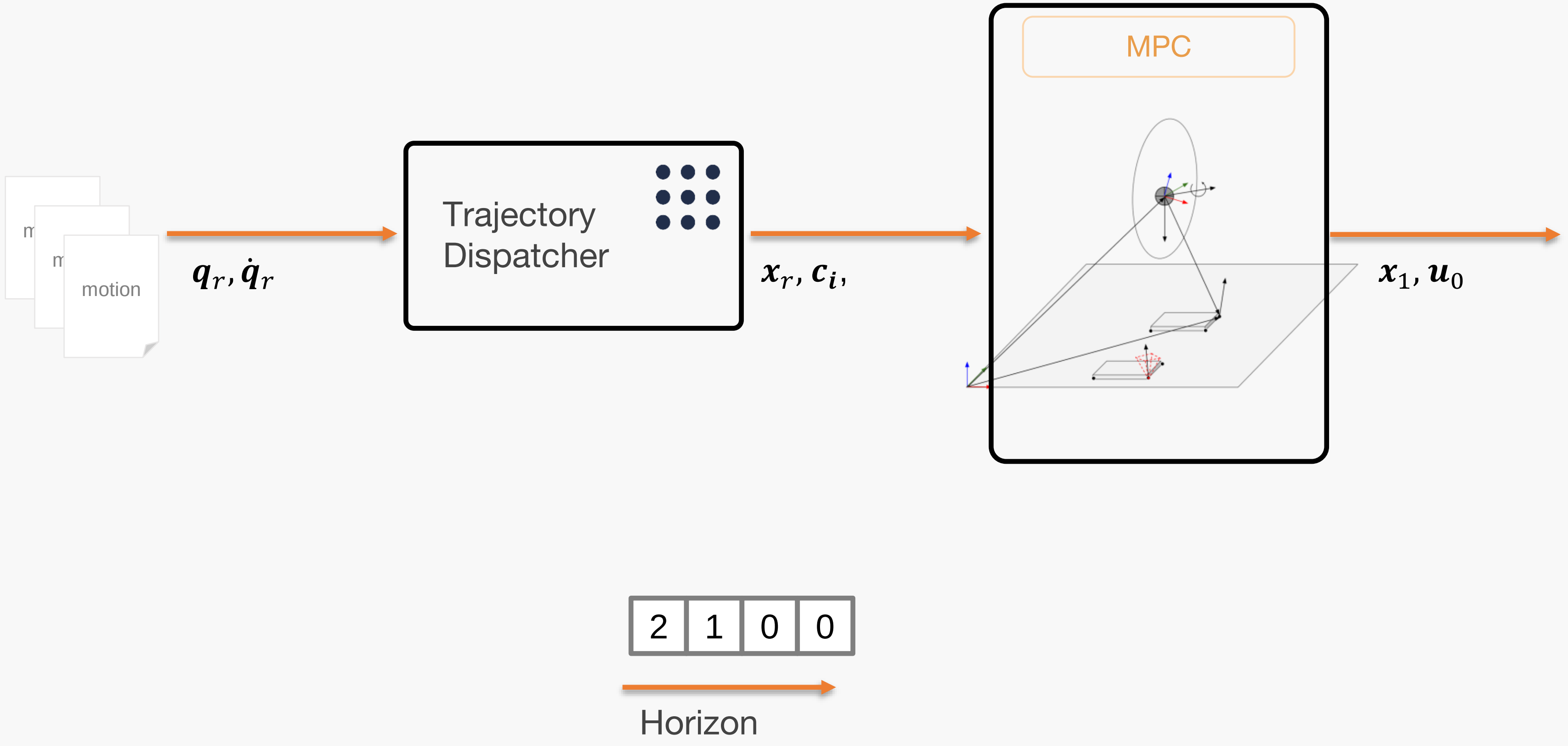
# TO ► MPC


motion
$\boldsymbol{q}_r, \dot{\boldsymbol{q}}_r$
Trajectory Dispatcher
$\boldsymbol{x}_r, \boldsymbol{c}_i,$
MPC
$\boldsymbol{x}_1, \boldsymbol{u}_0$
New reference
2 3 2 1 0
Horizon

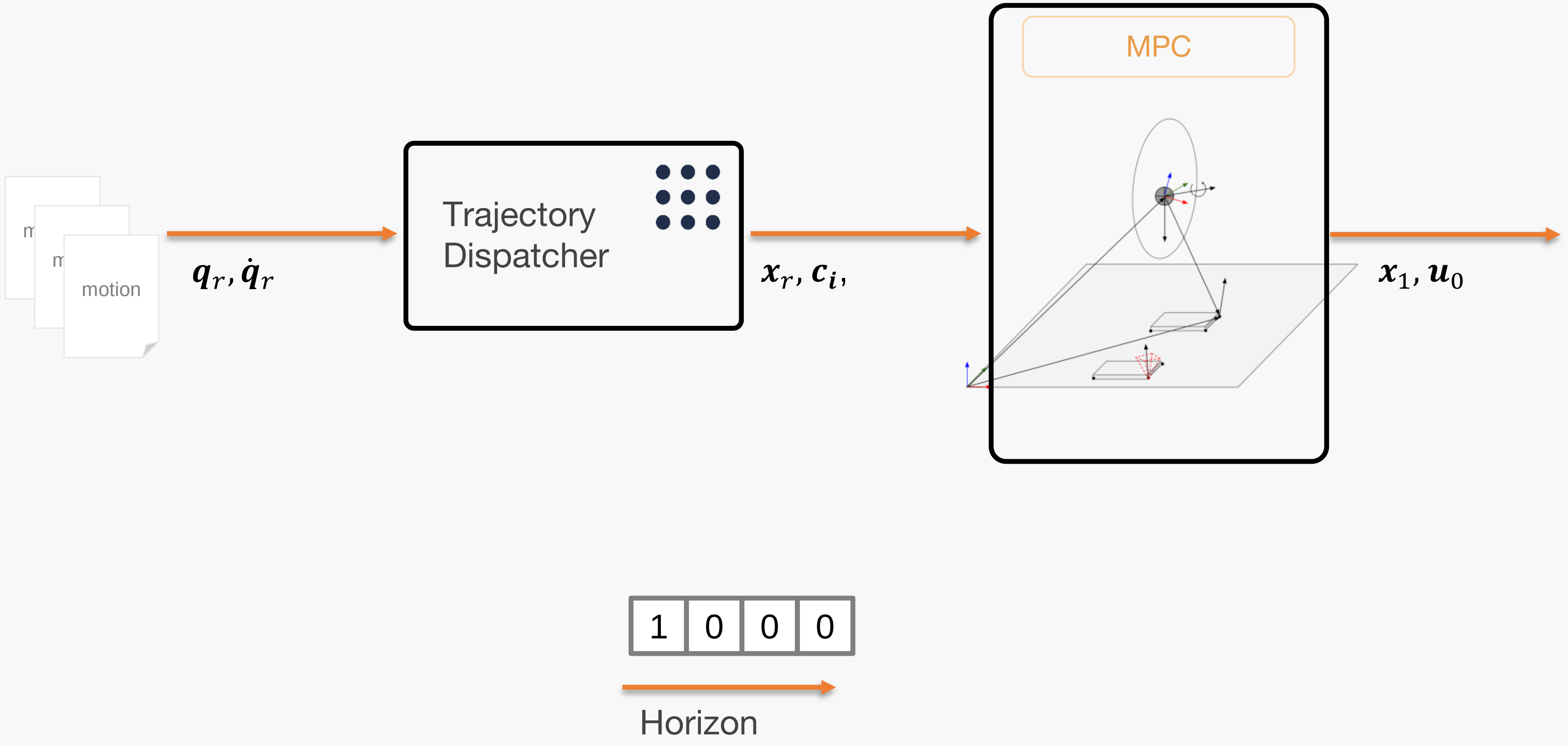
# TO ▶ MPC


motion
$\boldsymbol{q}_r, \dot{\boldsymbol{q}}_r$
Trajectory Dispatcher
$\boldsymbol{x}_r, \boldsymbol{c}_i,$
MPC
$\boldsymbol{x}_1, \boldsymbol{u}_0$
3 2 1 0
Horizon

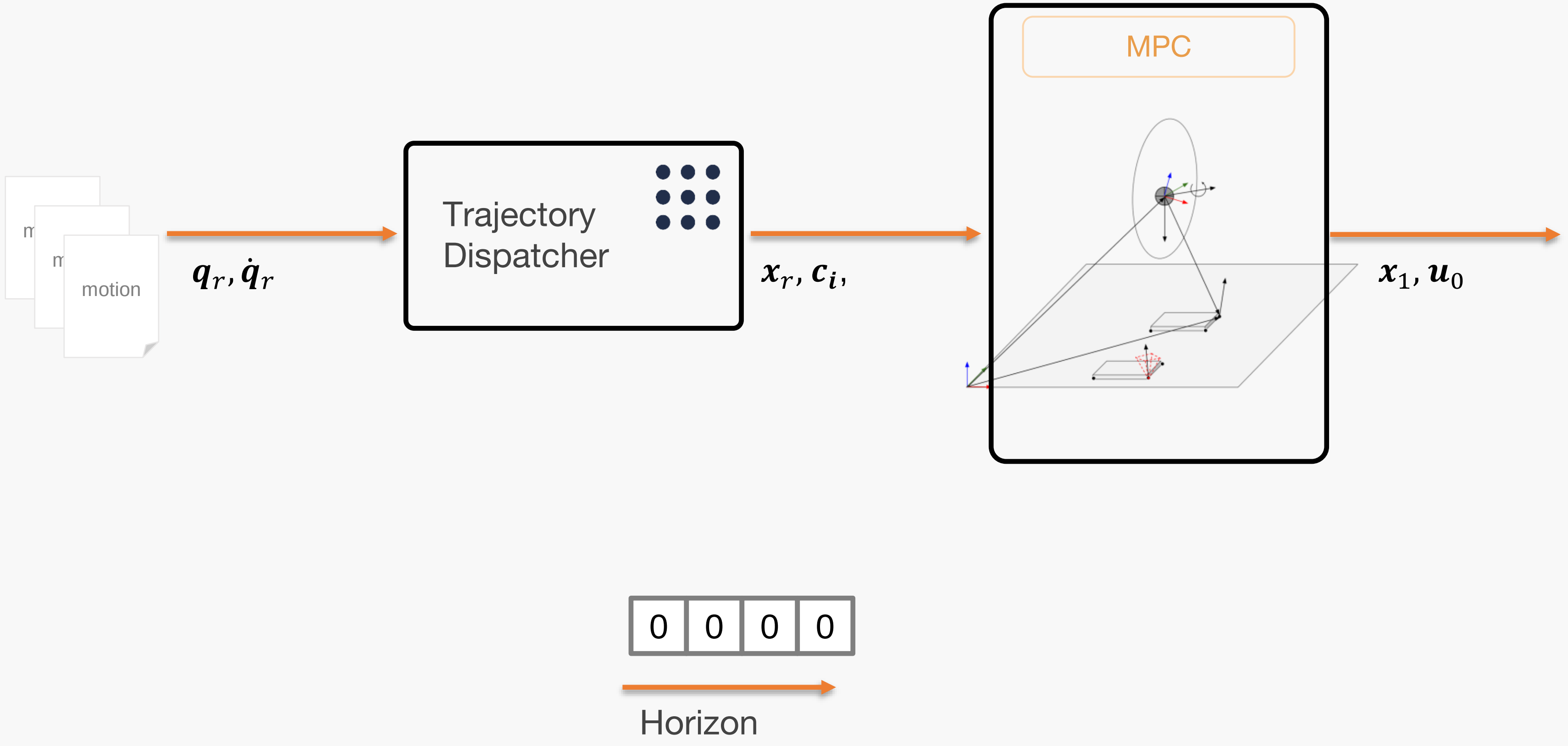
# TO ► MPC


motion
$q_r, \dot{q}_r$
Trajectory Dispatcher
$x_r, c_i,$
MPC
$x_1, u_0$
2 1 0 0
Horizon

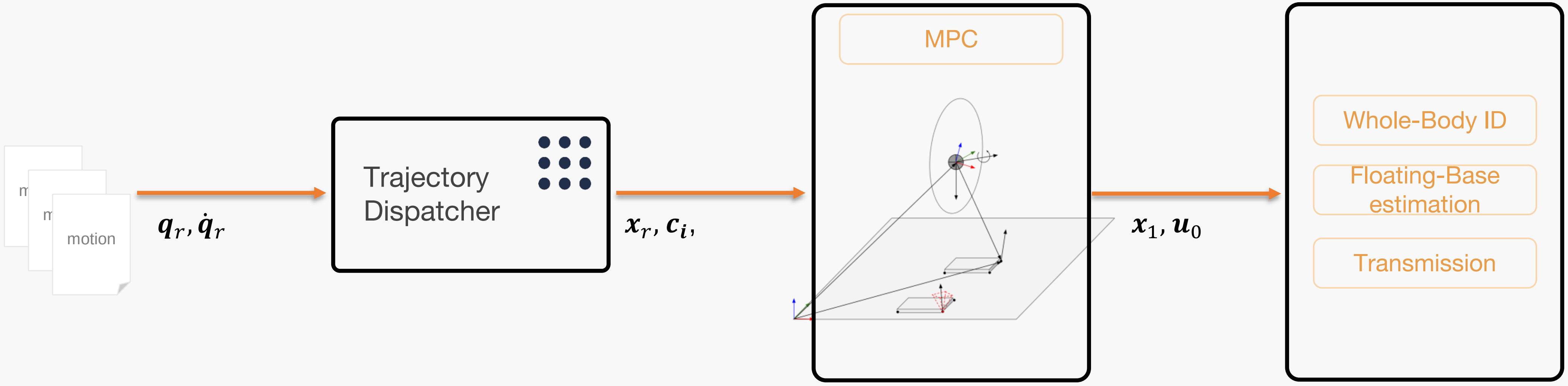
# TO ► MPC


motion
$q_r, \dot{q}_r$
Trajectory Dispatcher
$x_r, c_i,$
MPC
$x_1, u_0$
1 0 0 0
Horizon

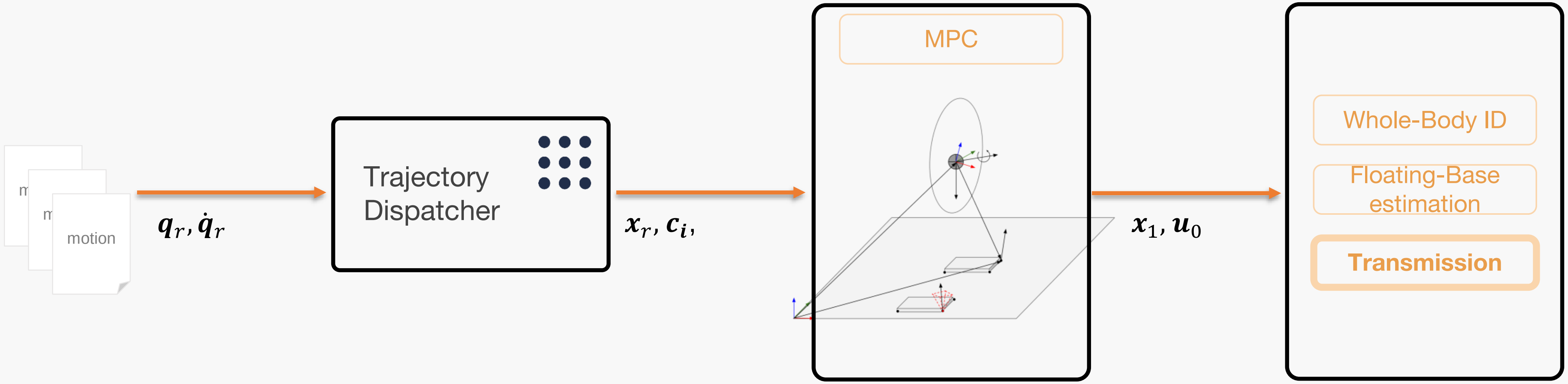
# TO ► MPC


motion
$q_r, \dot{q}_r$
Trajectory Dispatcher
$x_r, c_i,$
MPC
$x_1, u_0$
0 0 0 0
Horizon

# TO ▶ MPC ▶ WBC

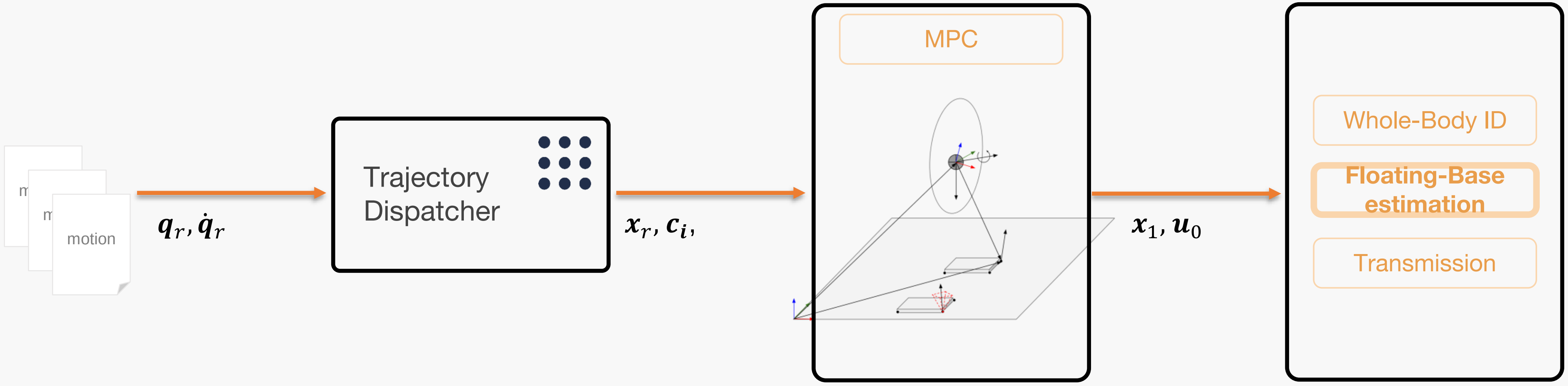

motion
$q_r, \dot{q}_r$
Trajectory Dispatcher
$x_r, c_i,$
MPC
$x_1, u_0$
Whole-Body ID
Floating-Base estimation
Transmission

# TO ▶ MPC ▶ WBC



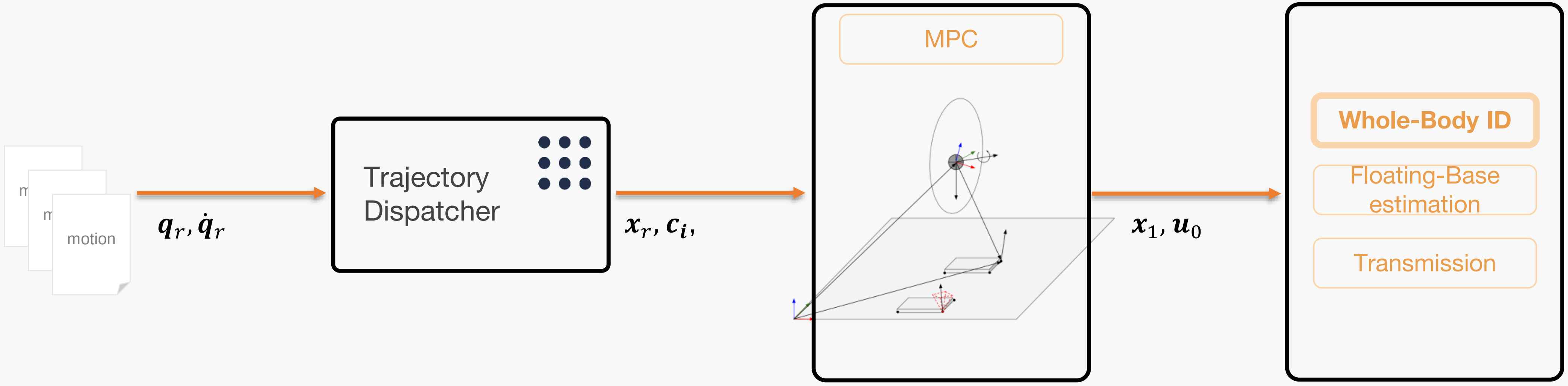

- Separated sub-mechanisms implemented as `ros_control` transmission
- mapping from/to actuator quantities

# TO ▶ MPC ▶ WBC



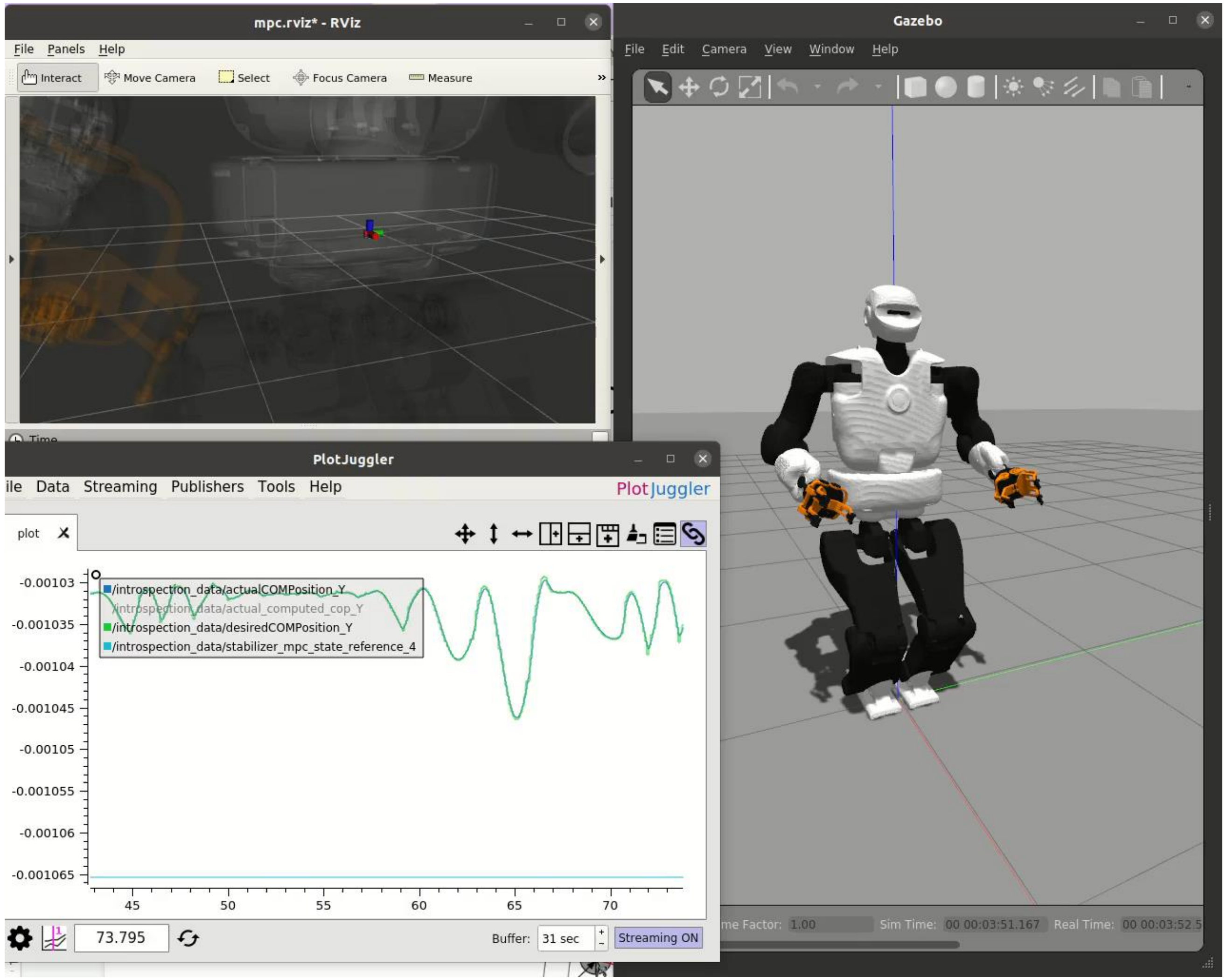

- Simple model
- QP-based velocity estimation + FK

# TO ▶ MPC ▶ WBC



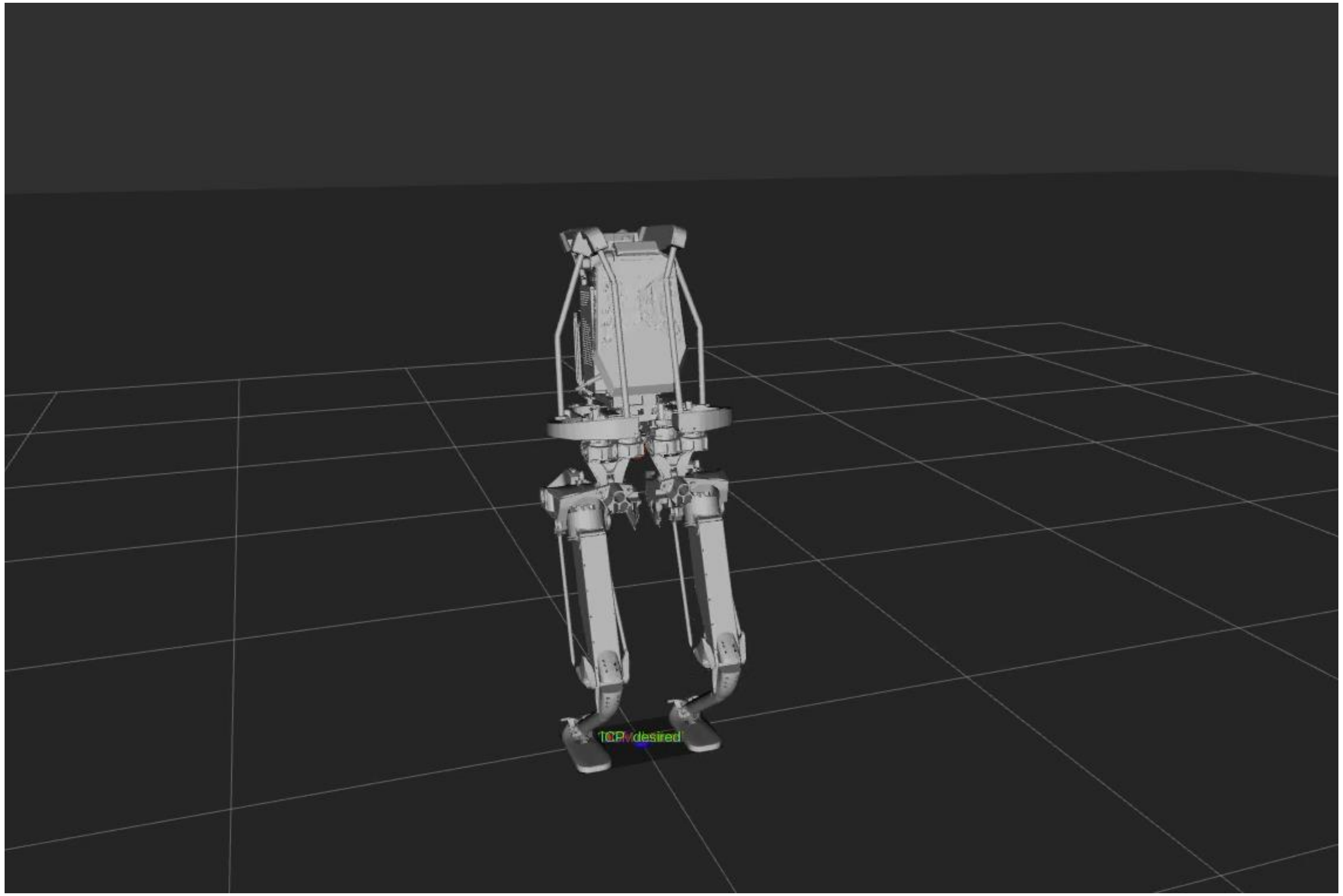

- Simple model
- Acceleration-based QP with closed kinematics constraints, single priority
- Constrained forces are computed separately and mapped as torques in the ID

# TO ► MPC ► WBC: Lateral Swing



PAL ROBOTICS

# TO ▶ MPC ▶ WBC: Jump



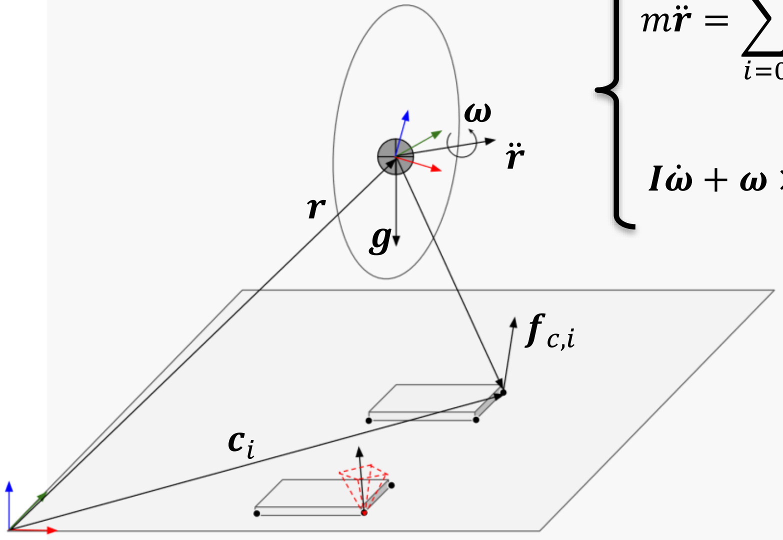

- Centroidal MPC:
  - Solution time ~3 ms
  - 40 nodes
  - dt = 30 ms
  - mpc thread = 30 ms

# TO ► MPC ► WBC: Jump

- The MPC is used for both jump and landing phases
- During aerial phase, the joint references from the TO are used
- **During landing phase using a constant state reference is more effective**
- Closing the loop with the WBC requires several tuning on both the MPC and the WBC to achieve good tracking

# MPC

## SRBD

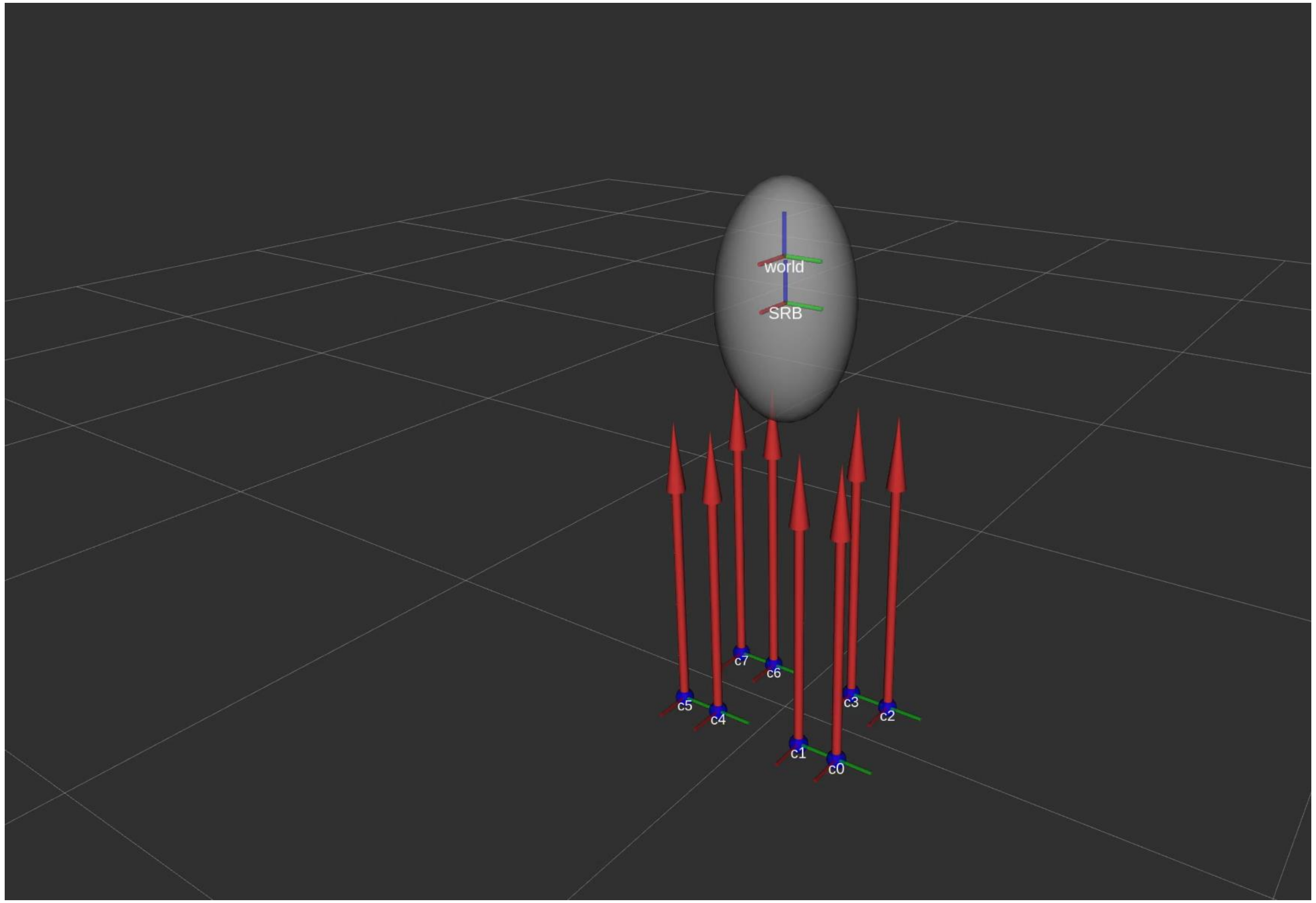



$$\begin{cases} m\ddot{\boldsymbol{r}} = \sum_{i=0}^{c-1} \boldsymbol{f}_i + m\boldsymbol{g} \\ \boldsymbol{I}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \boldsymbol{I}\boldsymbol{\omega} = \sum_{i=0}^{c-1} (\boldsymbol{c}_i - \boldsymbol{r}) \times \boldsymbol{f}_i \end{cases}$$

$$\boldsymbol{x}_k = \begin{bmatrix} \boldsymbol{r} \\ \boldsymbol{\rho} \\ \boldsymbol{c}_0 \\ \vdots \\ \boldsymbol{c}_{c-1} \\ \dot{\boldsymbol{r}} \\ \boldsymbol{\omega} \\ \dot{\boldsymbol{c}}_0 \\ \vdots \\ \dot{\boldsymbol{c}}_{c-1} \end{bmatrix}, \qquad \boldsymbol{u}_k = \begin{bmatrix} \ddot{\boldsymbol{r}} \\ \dot{\boldsymbol{\omega}} \\ \ddot{\boldsymbol{c}}_0 \\ \vdots \\ \ddot{\boldsymbol{c}}_{c-1} \\ \boldsymbol{f}_0 \\ \vdots \\ \boldsymbol{f}_{c-1} \end{bmatrix}$$

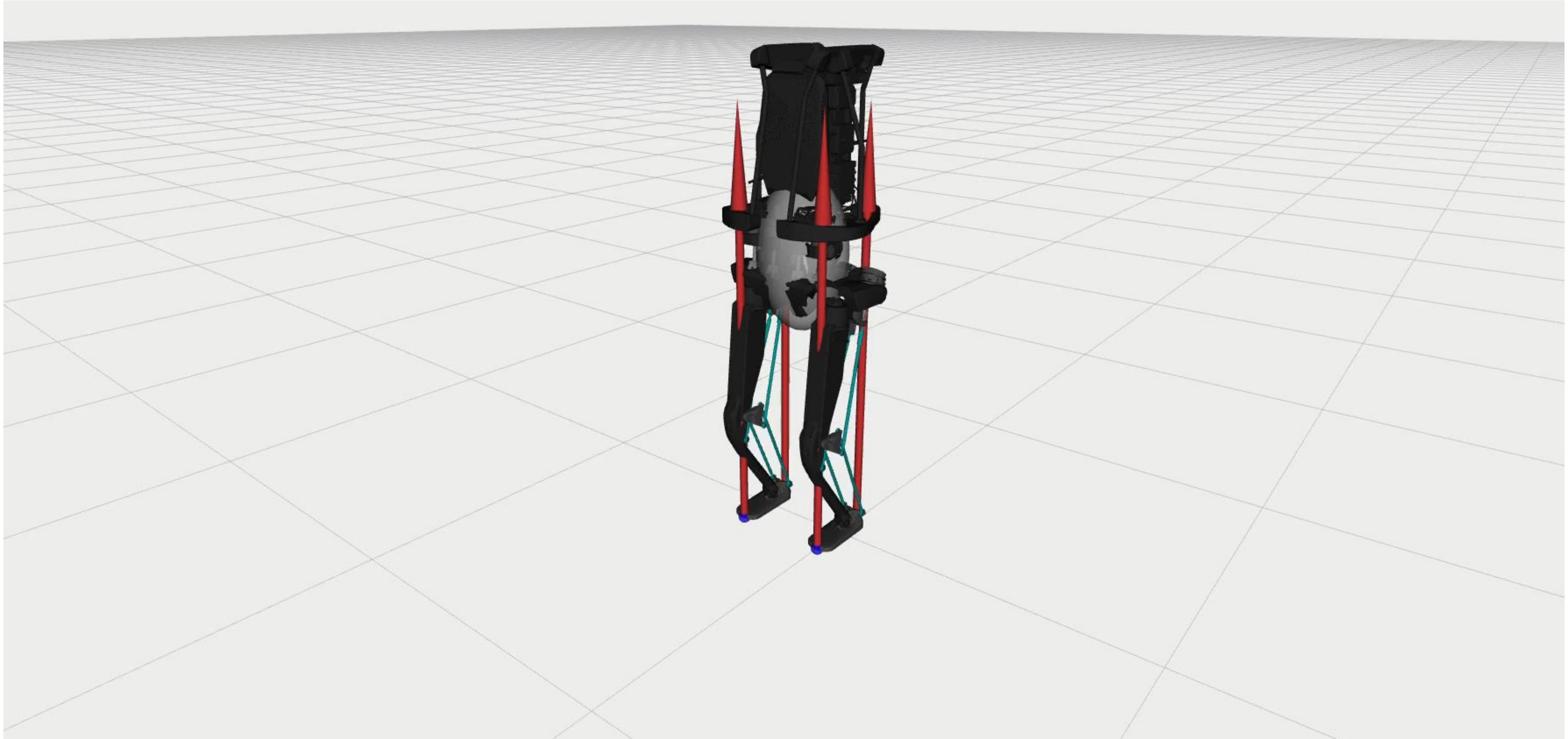
- We optimize over the SRBD state and contacts
- We keep all the non-linearities
- Explicit contact scheduling
- Receding horizon
- Inverse Dynamics approach

# MPC




- Tracking of linear CoM and base angular velocities
- 20 nodes
- T = 1 second
- 2 steps ahead
- ipopt (ma27), ~0.035 sec with line feet, at least 5 iterations
- SQP (OSQP) with Gauss-Newton approximation, ~0.025 sec with line feet, 1 iteration is fine

# MPC ► WBC (IK, FULL-MODEL)

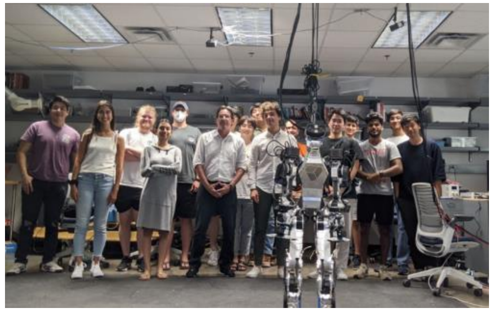


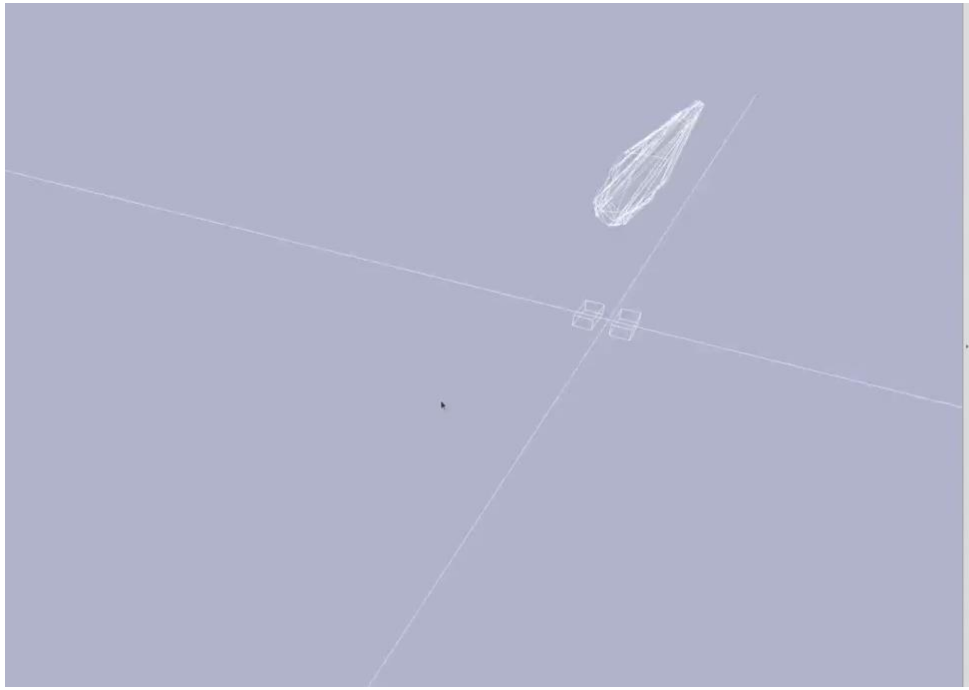
# MPC ▶ WBC

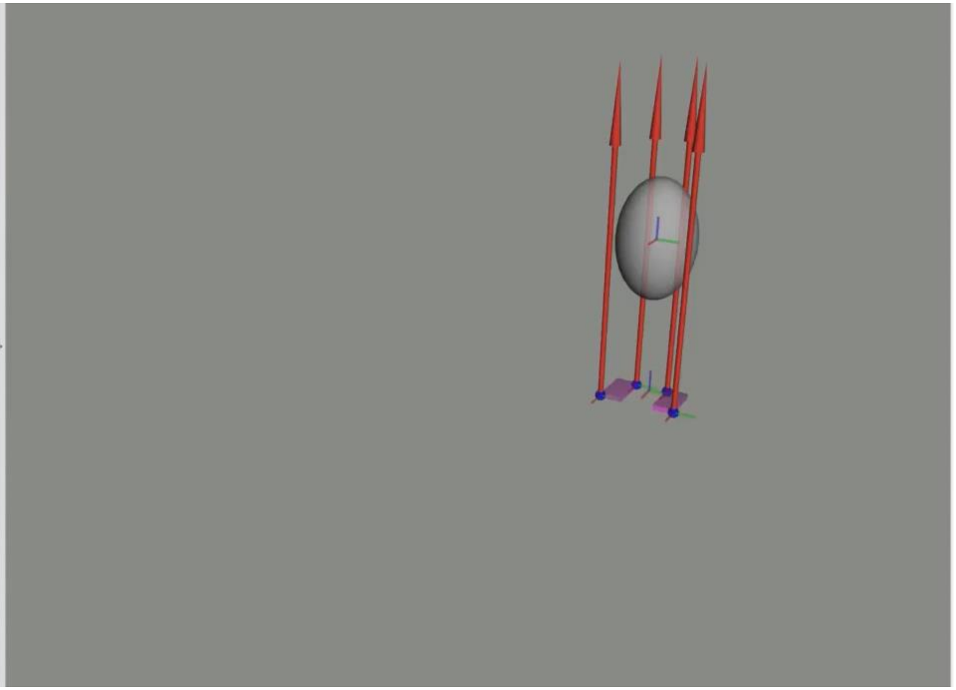
The University of Texas at Austin
Aerospace Engineering and Engineering Mechanics
Cockrell School of Engineering

**Luca Rossini, PhD Student (IIT)**

Visiting student at the Human Centered Robotics Laboratory, The University of Texas at Austin, lead by Prof. Luis Sentis.







# Conclusion

- Planning and Control pipeline based on Optimal Control techniques
- **TO:** based on simple model of Kangaroo permits to generate a great variety of motions
- **MPC:**
  - **Linearized SRBD:** faster but approximations penalizes angular part of the dynamics
  - **Full SRBD:** slower but usable as Reference Governor or fully closed loop
- Experiments on Kangaroo coming soon!

# Thank you

**Enrico Mingo Hoffman**
enrico.mingo@pal-robotics.com

**pal-robotics.com**

Humanoids 2022