Modular Passive Tracking for stack of tasks applying on the WBC of Humanoid robot

Students: Alessandro Angelo Anzellini - 2031062
Elisa Martinelli - 1853982
Daniele Teni - 2115180
An Nguyen - 2113021

Course: Underactuated robotics
Supervisor: Dr. Scianca
Professor: Prof. Leonardo Lanari

Table of contents

Introduction

- MPTC employs a passivity-based strategy to regulate multi-objective tasks, ensuring system stability even in overdetermined and conflicting scenarios.
- Combine MPTC with QP optimization under dynamic constraints to solve the Whole-body controller of humanoid
- Setup experiments to validate the proposed method

Task Space Robot Dynamic

- Robot Dynamics in the joint space:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + \tau_g(q) = S(\tau_j + \tau_{int}) + L_{all}^{\top} w_{all}$$

- The right hand side components:
  - S is the selection matrix
  - τ_j, τ_{int} is the actuated and disturbance torque
  - L_{all} is the stack of Jacobian of the links under the application of the stack of wrenches w_{all}

Task Space Robot Dynamic

- Task space velocity and acceleration:

$$\dot{x}_k = J_k\dot{q}, \quad \ddot{x}_k = J_k\ddot{q} + \dot{J}_k\dot{q}.$$

- Substitute $\ddot{q}$ from the generalized dynamics:

$$\ddot{x}_k = (\dot{J}_k - J_k M^{-1} C)\dot{q} + J_k M^{-1}(\tau - \tau_g) = Q_k\dot{q} + J_k M^{-1}(\tau - \tau_g)$$

- Denotes task space velocity and acceleration errors:

$$\dot{\tilde{x}}_k = \dot{x}_{k,ref} - \dot{x}_k, \qquad \ddot{\tilde{x}}_k = \ddot{x}_{k,ref} - \ddot{x}_k$$

S

Task Space Robot Dynamic

- Task space inertia and Coriolis matrix:

$$M_k = \left(J_k M^{-1} J_k^T\right)^{-1}, \quad C_k = M_k Q_k T_k^T$$

- The mapping from the generalize torque to task force:

$$T_k = M_k J_k M^{-1}$$

- All these components are needed to develop the controller

MPTC - for one task

- Lyapunov Energy function (for one task):

$$V_k = \frac{1}{2}\dot{\tilde{x}}_k^\top M_k \dot{\tilde{x}}_k + \frac{1}{2}\tilde{x}_k^\top K_k \tilde{x}_k$$

- Derivative of Lyapunov function:

$$\begin{aligned}\dot{V}_k &= \dot{\tilde{x}}_k^\top \left(M_k \ddot{\tilde{x}}_k + \frac{\dot{M}_k}{2}\dot{\tilde{x}}_k + K_k \tilde{x}_k \right) \\ &= \dot{\tilde{x}}_k^\top \left(T_k(\tau_g - \tau) + M_k Q_k \dot{q} + M_k \ddot{x}_{k,\mathrm{ref}} + C_k \dot{\tilde{x}}_k + K_k \tilde{x}_k \right)\end{aligned}$$

- Choose the desired task force to cancel out terms in bracket of $\dot{V}_k$

$$f_{k,\mathrm{des}} = T_k \tau_g + M_k Q_k \dot{q} + M_k \ddot{x}_{k,\mathrm{ref}} + (C_k + D_k)\dot{\tilde{x}}_k + K_k \tilde{x}_k$$

MPTC - for one task

- Define the task force error:

$$\tilde{f}_k = f_{k,des} - f_k \implies f_k = f_{k,des} - \tilde{f}_k = T_k\tau$$

- Substitute $T_k\tau$ with the expression of $f_{k,des}$ above in $\dot{V}_k$

$$\dot{V}_k = -\dot{\tilde{x}}_k^T D_k \dot{\tilde{x}}_k + \dot{\tilde{x}}_k^T \tilde{f}_k = \dot{V}_{k,des} + \dot{\tilde{V}}_k \leq \dot{\tilde{x}}_k^T \tilde{f}_k$$

- The task is passive w.r.t. $\tilde{f}_k$ (force error) and $\dot{\tilde{x}}_k$ (velocity error)

- Render the task space error dynamics:

$$M_k\ddot{\tilde{x}}_k + (C_k + D_k)\dot{\tilde{x}}_k + K_k\tilde{x}_k = \tilde{f}_k$$

MPTC - stack of tasks

- Extend the single-task formulation to manage multiple tasks in parallel using stacked vectors and optimization
- Desired task forces (stacked): $f_{\mathrm{des}} = [f_{1,\mathrm{des}}^\top, f_{2,\mathrm{des}}^\top, \ldots, f_{n_T,\mathrm{des}}^\top]^\top$
- Optimized command generation: $f_{\mathrm{cmd}} = TU u_{\mathrm{cmd}}$
- Command error: $\tilde{f}_{\mathrm{cmd}} = f_{\mathrm{des}} - f_{\mathrm{cmd}}$
- Quadratic cost function: minimizing tracking error

$$G = \frac{1}{2}\tilde{f}_{\mathrm{cmd}}^\top W \tilde{f}_{\mathrm{cmd}}$$

MPTC - Applying on humanoid robot

- Actuation mapping matrix

$$U = \begin{bmatrix} S & L_{EE}^T \end{bmatrix} = \begin{bmatrix} S & \Gamma_l . J_{lfoot}^T & \Gamma_r . J_{rfoot}^T \end{bmatrix}$$

- Γ_l, Γ_r capture the contact status (1: contact, 0: swing) of the left and the right foot.
- The optimization variables:

$$u_{cmd} = \begin{bmatrix} \tau_{j,cmd}^T & w_{lfoot}^T & w_{rfoot}^T \end{bmatrix}$$

- Subject to unilateral and friction cone constraints

Implementation and Simulation

- The proposed approach is realized using DART to control the Kawada HRP-4 robot.
- It includes three scenarios: normal walking, walking under an external push, and walking on uneven terrain.

Implementation and Simulation

The proposed approach employs an QP solver running at 100 Hz to compute joint torques for the defined tasks below:

- Two 6-dimensional tasks to control the position and orientation of each foot from the given footstep planner
- A 3-dimensional task to track the desired CoM position from ISMPC
- Two 3-dimensional tasks to regulate the desired orientation of the torso and the base
- A 10-dimensional task to handle the redundancy of the ten extra joints

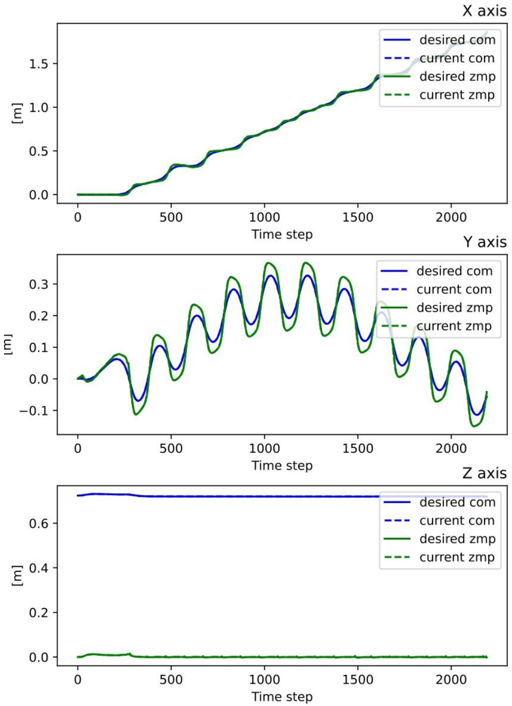
Detail of implementation (DRAFT ?)

- By studying the behavior of the simulation, it was observed that multiplying the matrices K_k , D_k by the task-space inertia matrix M_k, instead of using diagonal matrices, results in improved performance and reduces the effort required for fine-tuning.

- The following approach also uses the Pinocchio framework to compute the Coriolis matrix.

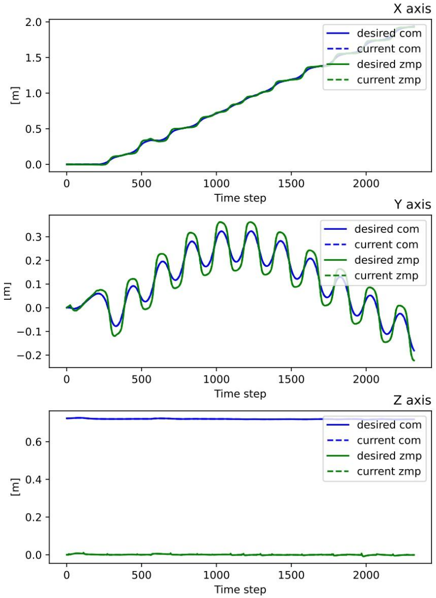
- The QP problem is solved with the linear solver osqp in CasADi

Normal Walk



The robot walks on flat terrain following trajectories generated by a high-level planner based on velocity commands applied to a virtual unicycle model.
By modulating these inputs, the robot successfully tracks a variety of feasible paths produced by the planner.

Push applied on the foot



In this Scenario , the robot is subjected to an external force of −4 N along the x axis, applied to the left foot during the swing phase, lasting 0.30 s.

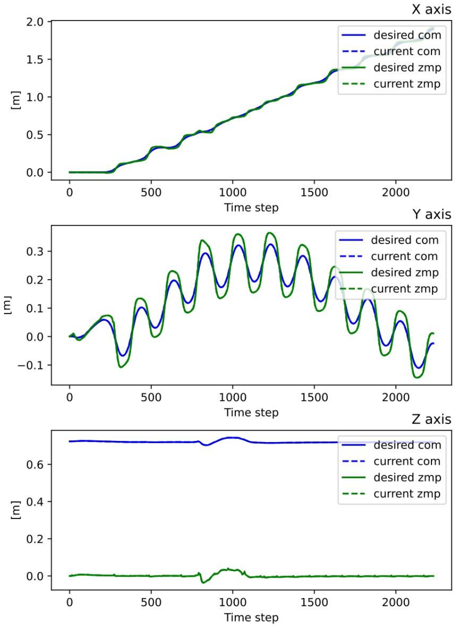
With the proposed control implementation, the robot maintains stable locomotion despite the disturbance.

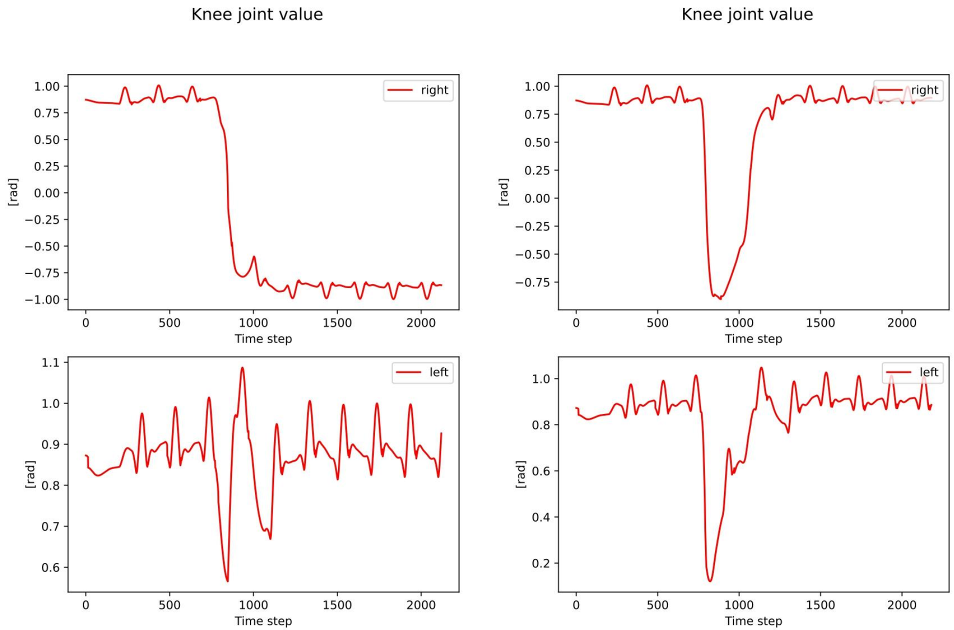
Push applied on the Torso/base

- The robot can withstand forces applied to the torso or base up to 10 N along both the x and y axes simultaneously.
- When the robot is pushed with a greater force, the knee may switch to an alternative configuration.
- To correct this posture and restore normal walking behavior, two one-dimensional tasks were added for the knee joint.

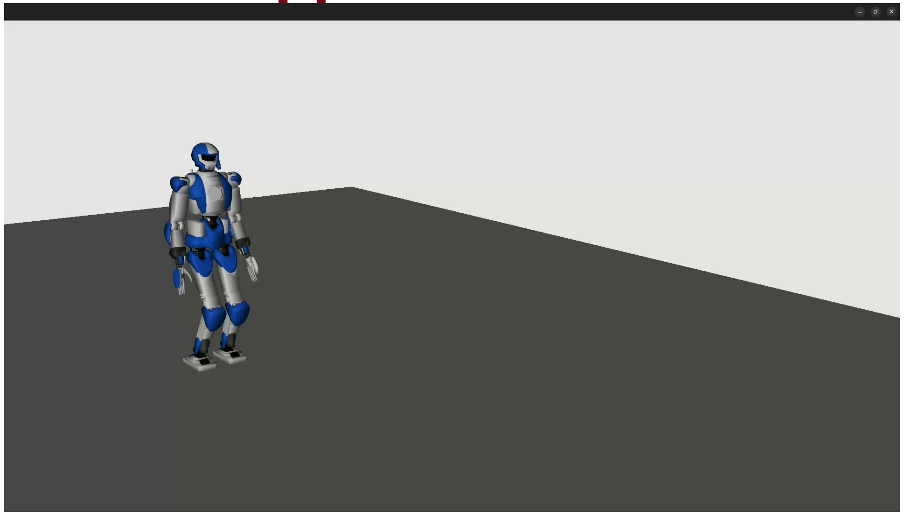
Push applied on the Torso/base

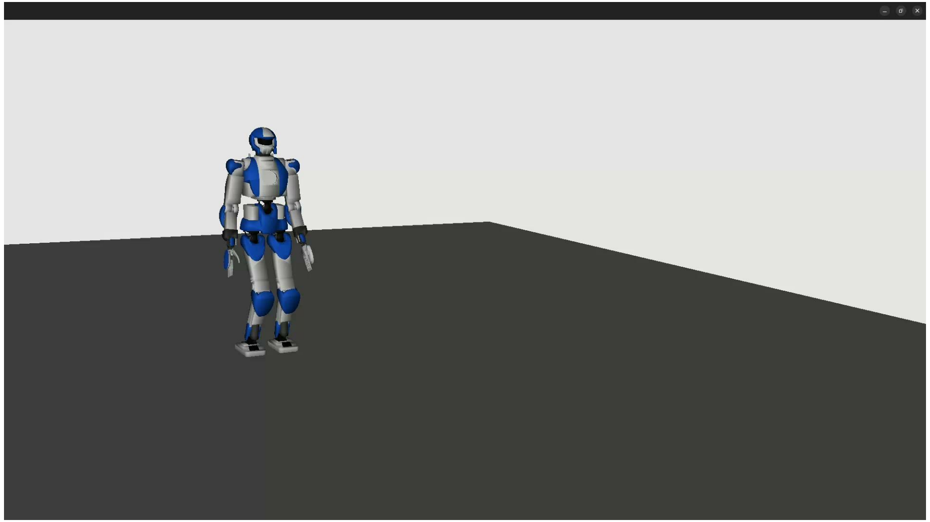
without additional task vs with additional task





Push applied on the Torso/base

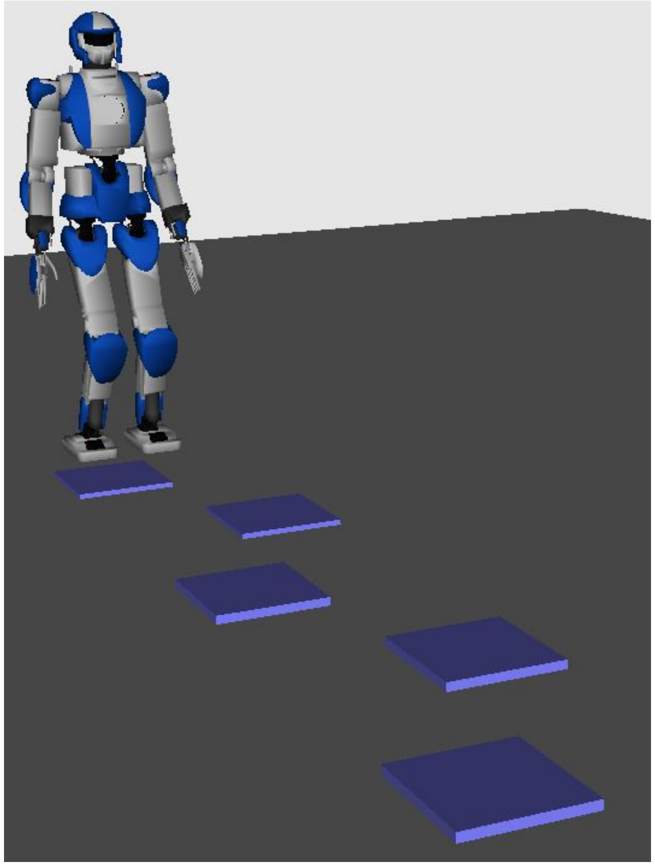




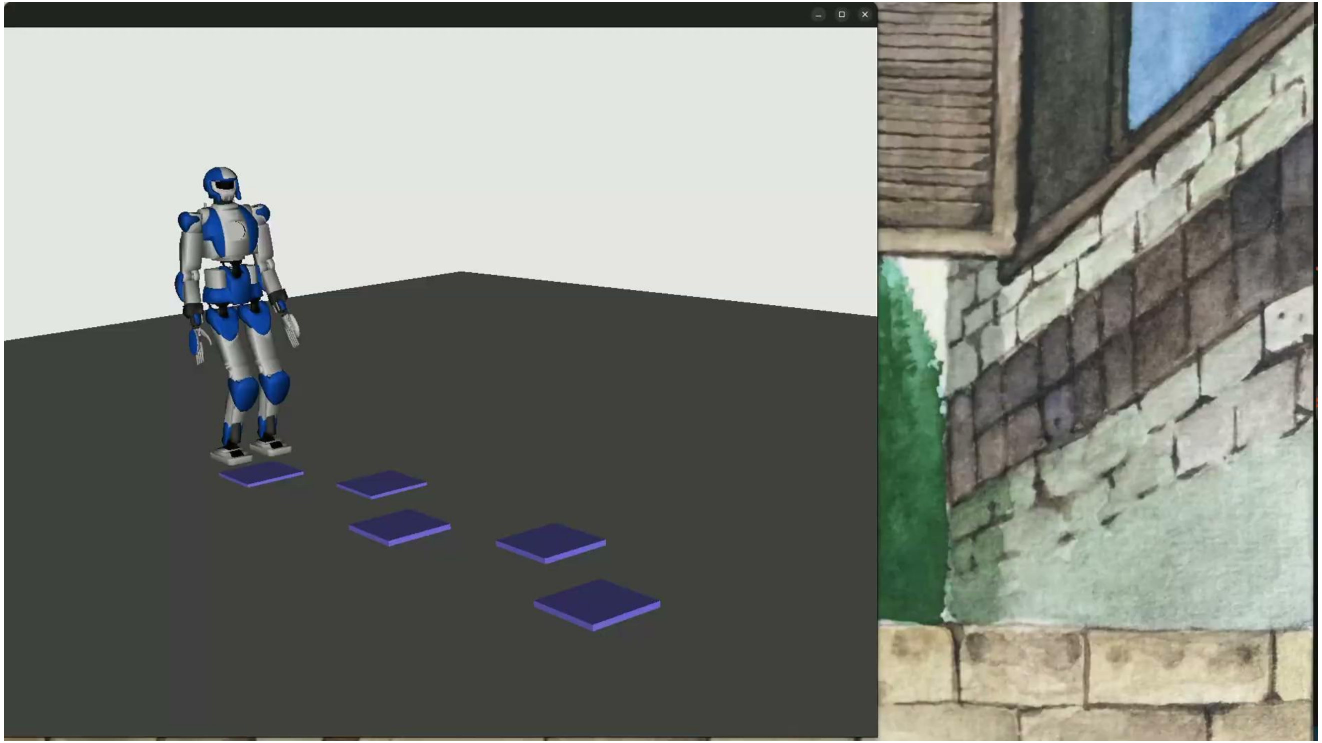
with additional task vs without additional task

Non flat environment

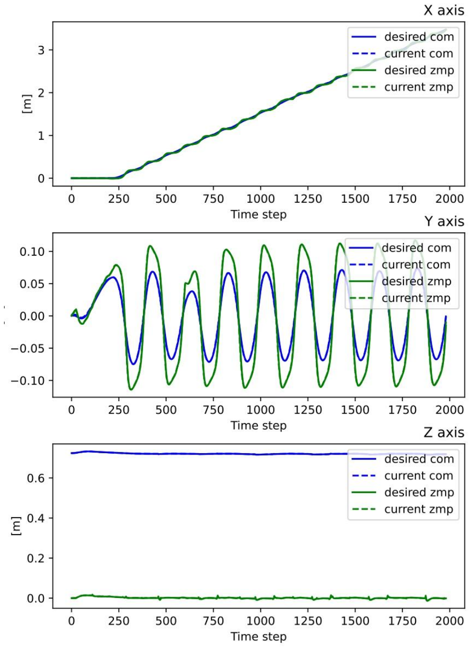
- The robot was tested on uneven terrain, where obstacles up to 2 cm in height
- The robot successfully executed the trajectory
- Due to the significant disturbances introduced by the terrain, the gains of the task dynamics had to be finely tuned

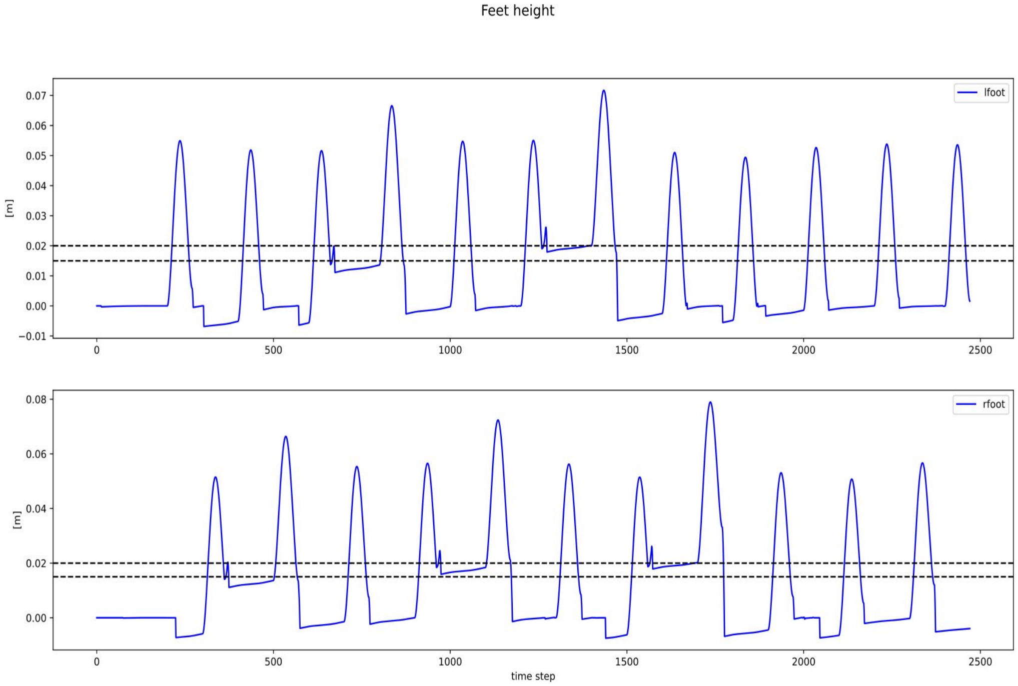


Non flat environment



Non flat environment





Comparison with Baseline

- The proposed approach proves to be **more robust** than the baseline inverse dynamics controller, managing to withstand nearly twice the external force.
- Except for the final experiment on uneven terrain, all other scenarios required **low effort of fine-tuning** of the control gains.
- **The computational performance** is **equivalent across both models**.
- Failures during simulation (especially uneven terrain) are primarily due to the gait generation MPC. Employing a different gait generation strategy could further improve the robot’s robustness.

Conclusion

- Successfully set up the simulation to validate the **robustness** of this framework with disturbances of **external pushes** and **uneven terrain**
- Adding task correcting knees' configuration to maintain proper walking pose

Concerns:

- Contact status variables do not capture properly the contact events, especially in the case of uneven terrain
- Failures during simulation due to the gait generation MPC is still an open question

Thank you for the attention!

Reference

[1] Johannes Englsberger, Alexander Dietrich, George-Adrian Mesesan, Gianluca Garofalo, Christian Ott, and Alin Olimpiu Albu-Sch¨affer. Mptc - modular passive tracking controller for stack of tasks based control frameworks. In 16th Robotics: Science and Systems, RSS 2020, 2020.

[2] N. Scianca, D. De Simone, L. Lanari, and G. Oriolo. Mpc for humanoid gait generation: Stability and feasibility. In 16th Robotics: Science and Systems, RSS, 2020.

[3] N. Scianca, D. De Simone, L. Lanari, and G. Oriolo. Ispc. In ismpc, 2020.

[4] Jeongseok Lee, Michael X. Grey, Sehoon Ha, Tobias Kunz, Sumit Jain, Yuting Ye, Siddhartha S. Srinivasa, Mike Stilman, and C. Karen Liu. Dart: Dynamic animation and robotics toolkit. Journal of Open Source Software, 3(22):500, 2018.

[5] Joel A E Andersson, Joris Gillis, Greg Horn, James B Rawlings, and Moritz Diehl. CasADi – A software framework for nonlinear optimization and optimal control. Mathematical Programming Computation, 11(1):1–36, 2019.

[6] J. Carpentier, G. Saurel, G. Buondonno, J. Mirabel, F. Lamiraux, O. Stasse, and N. Mansard. The Pinocchio C++ library – A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives. In International Symposium on System Integration (SII), 2019.

[7] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo. Robotics: Modelling, Planning and Control. Springer Publishing Company, Incorporated, 1st edition, 2008.