SAPIENZA
UNIVERSITÀ DI ROMA

DEPARTMENT OF COMPUTER CONTROL AND MANAGEMENT ENGINEERING

Trajectory Generation for Legged Robots Based on a Closed-Form Solution of Centroidal Dynamics

AUTONOMOUS AND MOBILE ROBOTICS

Professor:
Prof. Giuseppe Oriolo
Supervisor:
Nicola Scianca

Students:
Francesco Danese 1926188
Giuseppina Iannotti 1938436
Alessia Pontiggia 1892079

Academic Year 2024/2025

Contents

Abstract

Hello World !!

1 Introduction

2 Related Works

Understanding Centroidal Dynamics [**?**] is essential for controlling humanoid robots. Centroidal Dynamics [**?**] refers to the dynamics of a robot projected onto its Center of Mass (CoM), which represents the average position of all its mass and serves as the point where the overall gravitational force can be considered to act. Although humanoid robots are complex systems with high-dimensional, nonlinear dynamics, the motion of their Center of Mass (CoM) can often be described using simpler, more intuitive models. Focusing on the CoM allows the global dynamics of the robot to be captured without modeling each joint and link individually, significantly reducing system complexity. This approach provides an effective framework for planning, control, and stability, enabling the development of robots with robust, agile, and human-like behavior. Thanks to these advantages, many reduced order models have been developed by simplyfing CD in different ways

Thanks to these advantages, many reduced order models and control algorithms have been developed that incorporate the CoM and Centroidal Dynamics as key components. Among the earliest and most influential models is the Linear Inverted Pendulum Model (LIPM) [**?**], a point-mass approximation that simplifies the robot's dynamics by making two assumptions: (1) there is no angular momentum about the CoM, and (2) the CoM maintains a constant height. These assumptions allow the otherwise nonlinear centroidal dynamics to be linearized, greatly simplifying trajectory planning. Stability within this framework is maintained by ensuring that the Zero Moment Point (ZMP)—the point where the resultant ground reaction force would act—remains inside the support polygon formed by the robot's feet.

To extend the capabilities of LIPM, the Variable-Height Linear Inverted Pendulum Model (VH-LIPM) [**?**] was introduced. This model relaxes the constant height assumption, allowing vertical CoM motion, thereby enabling more dynamic and versatile movements such as stepping on uneven terrain or climbing.

Another important model is the Spring-Loaded Inverted Pendulum (SLIP) [**?**], which represents the CoM supported by a massless, compliant leg modeled as a spring. Unlike the rigid models, SLIP captures the energy storage and return mechanisms crucial for running, hopping, and other dynamic gaits. The ability of SLIP to express vertical oscillations of the CoM makes it essential for modeling and analyzing dynamic, non-walking locomotion.

These reduced order models have been extensively used in trajectory optimization approaches for motion planning. Trajectory optimization involves defining a motion task through state variables, control inputs, cost functions, and constraints, and then solving for the trajectory that optimizes performance while respecting dynamic feasibility. The reduced models serve as simplified system representations within these optimization problems. For instance, LIPM-based trajectory planners might define

the desired ZMP trajectory as a cost term or enforce ZMP constraints to guarantee stability. More advanced formulations integrate full centroidal dynamics, incorporating both CoM motion and base orientation, with stability constraints expressed through support polygons or centroidal wrench cones.

However, numerical trajectory optimization approaches often require fine time discretization—small time steps—to maintain solution accuracy, leading to a high number of decision variables. Moreover, feasibility is typically ensured only at discrete time nodes, and ensuring feasibility between these nodes remains a challenge.

Integration of Trajectory Optimization If a closed-form solution to the dynamic model is available, it enables trajectory generation using far fewer decision variables while ensuring feasibility at all times, not just at discrete points. Some methods have attempted to derive closed-form or analytical solutions for centroidal dynamics, often through simplifying assumptions such as constant CoM height or neglecting vertical-axis rotation. While these approaches are beneficial for efficiency, they limit the applicability of the resulting motions, particularly for more complex tasks like multi-contact planning or acrobatics. One technique for analytical integration involves assuming constant contact wrenches over finite time intervals (zero-order hold), but this introduces unwanted angular momentum variations unless extremely small time steps are used. To address this limitation, the multi-contact Linear Inverted Pendulum Model (mc-LIPM) was proposed, modeling contact forces through a stiffness-displacement relationship between the CoM and the contact points. This method allows for larger integration steps—potentially spanning entire contact phases—without significant angular momentum disturbances. Nevertheless, while mc-LIPM improves over traditional LIPM approaches, it still lacks full expressiveness for modeling complex rotational dynamics. Despite their success, these reduced order models introduce fundamental limitations. LIPM and VH-LIPM are built on strong assumptions that oversimplify the nonlinear nature of real-world dynamics, restricting their use in highly dynamic or irregular environments. SLIP preserves more physical realism but lacks a closed-form solution, complicating its analytical handling in optimization frameworks. Moreover, trajectory optimization methods based on these models often lead to large-scale optimization problems that are computationally expensive, challenging their application to real-time control.

Limitations These simplifications may be tolerable for basic locomotion tasks such as walking or slow stepping, but they become inadequate for advanced applications like multi-contact maneuvers, dynamic manipulation, or acrobatic motions, where high-fidelity and continuous-time feasibility are crucial. Additionally, the lack of closed-form solutions exacerbates the computational burden, making fast online planning difficult. To overcome these challenges, a novel stiffness-based method is introduced.

This approach provides a closed-form solution to the trajectory optimization problem by modeling the interaction forces through stiffness relationships. It enables the generation of long, dynamically feasible trajectories with far fewer decision variables, offering better computational efficiency and making real-time, complex motion generation more practical and reliable.

3 Proposed Method

3.1 Stiffness-Based Centroidal Dynamics

We begin from the standard *centroidal dynamics* equations, which relate the motion of the robot's center of mass (CoM) to the total external wrench (force and moment) acting on the system:

$$m\,\ddot{\mathbf{p}} = \mathbf{f} \;-\; m\,\mathbf{g}, \tag{1a}$$

$$\dot{\mathbf{L}} = \boldsymbol{\eta}. \tag{1b}$$

Here:

- $\mathbf{p} \in \mathbb{R}^3$ is the position of the CoM, and $\ddot{\mathbf{p}}$ its acceleration.
- $\mathbf{L} \in \mathbb{R}^3$ is the total angular momentum about the CoM.
- $\mathbf{f} \in \mathbb{R}^3$ and $\boldsymbol{\eta} \in \mathbb{R}^3$ are, respectively, the translational and rotational components of the total external wrench.
- $m > 0$ is the total mass, and $\mathbf{g} \in \mathbb{R}^3$ is the gravity vector (e.g. $\mathbf{g} = [0, 0, -9.81]^\top$).

We assume all external wrenches are contact wrenches at n_e end-effectors ("ends") of the robot. Denote by $\mathbf{p}_l \in \mathbb{R}^3$ the world-frame position of the l-th end, and let

$$\mathbf{f}_l \in \mathbb{R}^3, \quad \boldsymbol{\eta}_l \in \mathbb{R}^3, \qquad l = 1, \ldots, n_e$$

be the translational and rotational components of the contact wrench at that end. Then the total wrench is

$$\mathbf{f} \;=\; \sum_{l=1}^{n_e} \mathbf{f}_l, \qquad \boldsymbol{\eta} \;=\; \sum_{l=1}^{n_e} \Big[(\mathbf{p}_l - \mathbf{p}) \times \mathbf{f}_l \;+\; \boldsymbol{\eta}_l \Big]. \tag{2}$$

The cross-product in (2) makes the system *bilinear* in $(\mathbf{p}, \mathbf{f}_l)$, coupling CoM motion with contact forces.

Spring-like parametrization of contact wrenches. Rather than holding $\mathbf{f}_l, \boldsymbol{\eta}_l$ constant, we introduce a *stiffness* parameter $\lambda_l \geq 0$ and a *CMP-offset* vector $\mathbf{r}_l \in \mathbb{R}^3$ for each end, plus a pure-moment direction $\hat{\boldsymbol{\eta}}_l \in \mathbb{R}^3$. Inspired by a spring model, we set:

$$\boxed{\mathbf{f}_l = m\,\lambda_l^2 \Big(\mathbf{p} \;-\; (\mathbf{p}_l + \mathbf{r}_l) \Big), \qquad \boldsymbol{\eta}_l = m\,\lambda_l^2\,\hat{\boldsymbol{\eta}}_l.} \tag{3}$$

Intuitively:

- λ_l^2 scales like a contact stiffness: larger $\lambda_l \to$ stronger repulsive force.
- $\mathbf{p}_l + \mathbf{r}_l$ is the *virtual pivot* (similar to a Centroidal Moment Pivot, CMP). The force pulls the CoM toward that point.
- $\hat{\boldsymbol{\eta}}_l$ encodes any pure moment about the CoM, scaled consistently by λ_l^2.

Derivation of the stiffness-based model

By substituting (2) and (3) into (1a) and neglecting $O(\epsilon^2)$ terms one obtains

$$\begin{aligned} m\,\ddot{\mathbf{p}} &= \sum_{l=1}^{n_e} m\,\lambda_l^2\big(\mathbf{p} - (\mathbf{p}_l + \mathbf{r}_l)\big) \;-\; m\,\mathbf{g} \\ &\approx \sum_{l=1}^{n_e} m\,\lambda_l^2\big(\mathbf{p} - (\mathbf{p}_l + \mathbf{r}_l)\big) \;-\; m\,\mathbf{g} \;+\; m\,\epsilon^2\mathbf{p} \\ &= m\Big(\sum_l \lambda_l^2 + \epsilon^2\Big)\mathbf{p} \;-\; m\Big(\sum_l \lambda_l^2(\mathbf{p}_l + \mathbf{r}_l) + \mathbf{g}\Big) \\ &= m\,\bar{\lambda}^2\big(\mathbf{p} - \bar{\mathbf{p}} - \bar{\mathbf{r}}\big), \end{aligned} \tag{4}$$

where

$$\bar{\lambda}^2 = \sum_l \lambda_l^2 + \epsilon^2, \quad \bar{\mathbf{p}} = \frac{\sum_l \lambda_l^2\,\mathbf{p}_l + \mathbf{g}}{\bar{\lambda}^2}, \quad \bar{\mathbf{r}} = \frac{\sum_l \lambda_l^2\,\mathbf{r}_l}{\bar{\lambda}^2}.$$

Similarly, substituting into (1b) gives

$$\begin{aligned} \dot{\mathbf{L}} &= \sum_{l=1}^{n_e}\Big[(\mathbf{p}_l - \mathbf{p}) \times m\,\lambda_l^2(\mathbf{p} - \mathbf{p}_l - \mathbf{r}_l) + m\,\lambda_l^2\,\hat{\boldsymbol{\eta}}_l\Big] \\ &= \sum_l\big[(\mathbf{p} - \mathbf{p}_l) \times m\,\lambda_l^2\,\mathbf{r}_l\big] \;+\; \sum_l m\,\lambda_l^2\,\hat{\boldsymbol{\eta}}_l \\ &= \mathbf{p} \times m\,\bar{\lambda}^2\,\bar{\mathbf{r}} \;+\; \sum_l m\,\lambda_l^2\big(\hat{\boldsymbol{\eta}}_l - \mathbf{p}_l \times \mathbf{r}_l\big) \\ &\approx \big(m\,\ddot{\mathbf{p}} + m\,\bar{\lambda}^2(\bar{\mathbf{p}} + \bar{\mathbf{r}})\big) \times \bar{\mathbf{r}} \;+\; \sum_l m\,\lambda_l^2\big(\hat{\boldsymbol{\eta}}_l - \mathbf{p}_l \times \mathbf{r}_l\big) \\ &= m\big(\ddot{\mathbf{p}} \times \bar{\mathbf{r}} + \bar{\boldsymbol{\eta}}\big), \end{aligned} \tag{5}$$

where

$$\bar{\boldsymbol{\eta}} = \bar{\lambda}^2(\bar{\mathbf{p}} \times \bar{\mathbf{r}}) + \sum_l \lambda_l^2\big(\hat{\boldsymbol{\eta}}_l - \mathbf{p}_l \times \mathbf{r}_l\big).$$

Closed-form centroidal dynamics. Combining (4) and (5) yields the stiffness-based centroidal equations:

$$\ddot{\mathbf{p}} = \bar{\lambda}^2\big(\mathbf{p} - (\bar{\mathbf{p}} + \bar{\mathbf{r}})\big), \tag{6a}$$

$$\dot{\mathbf{L}} = m\big(\ddot{\mathbf{p}} \times \bar{\mathbf{r}} + \bar{\boldsymbol{\eta}}\big). \tag{6b}$$

The aggregated parameters $\bar{\lambda}, \bar{\mathbf{p}}, \bar{\mathbf{r}}, \bar{\boldsymbol{\eta}}$ recover the same expressions as before.

Discussion and special cases.

Remark 1 (Exactness vs. flight phase). If one ignores flight (i.e. always in contact, $\sum_l \lambda_l^2 > 0$), one may set $\epsilon = 0$ and (6) hold exactly. Otherwise $\epsilon > 0$ guarantees a well-defined $\bar{\lambda}$ in airborne phases.

Remark 2 (Ballistic motion). When all ends lose contact ($\lambda_l = 0$ for all l), one finds

$$\ddot{\mathbf{p}} = \epsilon^2\,\mathbf{p} - \mathbf{g} \approx -\,\mathbf{g}, \quad \dot{\mathbf{L}} = 0,$$

recovering the usual ballistic CoM motion and conservation of angular momentum.

Remark 3 (Relation to existing models). Stiffness-based (or force-to-point) parametrization has appeared before, but typically only at the *total* wrench level. Here we assign a separate $\lambda_l, \mathbf{r}_l, \hat{\boldsymbol{\eta}}_l$ to each end, which yields a unified multi-contact description. The classical CoP and (e)CMP emerge naturally as $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}} + \bar{\mathbf{r}}$, respectively.

3.2 Closed-Form Solutions and Discrete-Time Equations

We subdivide the time horizon $[0, T]$ into N consecutive intervals

$$[t_k,\, t_{k+1}], \quad k = 0, 1, \ldots, N-1, \qquad t_{k+1} = t_k + \tau_k.$$

We assume that *contact states* (i.e. which ends are in contact) change only at the boundaries t_k. Moreover, we apply a *zero-order hold* on the stiffness-based parameters

$$\{\lambda_l(t),\, \mathbf{r}_l(t),\, \hat{\boldsymbol{\eta}}_l(t)\} \quad \mapsto \quad \{\lambda_{l,k},\, \mathbf{r}_{l,k},\, \hat{\boldsymbol{\eta}}_{l,k}\} \quad \text{for } t \in [t_k, t_{k+1}),$$

meaning that each parameter is held constant over the interval.

State at the beginning of interval k. Let

$$\mathbf{p}_k = \mathbf{p}(t_k), \quad \mathbf{v}_k = \dot{\mathbf{p}}(t_k), \quad \mathbf{L}_k = \mathbf{L}(t_k),$$

and compute the aggregated quantities

$$\bar{\lambda}_k = \sqrt{\sum_{l=1}^{n_e} \lambda_{l,k}^2 + \epsilon^2}, \quad \bar{\mathbf{p}}_k = \frac{\sum_{l=1}^{n_e} \lambda_{l,k}^2\, \mathbf{p}_{l,k} + \mathbf{g}}{\bar{\lambda}_k^2}, \quad \bar{\mathbf{r}}_k = \frac{\sum_{l=1}^{n_e} \lambda_{l,k}^2\, \mathbf{r}_{l,k}}{\bar{\lambda}_k^2},$$

$$\bar{\boldsymbol{\eta}}_k = \bar{\lambda}_k^2\,(\bar{\mathbf{p}}_k \times \bar{\mathbf{r}}_k) \;+\; \sum_{l=1}^{n_e} \lambda_{l,k}^2 \big(\hat{\boldsymbol{\eta}}_{l,k} - \mathbf{p}_{l,k} \times \mathbf{r}_{l,k}\big).$$

Analytical solution on $[t_k, t_{k+1}]$. With $\bar{\lambda}_k, \bar{\mathbf{p}}_k, \bar{\mathbf{r}}_k, \bar{\boldsymbol{\eta}}_k$ constant, the CoM-dynamics

$$\ddot{\mathbf{p}} = \bar{\lambda}_k^2 \big(\mathbf{p} - (\bar{\mathbf{p}}_k + \bar{\mathbf{r}}_k)\big)$$

is a linear second-order ODE whose homogeneous+particular solution reads

$$\mathbf{p}(t) = (\bar{\mathbf{p}}_k + \bar{\mathbf{r}}_k) \;+\; C_k(\Delta t)\,\big(\mathbf{p}_k - (\bar{\mathbf{p}}_k + \bar{\mathbf{r}}_k)\big) \;+\; \frac{S_k(\Delta t)}{\bar{\lambda}_k}\,\mathbf{v}_k, \tag{7a}$$

$$\mathbf{v}(t) = \dot{\mathbf{p}}(t) = \bar{\lambda}_k\, S_k(\Delta t)\,\big(\mathbf{p}_k - (\bar{\mathbf{p}}_k + \bar{\mathbf{r}}_k)\big) \;+\; C_k(\Delta t)\,\mathbf{v}_k, \tag{7b}$$

where $\Delta t = t - t_k$ and

$$C_k(\Delta t) = \cosh\big(\bar{\lambda}_k\,\Delta t\big), \qquad S_k(\Delta t) = \sinh\big(\bar{\lambda}_k\,\Delta t\big).$$

Finally, substituting into the angular-momentum equation

$$\dot{\mathbf{L}} = m\big(\ddot{\mathbf{p}} \times \bar{\mathbf{r}}_k + \bar{\boldsymbol{\eta}}_k\big)$$

and integrating from t_k to t gives

$$\mathbf{L}(t) = \mathbf{L}_k \;+\; m\Big(\big(\mathbf{v}(t) - \mathbf{v}_k\big) \times \bar{\mathbf{r}}_k \;+\; (t - t_k)\,\bar{\boldsymbol{\eta}}_k\Big). \tag{7c}$$

Remark 4 (Zero-Order Hold). A zero-order hold means we approximate time-varying parameters by piecewise–constant values on each interval. This yields closed-form expressions above, at the cost of not capturing high-frequency parameter variations.

3.3 Integration of Base-Link Rotation

The centroidal state $(\mathbf{p}, \dot{\mathbf{p}}, \mathbf{L})$ does not specify the *orientation* $\mathbf{R}(t) \in SO(3)$ or the *base-link* angular velocity $\boldsymbol{\omega}(t) \in \mathbb{R}^3$. In a multi-body system one shows

$$\mathbf{L} \;=\; \underbrace{\mathbf{R}\,I\,\mathbf{R}^\top}_{I_{\text{sys}}(\mathbf{R})}\,\boldsymbol{\omega} \;+\; \mathbf{R}\,\hat{\mathbf{L}}, \tag{8}$$

where

- $I \in \mathbb{R}^{3\times 3}$ is the composite inertia in the base-link frame,
- $\hat{\mathbf{L}}$ is the angular momentum about the base link due to internal motions.

If we fix reference values I_{ref}, $\hat{\mathbf{L}}_{\text{ref}}$ (e.g. from a nominal whole-body motion), we solve for

$$\boldsymbol{\omega}(t) = \; I_{\text{sys}}(\mathbf{R})^{-1}\big(\mathbf{L} - \mathbf{R}\,\hat{\mathbf{L}}_{\text{ref}}\big) \approx \mathbf{R}\,I_{\text{ref}}^{-1}\big(\mathbf{R}^\top\mathbf{L} - \hat{\mathbf{L}}_{\text{ref}}\big). \tag{9}$$

Discrete quaternion update. Let $\mathbf{q}_k \in \mathbb{H}$ be the unit-quaternion representing $\mathbf{R}(t_k)$. Over $[t_k, t_{k+1}]$ we subdivide into n_{div} equal steps

$$t_k = t'_0 < \cdots < t'_i < \cdots < t'_{n_{\text{div}}} = t_{k+1}, \qquad \tau'_k = \frac{\tau_k}{n_{\text{div}}}, \quad t'_i = t_k + i\,\tau'_k.$$

At each substep we assume $\boldsymbol{\omega}$ nearly constant and update

$$\mathbf{q}_{k+1} = \underbrace{\mathbf{q}\big(\boldsymbol{\omega}(t'_{n_{\text{div}}})\,\tau'_k\big)}_{\text{quat. for small rotation}} \;\cdots\; \mathbf{q}\big(\boldsymbol{\omega}(t'_1)\,\tau'_k\big) \;\cdot\; \mathbf{q}\big(\boldsymbol{\omega}(t'_0)\,\tau'_k\big) \;\cdot\; \mathbf{q}_k, \tag{10}$$

where $\mathbf{q}(\boldsymbol{\theta})$ is the unit quaternion corresponding to the axis-angle $\boldsymbol{\theta} \in \mathbb{R}^3$. The designer chooses n_{div} to balance integration accuracy against computational cost of gradient evaluation in trajectory optimization.

4 Formulation of Trajectory Optimization Problem

4.1 State Equation

To mathematically describe the evolution of the system over discrete time intervals, we define at each time step k:

- a state vector x_k, containing all the information necessary to describe the system's configuration and motion;
- a control input vector u_k, based on the stiffness-based control strategy introduced in Section 3.

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{p}_k \\ \mathbf{q}_k \\ \mathbf{v}_k \\ \mathbf{L}_k \\ t_k \\ \{\mathbf{p}_{l,k}\}_{l=1,\dots,n_e} \\ \{\mathbf{q}_{l,k}\}_{l=1,\dots,n_e} \end{bmatrix} \tag{7}$$

$$u_k = \begin{bmatrix} \tau_k \\ \{\mathbf{v}_{l,k}\}_{l=1}^{n_e} \\ \{\boldsymbol{\omega}_{l,k}\}_{l=1}^{n_e} \\ \{\boldsymbol{\lambda}_{l,k}\}_{l=1}^{n_e} \\ \{\mathbf{r}_{l,k}\}_{l=1}^{n_e} \\ \{\boldsymbol{\eta}_{l,k}\}_{l=1}^{n_e} \end{bmatrix} \tag{8}$$

Each component of the state and input vectors has been introduced and detailed in Section 3.
This formulation enables explicit enforcement of physical constraints such as contact complementarity by treating velocities as control variables. For instance, assigning a high cost to velocities at inactive contacts discourages motion in those regions, effectively ensuring that contacts are only active when physically meaningful.

Equations Update The update to the timestamp is given by:

$$t_{k+1} = t_k + \tau_k \tag{9}$$

The state update for each end-effector's position and orientation is governed by the basic kinematic relations:

$$\mathbf{p}_{l,k+1} = \mathbf{p}_{l,k} + \mathbf{v}_{l,k}\tau_k \tag{10}$$

$$\mathbf{q}_{l,k+1} = q(\boldsymbol{\omega}_{l,k}, \tau_k) \cdot \mathbf{q}_k \tag{11}$$

Here, $q(\boldsymbol{\omega}_{l,k}, \tau_k)$ represents the quaternion corresponding to the angular displacement over the time interval τ_k, computed from the angular velocity $\boldsymbol{\omega}_{l,k}$. Integrating all these elements, the complete system dynamics are compactly expressed as a state transition function:

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k) \tag{12}$$

This function encapsulates both the kinematics and dynamics of the system, allowing for predictive simulation or optimization over multiple time steps.

4.2 Formulation of Optimal Control Problem

The cost function is a crucial component of the trajectory optimization problem, as it quantifies the performance of a given trajectory. The cost function is typically composed of several terms, each representing different aspects of the system's performance.

Task Related Cost In trajectory tracking tasks, it is important for the system to follow a planned or reference trajectory as closely as possible. The *task-related cost* measures how much the system's current state and inputs deviate from their desired (reference) values. Minimizing this cost ensures the system stays close to the intended path during motion. The task-related cost function is formulated as:

$$L_{\text{task},k} = \frac{1}{2}\|W_k^x(\mathbf{x}_k - \mathbf{x}_k^{\text{ref}})\|^2 + \frac{1}{2}\|W_k^u(\mathbf{u}_k - \mathbf{u}_k^{\text{ref}})\|^2 \tag{13}$$

where $(*)^{\text{ref}}$ represents the reference (target) values for the state x_k and the control input u_k. In this work, we focus on a waypoint-tracking task, where a series of intermediate waypoints are specified for the center of mass (CoM), base link, and limb endpoints. Desired positions and velocities along a smooth path connecting these waypoints are generated using spline curves. The system's stiffness parameters are computed by solving the following least-squares optimization problem at each time step k:

$$\min \left\| \sum_l \lambda_{l,k}^2 \right\|^2 \quad \text{subject to} \quad \sum_l \lambda_{l,k}^2 (\mathbf{p}_k^{\text{ref}} - \mathbf{p}_{l,k}^{\text{ref}}) = \mathbf{g} \tag{14}$$

This optimization distributes stiffness values to maintain support for the CoM against gravity. In addition, the reference values for the Centroidal Moment Pivot (CMP) offset and the moments at the end effectors are typically set to zero unless non-zero values are specifically chosen to induce desired dynamic effects. The weighting matrices W_k^x and W_k^u are design parameters that control the importance given to state and input deviations in the cost function. Finally, when dealing with rotational variables represented by quaternions, the deviation between the actual and reference orientations is calculated by transforming the quaternion difference into an equivalent angle-axis representation:

$$\mathbf{q} - \mathbf{q}^{\text{ref}} := \omega(\mathbf{q}^{\text{ref}^{-1}}\mathbf{q}) \tag{15}$$

where $\omega(\cdot)$ maps a unit quaternion into an angle-axis vector.

Inequality Constraints In physical systems involving contact dynamics, it is essential to ensure that certain physical conditions—such as feasible positions, contact forces, and proper stiffness—are always satisfied. These conditions are formulated as *inequality constraints* and play a key role in maintaining the physical realism of the motion and feasibility of the optimization problem. In this framework, inequality constraints are

used to regulate the position of each contact point relative to the CoM and the base link, enforce non-slipping conditions, and bound parameters like phase duration and stiffness values. First, the box constraint on the position of each end link relative to the CoM and the base link is expressed as:

$$\mathbf{p}_{\mathrm{l,min}} \leq \mathbf{q}^{-1}(\mathbf{p}_l - \mathbf{p}) \leq \mathbf{p}_{\mathrm{l,max}}, \tag{16}$$

where $\mathbf{p}$ is the CoM position, $\mathbf{p}_l$ is the position of the l-th end, and $\mathbf{q}^{-1}$ denotes the inverse transformation to the local frame. Simple range constraints are imposed on the duration of each phase and the stiffness values:

$$\tau_{\min} \leq \tau \leq \tau_{\max}, \tag{17}$$

$$0 \leq \lambda_l \leq \lambda_{\max}, \quad \forall l. \tag{18}$$

Next, for each end in contact, the contact wrench must satisfy non-slip and moment conditions. If we want to avoid relative motion of the contact surfaces, we must ensure sufficient friction. Thus, to achieve no slipping, the force must be within the friction cone, i.e., the tangential force f_t must satisfy

$$|f_t| \leq \mu f_n \Longrightarrow \sqrt{f_{l,x}^2 + f_{l,y}^2} \leq \mu f_{l,z} \tag{19}$$

where μ is the static friction coefficient, and $(f_{l,x}, f_{l,y}, f_{l,z})$ are the force components at the contact. Constraints on the moments at the contact point are given as:

$$-c_{\max,x} f_{l,z} \leq \eta_{l,x} \leq c_{\max,x} f_{l,z}, \tag{20}$$

$$c_{\min,y} f_{l,z} \leq \eta_{l,y} \leq c_{\max,y} f_{l,z}, \tag{21}$$

$$-\mu_z f_{l,z} \leq \eta_{l,z} \leq \mu_z f_{l,z}, \tag{22}$$

where $c_{\min}$ and $c_{\max}$ define the rectangular bounds of the center-of-pressure (CoP) region, and μ_z is the friction coefficient for the moment. All the inequality constraints can be compactly represented as:

$$g(\mathbf{x}_k, \mathbf{u}_k) \geq 0, \tag{23}$$

where $g(\cdot)$ is a differentiable vector-valued function, evaluated componentwise. To handle these constraints during optimization, a log-barrier function is introduced:

$$L_{\mathrm{limit}}(\mathbf{x}_k, \mathbf{u}_k) = \sum_{i=1}^{n_g} -\log \max(\epsilon, g_i(\mathbf{x}_k, \mathbf{u}_k)), \tag{24}$$

where n_g is the number of constraints, g_i is the i-th constraint function, and ϵ is a small positive constant that ensures numerical stability and prevents the log function from becoming undefined.

Contact Dependent Cost Modeling the interaction between a robot's ends and the environment is crucial in dynamic contact systems. The contact-dependent cost is introduced to enforce the complementarity between the contact forces, end velocities, and stiffness values. This ensures that if an end effector is in contact, its motion and interaction forces behave physically, and if it is not in contact, unnecessary forces or stiffness values are suppressed. By minimizing these costs, the system ensures physically meaningful transitions between contact and non-contact states throughout the trajectory.
The contact-dependent cost is defined as:

$$\begin{aligned} J_{\text{compl},k} = w_{\text{compl}}^2 \sum_l \Bigg(& \underbrace{\sum_i \delta\left[\sigma_{l,k} = i\right] \left(\boldsymbol{\eta}_i^\top (\mathbf{p}_{l,k} - \mathbf{o}_i)\right)^2}_{\text{contact distance constraint}} \\ & + \underbrace{\delta\left[\sigma_{l,k} \neq \emptyset\right] \left(\|\mathbf{v}_{l,k}\|^2 + \|\boldsymbol{\omega}_{l,k}\|^2\right)}_{\text{zero velocity constraint}} \\ & + \underbrace{\delta\left[\sigma_{l,k} = \emptyset\right] \lambda_{l,k}^2}_{\text{zero stiffness constraint}} \Bigg) \end{aligned} \tag{25}$$

where $\sigma_{l,k}$ denotes the contact state of the l-th end at time step k. In particular,

- $\sigma_{l,k} = i$ if the l-th end is in contact with the i-th contact surface.
- $\sigma_{l,k} = \emptyset$ if it is not in contact.

The operator $\delta[*]$ is an indicator function that returns 1 if the condition inside the brackets is true, and 0 otherwise.
Each term inside the cost function has a specific physical meaning:

- The first term enforces that if the l-th end is in contact with surface i, namely the distance in the normal direction (defined by normal vector $\boldsymbol{\eta}_i$) from the end's position $\mathbf{p}_{l,k}$ to the contact surface origin $\mathbf{o}_i$ must be zero.
- The second term requires that if the l-th end is in contact with any surface, its linear velocity $\mathbf{v}_{l,k}$ and angular velocity $\boldsymbol{\omega}_{l,k}$ must also be zero, representing static contact.
- The third term ensures that if the l-th end is not in contact, the associated stiffness $\lambda_{l,k}$ must be zero. Physically, this prevents generating unnecessary contact forces when there is no actual contact.

By properly tuning the weight parameter w_{compl}, these complementarity errors can be made acceptably small after optimization. A sufficiently large value of w_{compl} is necessary to enforce these physical consistency conditions without overly penalizing the overall optimization performance.

Final Cost Function and Problem Formulation After defining the task-related, limit-related, and contact-dependent costs, the overall cost function is constructed by summing these individual terms over all time steps. This aggregated cost captures all important objectives and physical constraints of the motion planning task.
The overall cost function is defined as:

$$J[\boldsymbol{\sigma}] = \sum_{k} \left[L_{\text{task},k} + L_{\text{limit},k} + L_{\text{compl},k}[\boldsymbol{\sigma}_k] \right] \tag{26}$$

The planning problem can then be formulated as the following optimal control problem:

$$\begin{aligned} &\text{find} \quad \mathbf{x}, \mathbf{u} \quad \text{that minimizes} \quad J[\boldsymbol{\sigma}](\mathbf{x}, \mathbf{u}) \\ &\text{subject to} \quad \mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k) \end{aligned} \tag{27}$$

Here, x and u denote the state and control input trajectories, respectively. The function $f(\mathbf{x}_k, \mathbf{u}_k)$ represents the discrete-time system dynamics, ensuring that the state evolution is dynamically feasible.

5 Simulation and Results

This section presents the results obtained using the proposed method. We begin by describing the reference trajectory used for each task. Subsequently, relevant plots, tables, and simulations are provided to illustrate system behavior and performance. Before introducing the implemented trajectories, we define the contact state notation:

- **0** indicates the foot is in contact with the ground.
- **-** indicates the foot is not in contact (i.e., lifted).

Note that the trajectories are described with respect to contact phases, not continuous time.

5.1 Trajectory Generation

Still Task

The objective of the still task is for the robot to remain stationary, maintaining its initial configuration throughout the entire duration of the trajectory. In this scenario, both feet stay in continuous contact with the ground, ensuring complete stability and zero locomotion. The contact sequence used is shown in Table **??**, where the first row represents the right foot and the second row the left foot.

Task	**N**	**Contact Sequence**
Still	4	0 0 0 0 0 0 0 0

Table 1: Contact sequence for still task

The reference trajectory for the "Still Task" is generated and updated in a step-by-step process. First, the system initializes two main components: the reference state vector, which represents the robot's position and motion, and the reference input vector, which contains the control inputs. The dimensions of these vectors are defined based on the number of phases in the task. Next, the initial values for the robot's Center of Mass (CoM), foot positions, and orientations are set in the reference state vector. At this point, time is set to zero

Algorithm 1 Reference Trajectory Initialization and Update for Still Task

Initialize $X_{\text{ref}} \in \mathbb{R}^{28\times(N+1)}$, $U_{\text{ref}} \in \mathbb{R}^{27\times N}$
Set initial CoM state, feet positions, and orientations in X_{ref}
time $\leftarrow 0$
for $t = 0$ to $N - 1$ **do**
    Read current contact state from σ
    Set phase duration τ and contact force gains λ
    Update U_{ref} with the above
    Set CoM and feet position, velocities, and orientation as the initial state values
    time $\leftarrow$ time+ phase duration
    Update $X_{\text{ref}}(t + 1)$ with the above
end for
return $X_{\text{ref}}, U_{\text{ref}}$

Walking Task

For the walking task, the trajectory is constructed based on the following principles: During double-support phases, where both feet are on the ground ([0, 0]), the Center of Mass (CoM) remains stationary. Movement along the X-axis occurs during single-support phases, when one foot is lifted ([-, 0] or [0, -]). If the preceding phase is double support, the CoM position remains unchanged; if the preceding phase is single support, the CoM advances by a specified displacement. The Z-coordinate (height) of each foot is zero at the beginning of each phase, as the feet are in contact with the ground. The X-coordinate increases alternately, depending on which foot was lifted in the previous phase. The foot in motion follows a trajectory that surpasses the stationary foot, simulating a natural walking pattern. Each foot's intra-phase height position is interpolated using a parabolic profile, starting and ending at the positions determined by the phase solutions, rather than employing fixed inputs (zero-hold). The contact sequence used is presented in Table 2, with the first row for the right foot and the second for the left foot.

Table 2: Contact sequence for walking task

Task	**N**	**Right Foot**	**Left Foot**
Walk	24	0 0 0 - 0 0 0 - 0 0 0 - 0 0 0 - 0 0 0 - 0	0 - 0 0 0 - 0 0 0 - 0 0 0 - 0 0 0 - 0 0 0

It starts by setting up matrices for the reference trajectory states (X_ref for positions and U_ref for control inputs) and defining the initial conditions for the center of mass (CoM), foot positions, and orientations. Then, for each step (from 0 to N-1), the algorithm reads the contact states (which indicate whether the foot is in contact with the ground), sets the phase duration (the length of time each walking phase lasts),

and adjusts the control parameters accordingly.

Algorithm 2 Reference Trajectory Initialization and Update for Walking Task

Initialize $X_{\text{ref}} \in \mathbb{R}^{28\times(N+1)}$, $U_{\text{ref}} \in \mathbb{R}^{27\times N}$
Set initial CoM state, feet positions, and orientations in X_{ref}
time $\leftarrow 0$, sig_idx $\leftarrow 0$
for $t = 0$ to $N - 1$ **do**
    Read contact states from σ
    Set phase duration and contact force gains λ
    Update U_{ref} with phase duration and λ
    sig_idx $\leftarrow$ sig_idx $+ 2$
end for
for $t = 1$ to N **do**
    Read current and previous contact states
    Set CoM velocity (v_x, v_y) and feet step velocities
    Update CoM and feet positions based on velocity and duration
    Update time and orientations in X_{ref}
    sig_idx $\leftarrow$ sig_idx $+ 2$
end for
return $X_{\text{ref}}, U_{\text{ref}}$

5.2 Implementation Details

In this section, we provide a summary of the key parameters and choices made in the implementation, particularly distinguishing between the still and walking scenarios. These parameters govern the state initialization, the dynamics, and the optimization settings for the robot's movement.

The solution for both tasks is computed with ipopt solver from Casadi optimizer. Both the reference and solution are contact-dependent, meaning that X and U contain values for each contact phase, thus to find the time dependent solution we used the dynamics.

Still Task

For the still scenario, the robot begins with a fixed initial state, with the Center of Mass (CoM) positioned at a specific point, neutral orientations for the feet, and zero velocities. The system is initialized with parameters that ensure no significant movement, providing a static reference for comparison with dynamic cases.

The key parameters for the still scenario include:

In this setup, the robot is essentially in a static position, with minimal movement.

Parameter	Value
Initial CoM Position	$[-6.1787 \times 10^{-4}, 4.433 \times 10^{-4}, 7.2398 \times 10^{-1}]$
Initial Feet Positions	$\begin{bmatrix} 1.0311 \times 10^{-17}, -1.0164 \times 10^{-1}, -1.3878 \times 10^{-17} \\ 1.0311 \times 10^{-17}, 1.0164 \times 10^{-1}, -1.3878 \times 10^{-17} \end{bmatrix}$
Initial Feet Orientations	$\begin{bmatrix} 1, 0, 0, 0 \\ 1, 0, 0, 0 \end{bmatrix}$
Initial Velocities	$[0, 0, 0]$

Table 3: Still Scenario Initial Parameters

Walking Task

In contrast, for the walking scenario, the robot's state is dynamic, with the CoM moving in response to lifting one foot while the other remains grounded. The robot's motion is controlled by optimizing the contact forces and foot movements according to the phases of walking. Here, the CoM velocity is initialized in the negative y-direction to simulate walking forward.

The key parameters for the walking scenario include:

Parameter	Value
Initial CoM Velocity	$[0, -0.07, 0]$
Feet Length	0.1 meters
Friction Coefficient (μ)	0.5
Friction Coefficient in Z (μ_z)	0.6
Gravity Factor (g)	$[0, 0, 9.81]$ m/s^2

Table 4: Walking Scenario Parameters

In the walking scenario, the optimization process adjusts the robot's foot positions and orientations, ensuring that the forces exerted during the gait cycle are balanced, with the robot transitioning between phases of double and single support. This approach is essential for dynamic stability during motion.

State Component	**Weight**
Position (p_k)	1
Orientation (q_k)	500
Velocity (v_k)	500
Angular Momentum (L_k)	1
Time (t_k)	1
Feet Positions (P_{L_k})	100
Feet Orientations (Q_{L_k})	0.0001

Table 5: State Weight Matrix (W_{x_k})

Input Component	**Weight**
Phase Duration (τ_k)	1
Linear Velocities (V_{L_k})	1
Angular Velocities (W_{L_k})	1
Contact Force Gains (λ_{L_k})	1
Feet Orientations (R_{L_k})	1
Contact States (η_{L_k})	1

Table 6: Input Weight Matrix (W_{u_k})

Cost and Weighting Matrices

The weight matrices ensure that the optimization algorithm focuses on controlling the position, velocity, and foot placement, while maintaining the robot's stability during movement. By adjusting the weights, the optimization can balance the various objectives, such as minimizing the deviation from the reference trajectory and maintaining the robot's stability in both still and walking scenarios.

Optimization Parameters

Finally, optimization parameters such as the complementarity weight (w_{compl}), the friction coefficients (μ and μ_z), and the maximum and minimum values for the joint torque ($\tau_{\min}$ and $\tau_{\max}$) are set. These values ensure that the optimization is constrained by the physical limits of the robot's actuators and the forces exerted by the environment.

Parameter	**Value**
Complementarity Weight (w_{compl})	1000
Friction Coefficient (μ)	0.5
Friction Coefficient in Z (μ_z)	0.6
Maximum Torque ($\tau_{\max}$)	10
Minimum Torque ($\tau_{\min}$)	0.4

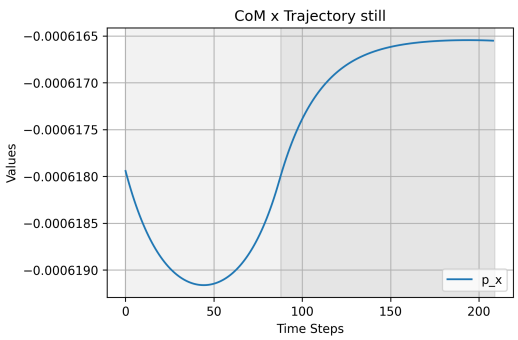
Table 7: Optimization Parameters

5.3 Still Task: results

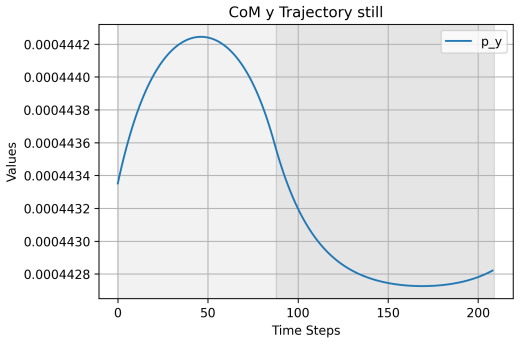
In this case the state vector has dimension 3, and the input vector has dimension 2.

CoM Plots

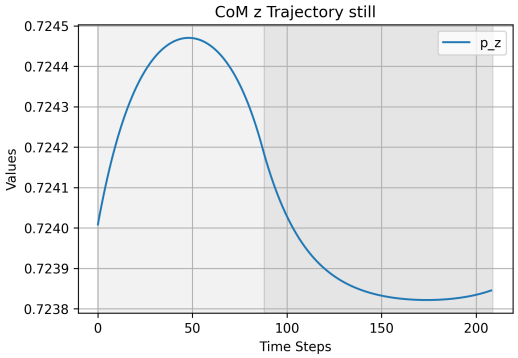
The following plots report the CoM trajectory in time. The alternation between light and dark grey on the background suggests the switch between phases. As observed, there is negligible motion in any direction, confirming the robot's stationary state.



(a) Com X Trajectory 1



(b) Com Y Trajectory 1



(c) Com Z Trajectory 1

Figure 1: Com Trajectory

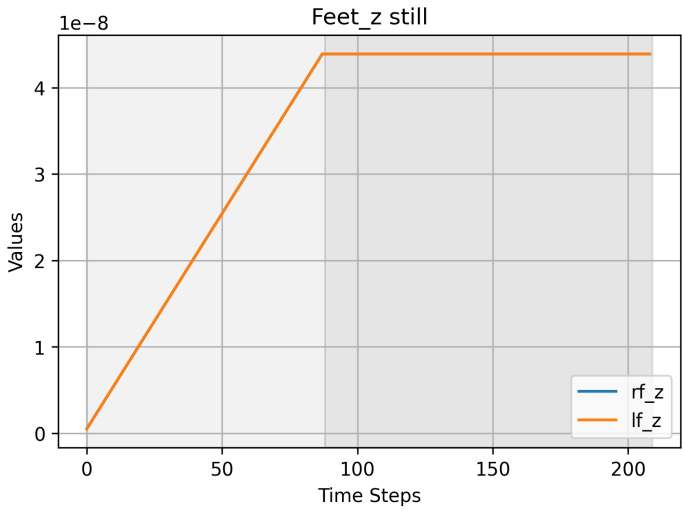


Figure 2: Feet along Z (scale is 1e-8)

State and Input Contact Values

In the following plots, the main components of the state and input vectors solutions are compared with the reference at each contact step.

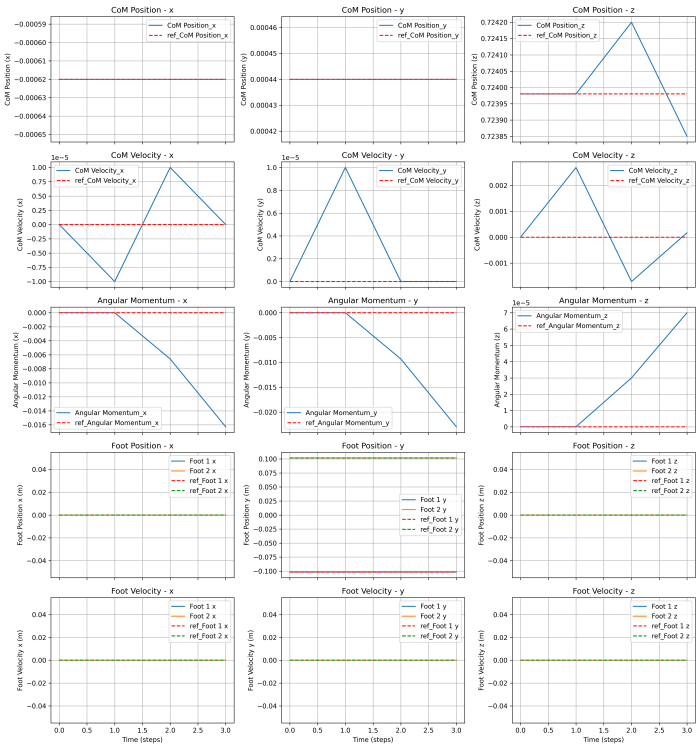


Figure 3: Trajectory vs Reference: state vector

Contact Forces

Another important dynamic aspect pertains to the forces involved. The input vector includes the contact wrench, which describes the mechanical influence that a contact point (such as a foot or hand) exerts on the robot, or vice versa. In order to satisfy the dynamic balance laws, the environment should exert a reaction force along the Z-axis that is equal to the gravitational force multiplied by the robot's mass. Table **??** below lists the contact forces exerted by the environment on both feet. As shown in the table, when both feet are on the ground, the gravitational force is evenly distributed between the left and right foot.

Right Foot Z	Left Foot Z	Σ_L^k
49.0644	49.0531	[0., 0.]
48.8976	49.9925	[0., 0.]
48.0847	49.0895	[0., 0.]

Table 8: Z-axis gravity force and Σ_L^k values

Components of contact wrench, rotational and translational forces, are graphically expressed in the following plots:

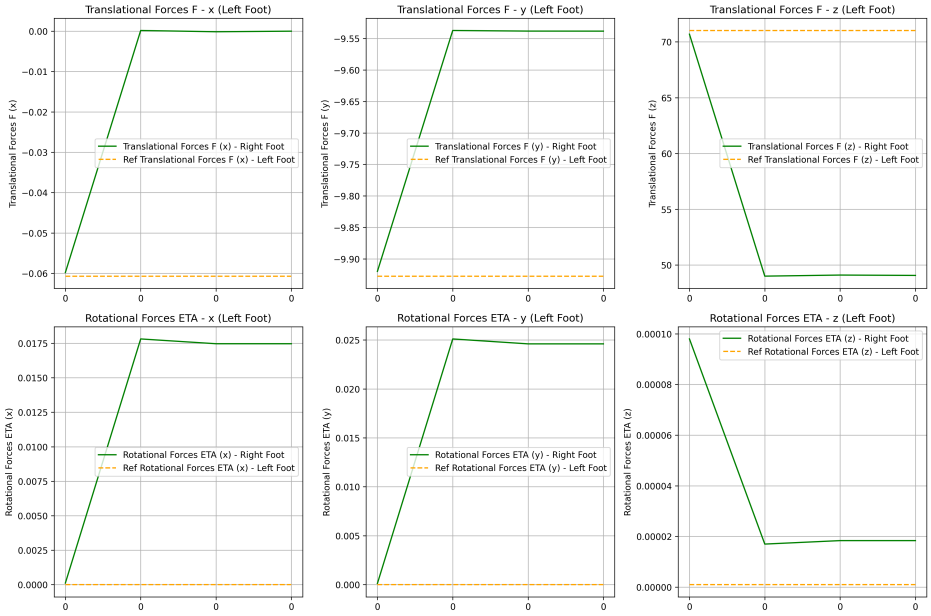


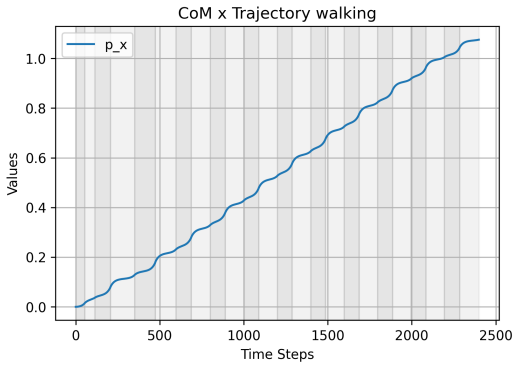
Figure 4: Trajectory vs Reference: Forces

5.4 Walking Task: results

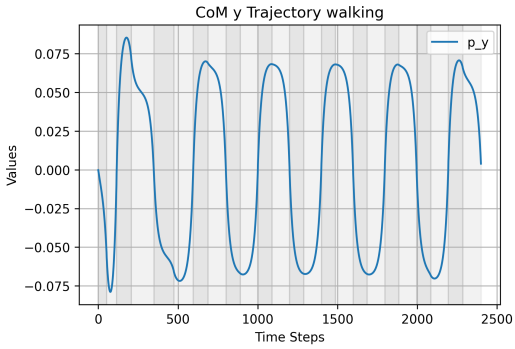
In this case the state vector has dimension 24 and the input vector has dimension 23.

CoM Plots

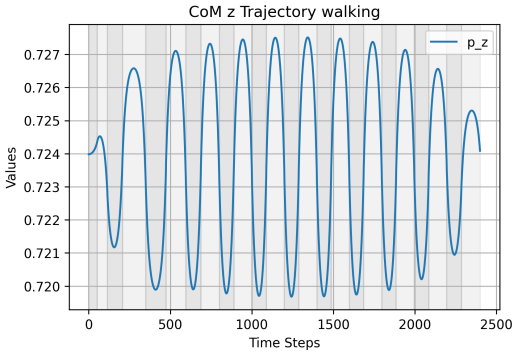
The following plots illustrate the trajectory of the Center of Mass (CoM) over time. The alternating light and dark grey backgrounds in the plots indicate the transitions between different phases of the walking motion. As shown in Fig. **??**, the CoM exhibits linear motion along the x-axis. In the y-direction, as illustrated in Fig. **??**, the CoM moves toward the foot that remains on the ground while the other foot is lifted. Specifically, if we consider the first two contact phases: during the first phase, the robot is in double support (both feet on the ground), while in the second phase, the left foot is expected to lift. At the end of the first phase, the CoM has shifted towards the right foot, demonstrating the transition in weight distribution. This pattern of alternating between double and single support phases continues throughout the walking cycle, with the CoM constantly adjusting its position. In the z-direction, shown in Fig. **??**, the CoM exhibits a slight up-and-down motion, with a relatively small range of displacement. This vertical motion is consistent with the natural bobbing motion of the body during walking, providing necessary stability and balancing forces. This pattern of CoM movement is crucial for maintaining dynamic stability as the robot progresses through each phase of the gait. In summary, the CoM trajectory provides key insights into the robot's walking mechanics, including the shift in weight distribution during transitions between support phases and the vertical adjustments required for balance. These movements are essential for achieving stable locomotion.



(a) Com X Trajectory 1



(b) Com Y Trajectory 1



(c) Com Z Trajectory 1

Figure 5: Com Trajectory

Furthermore, the image below represent the foot position along the Z-axis in time. As one can see, they follow a parabolic profile.

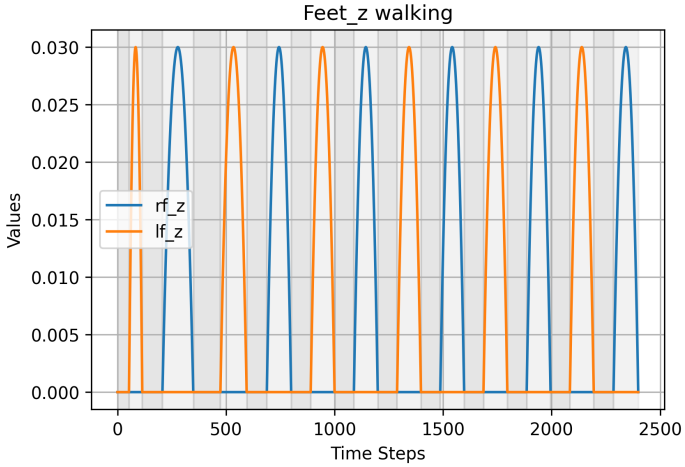


Figure 6: Feet along Z

State and Input Contact Values

As previously said, in the following plots, the main components of the state and input vectors solutions are compared with the reference at each contact step.

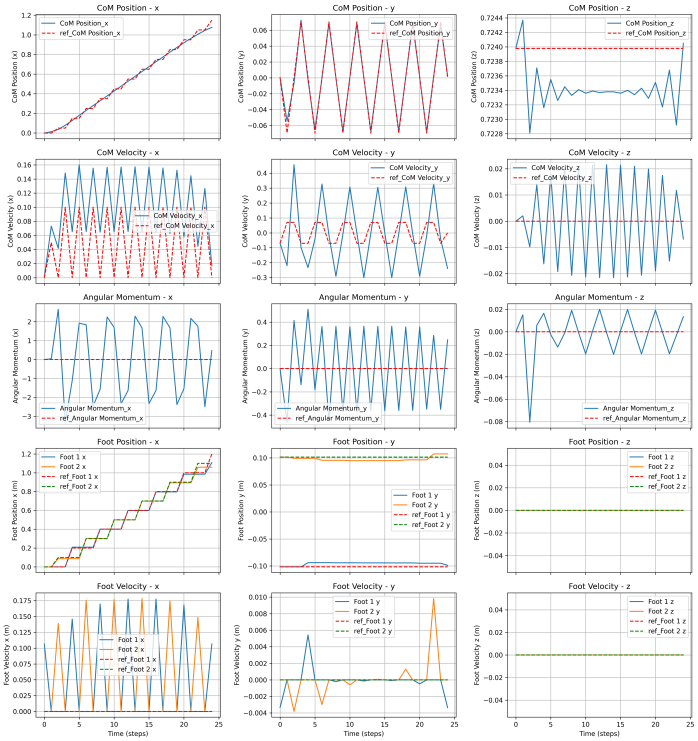


Figure 7: Trajectory vs Reference: state vector

Contact Forces

Another key dynamic aspect, which we have already discussed, concerns the forces involved. Specifically, the input vector includes the contact wrench, which captures the mechanical interaction between the robot and contact points, such as the feet or hands. To maintain dynamic balance, the environment must exert a reaction force along the Z-axis equal to the gravitational force, which is the robot's mass multiplied by gravity. Table **??** below lists the contact forces exerted by the environment on both feet. As shown, when both feet are on the ground, the gravitational force is evenly distributed between the left and right foot, while during single support, the force is applied only to the foot that remains on the ground.

Right Foot Z	Left Foot Z	Σ_L^k
49.0644	49.0531	[0., 0.]
97.9551	0	[0., -]
48.8973	49.65	[0., 0.]
0	97.4904	[-, 0.]
49.6528	49.1563	[0., 0.]
97.3081	0	[0., -]
49.2399	49.7241	[0., 0.]
0	97.2091	[-, 0.]
49.7388	49.293	[0., 0.]
97.1669	0	[0., -]
49.3007	49.7613	[0., 0.]
0	97.1486	[-, 0.]
49.762	49.31	[0., 0.]
97.1443	0	[0., -]
49.3062	49.7655	[0., 0.]
0	97.1495	[-, 0.]
49.7557	49.3045	[0., 0.]
97.1685	0	[0., -]
49.2815	49.7467	[0., 0.]
0	97.216	[-, 0.]
49.7003	49.254	[0., 0.]
97.3264	0	[0., -]
49.1286	49.6485	[0., 0.]
0	97.5828	[-, 0.]

Table 9: Z-axis gravity force and Σ_L^k values

Again, components of contact wrench, rotational and translational forces, are graphically expressed in the following plots:

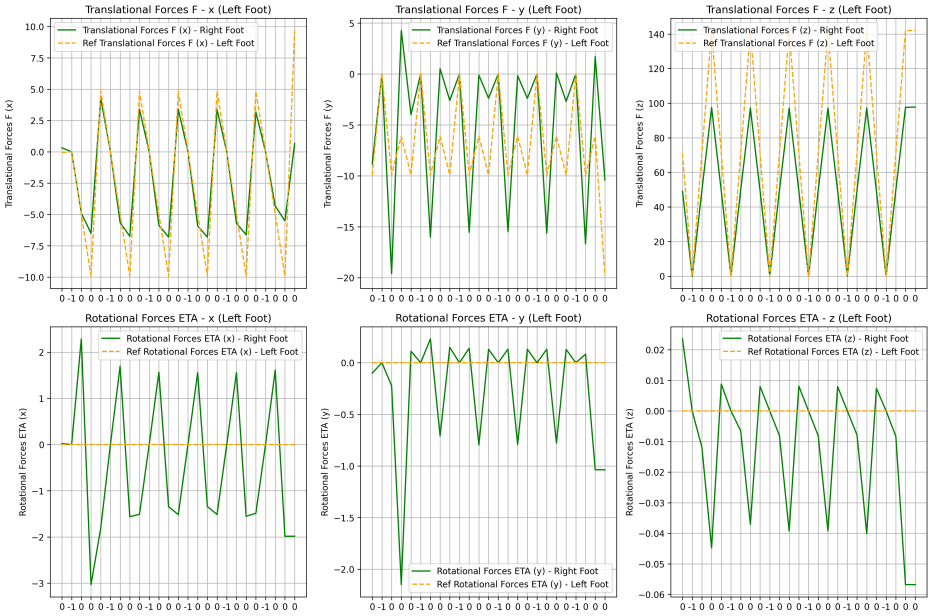


Figure 8: Trajectory vs Reference: Forces

Robot representation

In this section it is possible to visualize the path followed by the CoM, right and left foot in time:

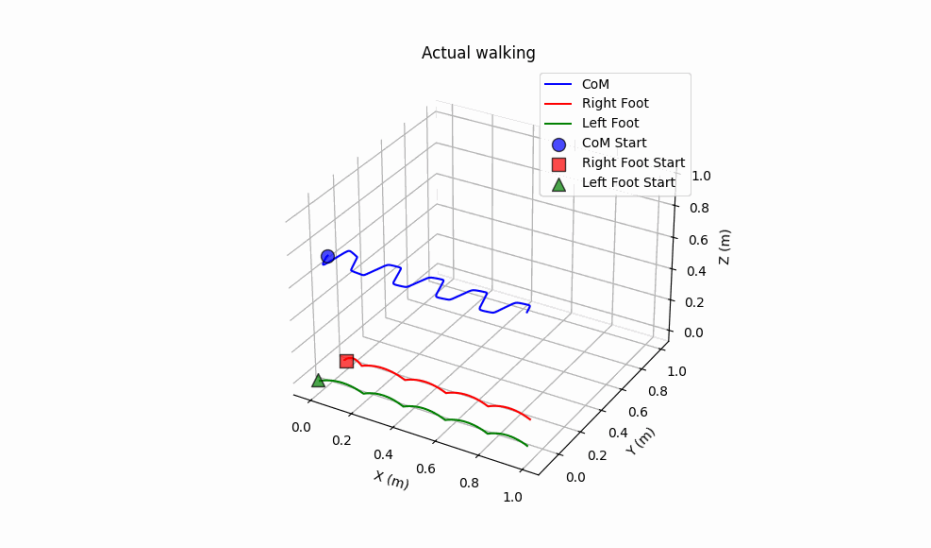


Figure 9: Walking