Real-Time Safe Bipedal Robot Navigation using Linear Discrete Control Barrier Functions

Chengyang Peng[1†], Victor Paredes[1†], Guillermo A. Castillo[2] and Ayonga Hereid[1]

***Abstract*— Safe navigation in real-time is an essential task for humanoid robots in real-world deployment. Since humanoid robots are inherently underactuated thanks to unilateral ground contacts, a path is considered safe if it is obstacle-free and respects the robot's physical limitations and underlying dynamics. Existing approaches often decouple path planning from gait control due to the significant computational challenge caused by the full-order robot dynamics. In this work, we develop a unified, safe path and gait planning framework that can be evaluated online in real-time, allowing the robot to navigate clustered environments while sustaining stable locomotion. Our approach uses the popular Linear Inverted Pendulum (LIP) model as a template model to represent walking dynamics. It incorporates heading angles in the model to evaluate kinematic constraints essential for physically feasible gaits properly. In addition, we leverage discrete control barrier functions (DCBF) for obstacle avoidance, ensuring that the subsequent foot placement provides a safe navigation path within clustered environments. To guarantee real-time computation, we use a novel approximation of the DCBF to produce linear DCBF (LDCBF) constraints. We validate the proposed approach in simulation using a Digit robot in randomly generated environments. The results demonstrate that our approach can generate safe gaits for a non-trivial humanoid robot to navigate environments with randomly generated obstacles in real-time.**

I. Introduction

Humanoid robots exhibit great dexterity and agility to perform different locomotion activities [1]–[3]. They are expected to be deployed to real-life environments such as warehouses and assembly lines. Such clustered environments pose the challenge of navigating around obstacles while maintaining stable walking in real-time [4], [5]. Yet humanoid robots are inherently underactuated due to their unilateral ground contacts; thus, a strong coupling exists between path planning and gait control. Their high-dimensional, nonlinear, and hybrid dynamics further complicate real-time motion planning. For fully actuated legged robots, one can decouple the path planning problem from motion control by finding a collision-free path without accounting for the robot dynamics and motion control [6], [7]. Then, a feedback controller aware of the robot dynamics can be developed to track the provided path. However, under-actuated humanoids would fall if the planned path did not account for the robot dynamics. This coupling of planning and dynamics usually requires solving specific gait optimization problems based on the robot's full-order [8]–[10] or reduced-order model [3], [11], [12]. However, using the full-order model for long-horizon path planning requires significant computation time, making them non-amenable for real-time online planning.

This work was supported in part by the National Science Foundation under grant FRR-21441568.

[1]Mechanical and Aerospace Engineering, The Ohio State University, Columbus, OH, USA. (peng.947, paredescauna.1, hereid.1)@osu.edu.

[2]Electrical and Computer Engineering, The Ohio State University, Columbus, OH, USA; castillomartinez.2@osu.edu

† These authors contributed equally.

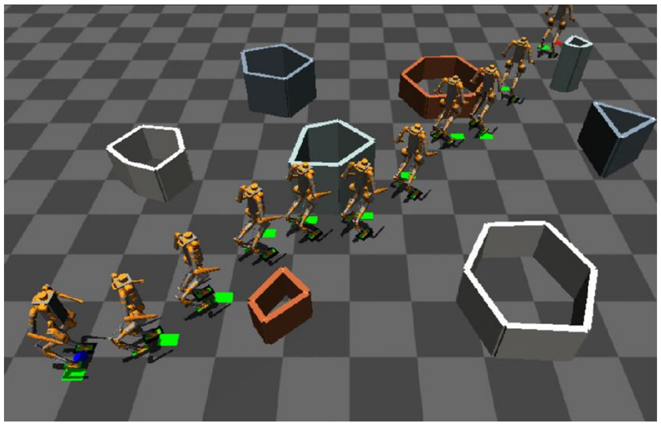
Fig. 1. Safe navigation planning is tested in MuJoCo with the Digit robot.

To mitigate the computational challenge, reduced-order template models are often used to approximate the walking dynamics of the robot and plan gaits with reduced computational burden [13]–[15]. A famous example of such template models is the linear inverted pendulum (LIP) model [16]. In particular, the LIP model represents the robot dynamics using the center of mass (CoM) position and velocity and allows the control of the CoM velocity at the end of the next step. The LIP model provides a lower-dimensional representation of the robot dynamics that can be used to plan a desired foot placement that renders stable walking gaits.

In this work, we introduce a modified 3D-LIP model with heading angles to use the LIP model for both path and gait planning and enforce necessary kinematic constraints. Including heading angles and turning rates allows for expressing kinematic constraints in the local coordinate frame of the robot, appropriately addressing the different motion constraints in sagittal and frontal planes, as well as left foot stance and right foot stance, of walking robots. A model predictive control based on the discrete-time step-to-step dynamics of this 3D-LIP model, denoted as LIP-MPC, is proposed to unify path and gait planning. In particular, our MPC formulation uses discrete control barrier functions (DCBFs) for safe obstacle avoidance. Control barrier functions have been successfully applied to controlling legged robots and are now widely used to ensure safety in path planning [17]–[20].

DCBF, particularly, is well-suited for the discrete-time step-to-step dynamic model [21], [22]. However, many barrier functions that describe obstacles are nonlinear, leading to nonlinear constraints when foot placement is treated as the decision variable. Hence, despite the 3D-LIP model being linear, the nonlinearity in both kinematic and path constraints hinders the real-time computation of the MPC problem.

To mitigate computational efficiency, this paper proposes two key ingredients: pre-computing heading angles and approximated linear DCBFs. Preprocessing the turning rate for each prediction step allows us to linearize the kinematic constraints within the MPC. Additionally, we introduce a novel linear discrete-time Control Barrier Function (LDCBF) to establish linear, feasible obstacle avoidance constraints for convex obstacles. These adjustments transform the optimization problem into a linearly constrained quadratic programming (QP) problem, significantly reducing computational demands and enabling real-time, safety-critical navigation for bipedal robots.

The rest of the paper is organized as follows: Section II introduces the 3D-LIP model with heading angles and presents the formulation of the LIP-MPC with linear kinematic constraints for feasible gait planning. Section III details the design of the obstacle avoidance constraints, introducing the novel LDCBF expression. Section IV shows the application of the proposed LIP-MPC in simulation, showcasing the real-time safe navigation of the bipedal robot Digit. We presented the results of two different turning rate preprocessing strategies: global goal-oriented and subgoal-oriented. Finally, Section V concludes the contributions, limitations, and future work.

II. Real-Time Gait Planing with LIP-MPC

The full-order dynamics of bipedal robots are high-dimensional, nonlinear, and hybrid, which poses significant computational challenges for planning and control. In this section, we introduce a three-dimensional linear inverted pendulum (3D-LIP) model with heading angles, and formulate a model predictive control formulation that uses the step-to-step discrete dynamics for step planning. We propose several novel measures to ensure kinematics and safety constraints can be represented as linear constraints, thereby allowing real-time gait generation in crowded environments.

A. 3D-LIP Model with Heading Angle

The LIP model assumes the robot keeps a constant center of mass (CoM) height H during the walking step. The motion of the robot along the x direction can be expressed as:

$$\dot{v}_x = \frac{g}{H}(p_x - f_x), \tag{1}$$

where (p_x, v_x) are the CoM position and velocity in x-axis, f_x is the stance foot position, and g is the gravitational acceleration. If we assume that each walking step takes a fixed time T, the closed-form solution of the step-to-step discrete dynamics can be written as:

$$\begin{bmatrix} p_{x_{k+1}} \\ v_{x_{k+1}} \end{bmatrix} = \mathbf{A_d} \begin{bmatrix} p_{x_k} \\ v_{x_k} \end{bmatrix} + \mathbf{B_d} f_{x_k}, \tag{2}$$

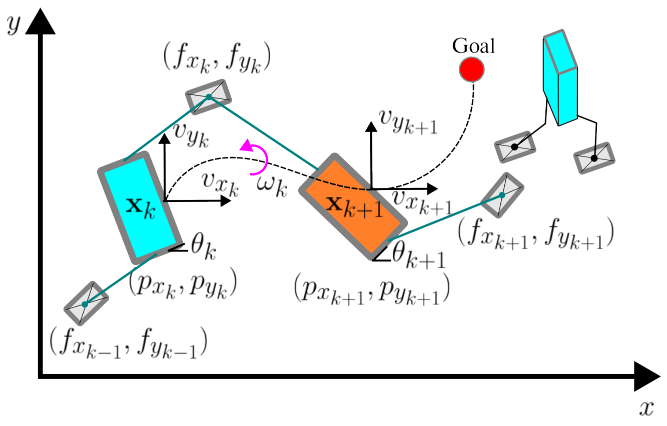

Fig. 2. Representation of state transition of the LIP model in the global frame. The state $\mathbf{x}_k$ in step k (shown in cyan) evolves to the state $\mathbf{x}_{k+1}$ (shown in orange) when stepping at point (f_{x_k}, f_{y_k}) and with a turning rate of ω_k.

with

$$\mathbf{A_d} := \begin{bmatrix} \cosh(\beta T) & \frac{\sinh(\beta T)}{\beta} \\ \beta \sinh(\beta T) & \cosh(\beta T) \end{bmatrix}, \quad \mathbf{B_d} := \begin{bmatrix} 1 - \cosh(\beta T) \\ -\beta \sinh(\beta T) \end{bmatrix}, \tag{3}$$

where $\beta = \sqrt{g/H}$, and p_{x_k} and v_{x_k} represent the CoM position and velocity at the beginning of k-th step.

The motion in the y-axis direction has the same expression as the x-axis. In this work, we will also consider the heading angle θ and its turning rate ω, which can be used to determine the orientation of the local robot coordinates. This allows kinematics constraints (e.g., body velocity, leg reachability, etc.) to be expressed properly in our gait planning problem. By defining the state $\mathbf{x} := [p_x, v_x, p_y, v_y, \theta]^T \in \mathcal{X} \subset \mathbb{R}^5$ and the control variable $\mathbf{u} := [f_x, f_y, \omega]^T \in \mathcal{U} \subset \mathbb{R}^3$, the step-to-step dynamics of the 3D-LIP model can be written as:

$$\mathbf{x}_{k+1} = \mathbf{A_L}\mathbf{x}_k + \mathbf{B_L}\mathbf{u}_k, \tag{4}$$

with:

$$\mathbf{A_L} := \begin{bmatrix} \mathbf{A_d} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{A_d} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & 1 \end{bmatrix}, \quad \mathbf{B_L} := \begin{bmatrix} \mathbf{B_d} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{B_d} & \mathbf{0} \\ 0 & 0 & T \end{bmatrix}. \tag{5}$$

We show in Fig. 2 the projection of the LIP states and controls in the $x-y$ plane during multiple steps.

B. Safe Gait Planning with Model Predictive Control

Our work uses the step-to-step 3D-LIP dynamics in (4) to formulate a Model Predictive Control (MPC) problem to compute optimal stepping positions for stable locomotion and safe navigation. To be able to respond to the instantaneous states of the robot in real time, we compute the next discrete states (i.e., the LIP states at the beginning of a step) using the closed form solution of (1). Let us denote

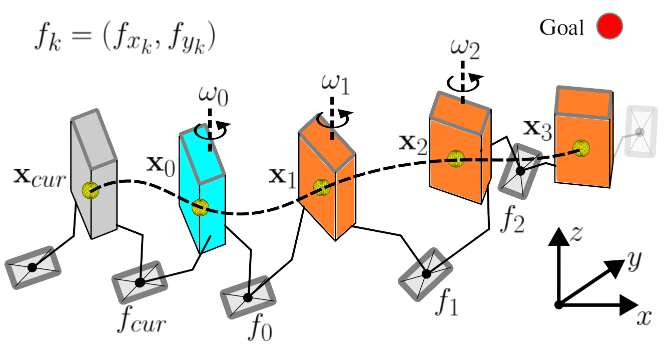

Fig. 3. An illustration of the LIP-MPC formulation when $N = 3$. The planner first estimates the state at the end of the current step, $\mathbf{x}_0$, given the current instantaneous state, $\mathbf{x}_{cur}$, and the stance foot position contained in $\mathbf{u}_{cur}$. The turning rates in the subsequent steps are pre-computed, where the stepping positions f_k will be optimized by the LIP-MPC.

it as $\mathbf{x}_0$, which will serve as the starting point in the MPC formulation. Thus, the LIP-MPC can be stated as:

$$
\begin{aligned}
J^* = \min_{\mathbf{u}_{0:N-1}} \quad & \sum_{k=1}^{N} q(\mathbf{x}_k) && (6)\\
\text{s.t} \quad & \mathbf{x}_k \in \mathcal{X}, \ k \in [1, N] \\
& \mathbf{u}_k \in \mathcal{U}, \ k \in [0, N-1] \\
& \mathbf{x}_{k+1} = \mathbf{A_L}\mathbf{x}_k + \mathbf{B_L}\mathbf{u}_k, \ k \in [0, N-1] \\
& \mathbf{c}_l \le \mathbf{c}(\mathbf{x}_k, \mathbf{u}_k) \le \mathbf{c}_u, \ k \in [0, N-1]
\end{aligned}
$$

where, $\mathcal{X}$ is the set of allowed states and $\mathcal{U}$ is the set of admissible controls. $q(\mathbf{x}_k)$ is the cost function, defined to evaluate the distance of a sequence of predicted states from the goal position (g_x, g_y). Minimizing this cost implies that the robot moves towards the goal position. In particular, we consider quadratic cost function, defined as:

$$
q(\mathbf{x}_k) = (p_{x_k} - g_x)^2 + (p_{y_k} - g_y)^2 \quad \forall k \in [1, N]. \tag{7}
$$

The kinematics and path constraints will be captured in $\mathbf{c}(\mathbf{x}_k, \mathbf{u}_k)$. These constraints are often nonlinear as they must be evaluated in the local coordinate frame, which hinders the real-time computation of (6). In the following discussion, we introduce several novel measures to linearize these constraints, thereby ensuring the optimization problem in (6) can be solved in real-time.

C. Heading Angle Preprocessing

The nonlinearity in $\mathbf{c}(\mathbf{x}_k, \mathbf{u}_k)$ is often due to the inclusion of the heading angle, θ_k. To linearize these constraints, we propose to pre-compute the turning rates $\bar{\omega}_k, \forall k \in [0, 1, ..., N-1]$ to keep them fixed in the MPC calculation. A straightforward and simple way is to calculate the target heading angle as the direction from the current position towards the goal position. The required turning rate $\bar{\omega}_k$ is calculated by smoothly turning from the current heading to the target heading angle in N steps. To avoid sharp turns, we also impose a limit on the turning rate $|\bar{\omega}_k| \le 0.156\pi \,\mathrm{rad}/s$. Fig. 3 shows the case when MPC prediction step $N = 3$, and the fixed turning rates for all steps.

D. Linearized Safety Constraints

In addition to the obstacle avoidance constraints, which will be presented in the next section, we enforce multiple kinematics constraints to ensure that the optimized stepping positions are physically feasible on the robot hardware. With pre-computed heading angles, these constraints will become linear inequality constraints, as discussed below.

Walking Velocities Since the 3D-LIP states in (4) are expressed in the world coordinates, to properly limit the walking velocities, we must compute the body velocities in the robot's local coordinates. In particular, the walking velocity constraint can be expressed as:

$$
\begin{bmatrix} v_{x_{\min}} \\ v_{y_{\min}} \end{bmatrix} \le \begin{bmatrix} \cos\theta_k & \sin\theta_k \\ -\sin\theta_k & \cos\theta_k \end{bmatrix} \begin{bmatrix} v_{x_{k+1}} \\ s_v v_{y_{k+1}} \end{bmatrix} \le \begin{bmatrix} v_{x_{\max}} \\ v_{y_{\max}} \end{bmatrix} \tag{8}
$$

$\forall k \in [0, N-1]$, where, $v_{x_{\min}}, v_{x_{\max}}, v_{y_{\min}}, v_{y_{\max}}$ are the lower bounds and upper bounds of the robot longitudinal and lateral velocities, respectively. s_v is a sign function that depends on which foot is the stance. If the right foot is the stance, then $s_v = 1$; otherwise, $s_v = -1$. This allows lateral velocity at the end of each step to be properly limited to ensure that the foot lands on the opposite side, thereby preventing potential leg crossing and collisions.

Leg Reachability. The swing foot reachability constraint is used to prevent over-extension of the swing leg. Our previous work [15] limits the Euclidean distance between the robot's center of mass (CoM) and the subsequent stepping position. Despite being straightforward, it introduces nonlinearity in the optimization. We reformulate the constraint to decompose the Euclidean distance into longitudinal and lateral components based on the local coordinate frame. This allows a linear expression of the reachability constraint, given as:

$$
\begin{bmatrix} -l_{\max} \\ -l_{\max} \end{bmatrix} \le \begin{bmatrix} \cos\theta_k & \sin\theta_k \\ -\sin\theta_k & \cos\theta_k \end{bmatrix} \begin{bmatrix} p_{x_k} \\ p_{y_k} \end{bmatrix} \le \begin{bmatrix} l_{\max} \\ l_{\max} \end{bmatrix} \tag{9}
$$

$\forall k \in [0, N-1]$, where, $l_{\max}$ is the maximum reachable distance of the swing foot in both directions on the ground.

Maneuverability Constraint. The maneuverability constraint provides a good safety strategy that decelerates the robot's walking speed while turning. It couples the turning rate with longitudinal velocity, as shown below:

$$
\begin{bmatrix} \cos\theta_k & \sin\theta_k \end{bmatrix} \begin{bmatrix} v_{x_k} \\ v_{y_k} \end{bmatrix} \le v_{x_{\max}} - \frac{\alpha}{\pi}|\omega_k|, \tag{10}
$$

where α is a positive coefficient that balances the turning rate and walking velocity. In our work, we empirically set $\alpha = 3.6$ for the Digit robot.

III. Obstacle Avoidance using LDCBF

We enforce obstacle avoidance by incorporating Discrete Control Barrier Functions (DCBF). We showed in our previous work that regular DCBFs produce nonlinear constraints [15], which are not amenable for real-time planning. In this section, we propose a novel strategy to express the obstacle avoidance constraint as linear DCBFs (LDCBF).

For the discrete states of the robot $\mathbf{x}_k \in \mathcal{X} \subseteq \mathbb{R}^n$, and discrete control inputs $\mathbf{u}_k \in \mathcal{U} \subseteq \mathbb{R}^m$, if there is a continuous and differentiable function $h : \mathbb{R}^n \rightarrow \mathbb{R}$, the safety set $\mathcal{C}$ and the safety boundary $\partial\mathcal{C}$ of the system may be defined as:

$$\begin{aligned} \mathcal{C} &= \{\mathbf{x}_k \in \mathcal{X} | h(\mathbf{x}_k) \geq 0\}, \\ \partial\mathcal{C} &= \{\mathbf{x}_k \in \mathcal{X} | h(\mathbf{x}_k) = 0\}. \end{aligned} \tag{11}$$

Then, $h(\cdot)$ is a discrete control barrier function (DCBF) if there exists a class κ function satisfying $0 < \gamma(h(\mathbf{x})) \leq h(\mathbf{x})$, and following conditions can be hold [21], [22]:

$$\forall\ \mathbf{x}_k \in \mathcal{C}. \quad \exists\ \mathbf{u}_k \text{ s.t. } \triangle\, h(\mathbf{x}_k, \mathbf{u}_k) \geq -\gamma(h(\mathbf{x}_k)), \tag{12}$$

where $\triangle h(\mathbf{x}_k, \mathbf{u}_k) := h(\mathbf{x}_{k+1}) - h(\mathbf{x}_k)$. In the discrete domain, γ can be also a scalar that $0 < \gamma \leq 1$. So, a DCBF constraint can be written as:

$$\exists\ \mathbf{u}_k \text{ s.t. } h(\mathbf{x}_{k+1}) + (\gamma - 1)h(\mathbf{x}_k) \geq 0. \tag{13}$$

A. LDCBF for Path Planning

To avoid obstacles, we need to compare the position of the robot against the location of the obstacles. For this purpose, we use the vector $\vec{x} := [p_x, p_y]$ that represent a position on the map. Moreover, the selector matrix S_x maps the 3D-LIP states to the robot's position via $\vec{x} = S_x\mathbf{x}$, with

$$S_x = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \tag{14}$$

In general, for path planning, each obstacle can be represented by a function $F(\vec{x}) = 0$ as shown in Fig. 4. Typical representations of this function involve circles and ellipses due to their simpler mathematical form. This function can be constructed such that $F(\vec{x}) < 0$ whenever $\vec{x}$ is inside the obstacle, and $F(\vec{x}) \geq 0$ otherwise. This fact shows that is convenient to choose $h(\mathbf{x}) = F(\vec{x})$ as a DCBF. Even if an obstacle have a complex shape, it is possible to construct a single $h(\mathbf{x})$ as a nonlinear composition of other simpler DCBF [23], [24]. However, as long as the DCBF function is nonlinear, the DCBF constraint (13) will also be generally non-linear, which is not amenable for real-time computation.

Given a robot position $\vec{x}_r(t)$ at an instant t, it is possible to approximate the safe region with a LDCBF constraint with a half-space provided by the hyperplane $h(\mathbf{x})$ by finding the point $\vec{c}$ that represents the closest point in $F(\vec{x}) = 0$ to $\vec{x}_r(t)$. The half-space approximation (Fig. 4, c) is given by:

$$h(\mathbf{x}) = \frac{\nabla F(\vec{c})^T}{||\nabla F(\vec{c})||}(\vec{x} - \vec{c}) \geq 0 \tag{15}$$

where, $\frac{\nabla F(\vec{c})^T}{||\nabla F(\vec{c})||}$ represents the normal vector of $F(\vec{x})$ at $\vec{x} = \vec{c}$ that points outwards the obstacle.

Proposition. *If $F(\vec{x})$ is convex, or we use a surrogate convex function $C(\vec{x})$ that contains $F(\vec{x})$, as shown in Fig. 4 b), we guarantee that the safe region given by (15) does not contain any unsafe points corresponding to the obstacle.*

Proof. Assume a differentiable function $F(\vec{x}) : \mathbb{R}^2 \rightarrow \mathbb{R}$. If $F(\vec{x})$ is convex, i.e, for any two points $\vec{x}$ and $\vec{y}$ in the obstacle contour, $F(\vec{y}) \geq F(\vec{x}) + \nabla F(\vec{x})^T(\vec{y} - \vec{x})$, then since $F(\vec{x}) = F(\vec{y}) = 0$ we get that $0 \geq \nabla F(\vec{x})^T(\vec{y} - \vec{x})$. Consequently, for any point $\vec{c}$, we observe that the half-space given by $h(\mathbf{x}) = \frac{\nabla F(\vec{c})^T}{||\nabla F(\vec{c})||}(\vec{x} - \vec{c}) \geq 0$ does not cross the obstacle, thus is a safe region. □ □

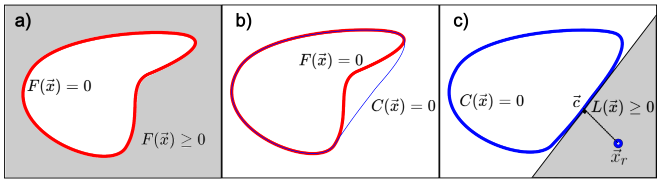

Fig. 4. a) A function $F(\vec{x})$ represents the outline of an obstacle. The safe path corresponds to the condition $F(\vec{x}) \geq 0$, where $\vec{x}$ represents a position in the map. b) The first step to linearize the DCBF condition is to generate the convex function $C(\vec{x})$ that contains $F(\vec{x})$. c) We use this convex function to consider the robot position $\vec{x}_r$ and find the half-plane formed by the closest point $\vec{c}$ and its normal vector, representing the linear approximation of the safe region.

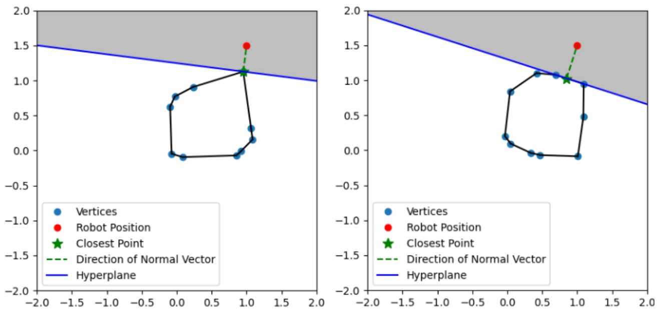

Fig. 5. Left: The closest point $\vec{c}$ to the robot is a vertex of the convex hull. Right: The closest point $\vec{c}$ lies within an edge of the convex hull. In both cases, the unit normal vector is calculated as the normalization of the line connecting the closest point to the robot position.

Moreover, finding a continuous function $F(\vec{x})$ that represents an obstacle might not be practical. Instead, we can quickly generate a convex polygon that contains the obstacle. In the discontinuous case, the convexity also guarantees that the resulting half-space does not contain any part of the obstacle itself. The construction of the hyperplane can be visualized in Fig. 5. The procedure consists of finding the closest point $\vec{c}$ that lives in $F(\vec{x}) = 0$ and it is closest to the robot position $\vec{x}_r(t)$ at time t. This closest point $\vec{c}$ can be either on an edge or be one of the vertices of the convex polygon. The computation of the normal vector of an edge is $\eta = \frac{\nabla F(\vec{c})^T}{||\nabla F(\vec{c})||}$, however, for a vertex, we use $\eta = \frac{\vec{x}_r(t) - \vec{c}}{||\vec{x}_r(t) - \vec{c}||}$, as illustrated in Fig. 6. In either case, the half-space is represented by:

$$h(\mathbf{x}) = \eta^T(\vec{x} - c) \geq 0. \tag{16}$$

Proposition. *Given a linear discrete model that represents the motion of the robot from x_k to x_{k+1} as in (4) and a DCBF constraint of the form (13). It can be shown that $h(x_k) = \eta^T(\vec{x}_k - \vec{c})$ can yield a LDCBF constraint if the closest point $\vec{c}$ and the normal vector η are approximated as constants for the next time-instant $(k + 1)$.*

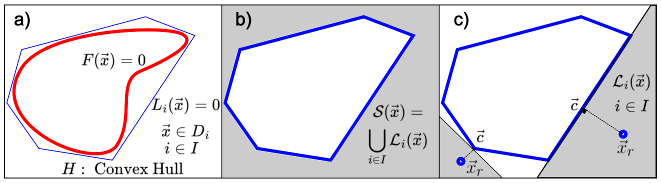

Fig. 6. a) A general nonlinear $F(\vec{x})=0$ obstacle contour can be approximated by a convex polygon with lines $L_i(\vec{x})=0$ delimiting a convex hull. b) Each line produces a safe half-space $\mathcal{L}_i=\{L_i(\vec{x})\geq 0\}$. The union of these half-spaces $\mathcal{S}(\vec{x})$ produces the safe region for the robot. c) We simplify $\mathcal{S}(\vec{x})$ considering a unique hyperplane per obstacle depending on the position of the robot $\vec{x}_r$. We compute the closest point to the robot position $\vec{c}$ and construct the LDCBF as done in (16).

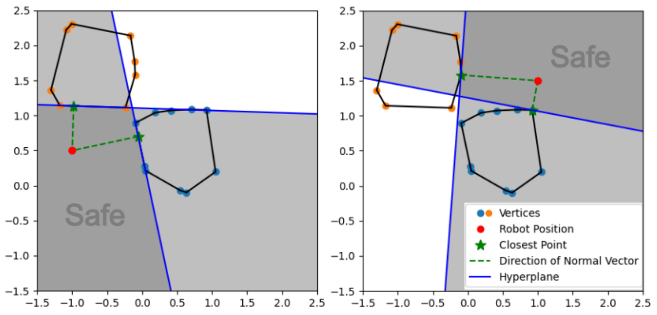

Fig. 7. Left: A robot starting in the lower left side of the map will experience a safe region composed by the union of half-spaces produced by each obstacle. Right: As the robot moves, the safe region will be updated potentially allowing the robot to reach a target position.

Proof. We show this by expanding the LDCBF constraint:

$$h(\mathbf{x}_{k+1})+(\gamma-1)h(\mathbf{x}_k)\geq 0$$
$$\eta^T S_x B_L\mathbf{u}_k+\eta^T S(A_L\mathbf{x}_k+(\gamma-1)\mathbf{x}_k)-\eta^T\gamma\vec{c}\geq 0,$$

where $S_xB_L\neq 0$ and the constraint is linear in $\mathbf{u}_k$, it can be implemented in a QP-based optimization in real-time. □

In the case of multiple obstacles, we add one linear constraint per obstacle, producing a safe region given by the intersection of each respective half-space as in Fig. 7.

IV. Simulation Results

This section presents simulation results that demonstrate the effectiveness and performance of the proposed LIP-MPC in navigating clustered environments.

A. Simulation Setup

To ensure the safe navigation of the robot, we implemented a hierarchical structure where the LIP-MPC is responsible for generating the next stepping position at 20 Hz update frequency, and a low-level task space controller keeps the CoM height constant, the torso upright and places the swing foot at its desired location at 1 kHz update frequency. We utilized the Agility Robotics' Digit humanoid as our testing platform in simulation. To assess the efficacy and performance of our proposed method, we randomly generated several test environments within the MuJoCo simulator, each featuring eight polygon-shaped obstacles of varying sizes and shapes. In these test scenarios, the starting position was set at $[0,0]$ m, with the goal at $[10,10]$ m. The robot's Center of Mass (CoM) height was maintained at $H=1$ m, the step duration was set to $T=0.4$ s, and the MPC prediction horizon was defined as $N=3$. Table I presents the selected values of weights and limits used throughout all the tests in this paper.

TABLE I. The value of each control parameter used in the simulation throughout this work.

Parameters	Value
$[v_{x_{\min}}, v_{x_{\max}}]$	$[-0.1, 0.8]$ m/s
$[v_{y_{\min}}, v_{y_{\max}}]$	$[0.1, 0.4]$ m/s
$l_{\max}$	$0.1\sqrt{3}$m
α	1.44
γ	0.3

B. Global Goal Oriented Planning

Our approach begins by determining the required turning rate for each prediction step based on the global goal position and then calculating the corresponding foot placement through our linearized LIP-MPC. We tested this method in two randomly generated environments. Fig. 8 illustrates the active LDCBF and dynamically changing safety region from the starting position to the goal in one test. The form and number of LDCBF constraints adapt based on the robot's position. To minimize redundancy and overlap in constraint effects, only obstacles within a 4-meter radius are considered as active LDCBF constraints in our method.

The simulation results for both environments are shown in Fig. 9. The linearized LIP-MPC successfully generated stable stepping positions, guiding the robot safely to the goal without falling. Since the target heading angle is computed so that the robot always faces the goal during walking, the heading angle remains relatively constant during the test, as shown in Fig. 9. This strategy reduces the robot's flexibility. When the obstacles are located in the robot's walking direction, this approach avoids obstacles in slow sidewalking rather than with more agile forward walking movements. While bipedal robots are omnidirectional, they move faster and more stably in longitudinal than lateral directions. In these two tests, the robot takes an average of 75 steps with nearly 31 seconds to reach the goal.

C. Sub-goals Oriented Planning

We enhance the navigation framework by introducing sub-goals between the starting position and the final goal. This allows for more flexible steering during the navigation to the goal while preserving the linear form of the LDCBFs. In particular, we employ the Rapidly-exploring Random Tree (RRT) algorithm as a global planner to generate sub-goals

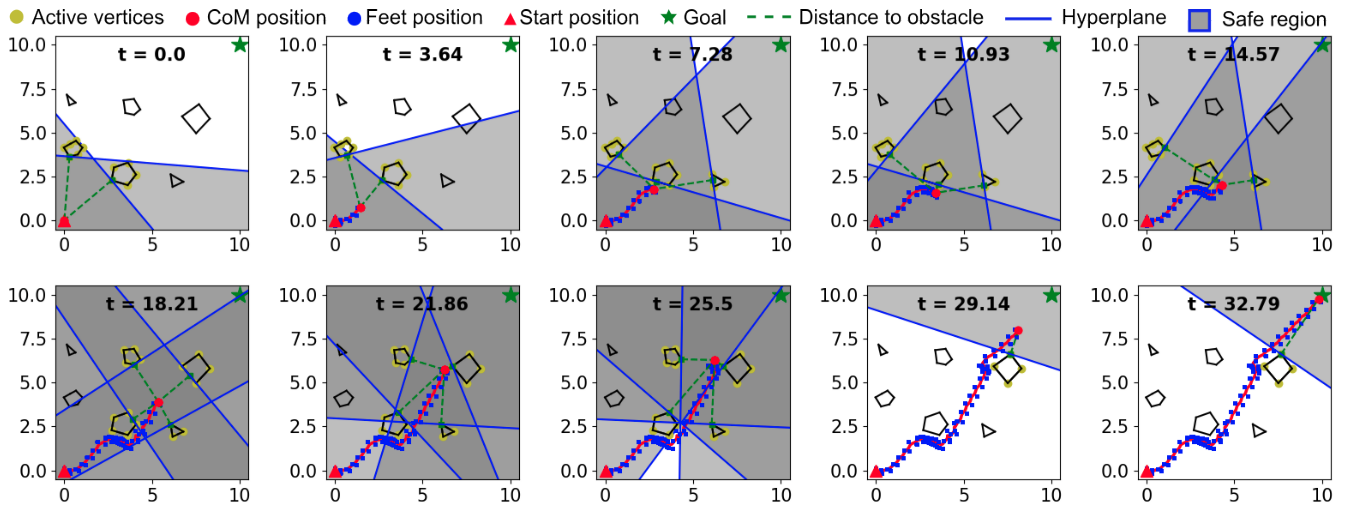

Fig. 8. Evolution of the active safe half-spaces during the robot motion using the safe MPC framework. The robot starts at (0,0) m and moves toward the goal at (10,10) m. At any given time, only obstacles within 4 meters of the robot are considered active and shown in yellow. The intersection of each corresponding half-space provides the safe region.

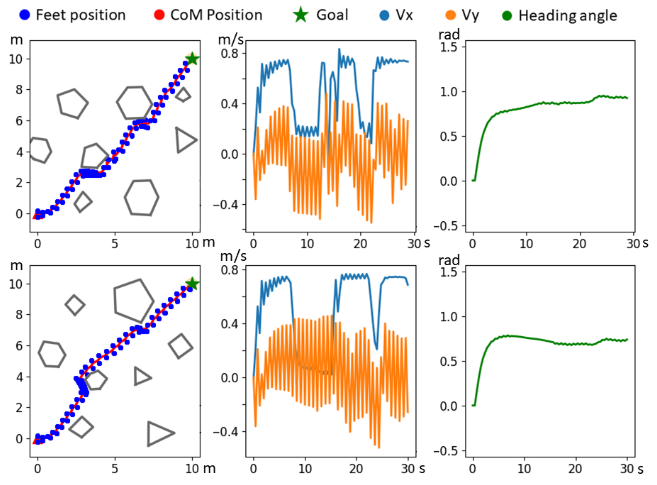

Fig. 9. The results of the global goal-oriented method in two different cases. The First column shows the robot foot displacements and CoM trajectories. The second column shows the longitudinal and lateral velocity $[m/s]$ change over time [s] during the navigation. The third column shows the heading angle changes $[rad]$ over time [s].

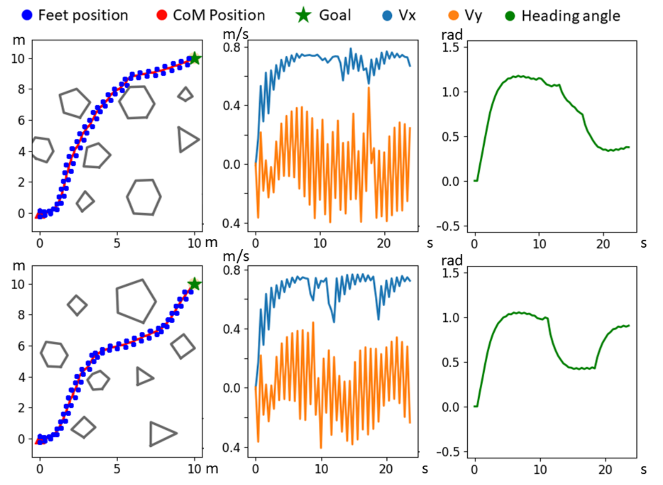

Fig. 10. The results of the subgoals-oriented method in two different cases. It shows a smoother path (the first column) of the robot, and the longitudinal velocity can keep a high velocity (the second column) compared to the global goal-oriented approach. The third column illustrates the more frequent changes in heading angle.

that guide the linearized LIP-MPC. The resulting robot CoM trajectories in the same two environments are shown in Fig. 10. This allows the robot to walk more often in the longitudinal direction, reaching the goal faster with 62 steps in 26 seconds. Compared to the previous method, the sub-goal-oriented approach produces a smoother trajectory with a higher average forward velocity and requires fewer steps.

V. Conclusion

This paper presents a linearized LIP-MPC structure for bipedal locomotion planning. The proposed method begins by determining the turning rate of each step and then obtains a foot placement sequence through a QP-based MPC. Moreover, we also introduce a novel LDCBF with a linear structure applicable to convex obstacles. The results demonstrate the reliability of our method for navigation in various obstacle environments and the feasibility of realizing real-time gait control of bipedal robots. We also discuss the benefits of using a global planner to generate sub-goals such that the robot can achieve more flexible navigation. Despite its effectiveness, there is room for optimizing the pre-computation of turning rates, to ensure safe and optimal steering in clustered environments. Our future work will focus on developing novel model-based and learning-based approaches for optimal steering and hardware realization of the proposed approaches in real-world experiments.

References

[1] D. L. Wight, E. G. Kubica, and D. W. L. Wang, "Introduction of the Foot Placement Estimator: A Dynamic Measure of Balance for Bipedal Robotics," *Journal of Computational and Nonlinear Dynamics*, vol. 3, no. 1, p. 011009, 11 2007.

[2] A.-C. Hildebrandt, M. Klischat, D. Wahrmann, R. Wittmann, F. Sygulla, P. Seiwald, D. Rixen, and T. Buschmann, "Real-time path planning in unknown environments for bipedal robots," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 1856–1863, 2017.

[3] J. Liu, M. Li, J.-K. Huang, and J. W. Grizzle, "Realtime safety control for bipedal robots to avoid multiple obstacles via clf-cbf constraints," *arXiv preprint arXiv:2301.01906*, 2023.

[4] M. Wermelinger, P. Fankhauser, R. Diethelm, P. Krüsi, R. Siegwart, and M. Hutter, "Navigation planning for legged robots in challenging terrain," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 1184–1189.

[5] S. Kuindersma, R. Deits, M. Fallon, A. Valenzuela, H. Dai, F. Permenter, T. Koolen, P. Marion, and R. Tedrake, "Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot," *Autonomous robots*, vol. 40, pp. 429–455, 2016.

[6] N. Sleumer and N. Tschichold-Gürmann, "Exact cell decomposition of arrangements used for path planning in robotics," *Technical Report/ETH Zurich, Department of Computer Science*, vol. 329, 1999.

[7] A. Gasparetto, P. Boscariol, A. Lanzutti, and R. Vidoni, "Path planning and trajectory planning algorithms: A general overview," *Motion and Operation Planning of Robotic Systems: Background and Practical Approaches*, pp. 3–27, 2015.

[8] H. Dai, A. Valenzuela, and R. Tedrake, "Whole-body motion planning with centroidal dynamics and full kinematics," in *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2014, pp. 295–302.

[9] M. Diehl, H. G. Bock, H. Diedam, and P.-B. Wieber, "Fast direct multiple shooting algorithms for optimal robot control," *Fast motions in biomechanics and robotics: optimization and feedback control*, pp. 65–93, 2006.

[10] K. Mombaur, "Using optimization to create self-stable human-like running," *Robotica*, vol. 27, no. 3, pp. 321–330, 2009.

[11] P.-B. Wieber, "Trajectory free linear model predictive control for stable walking in the presence of strong perturbations," in *2006 6th IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2006, pp. 137–142.

[12] G. García, R. Griffin, and J. Pratt, "Mpc-based locomotion control of bipedal robots with line-feet contact using centroidal dynamics," in *2020 IEEE-RAS 20th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2021, pp. 276–282.

[13] S. Teng, Y. Gong, J. W. Grizzle, and M. Ghaffari, "Toward safety-aware informative motion planning for legged robots," *arXiv preprint arXiv:2103.14252*, 2021.

[14] R. J. Griffin and A. Leonessa, "Model predictive control for dynamic footstep adjustment using the divergent component of motion," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1763–1768.

[15] C. Peng, V. Paredes, and A. Hereid, "Unified path and gait planning for safe bipedal robot navigation," *arXiv preprint arXiv:2403.17347*, 2024.

[16] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa, "The 3d linear inverted pendulum mode: A simple modeling for a biped walking pattern generation," in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180)*, vol. 1. IEEE, 2001, pp. 239–246.

[17] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3861–3876, 2016.

[18] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *2019 18th European control conference (ECC)*. IEEE, 2019, pp. 3420–3431.

[19] A. Manjunath and Q. Nguyen, "Safe and robust motion planning for dynamic robotics via control barrier functions," in *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 2122–2128.

[20] C. Peng, O. Donca, G. Castillo, and A. Hereid, "Safe bipedal path planning via control barrier functions for polynomial shape obstacles estimated using logistic regression," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 3649–3655.

[21] A. Agrawal and K. Sreenath, "Discrete control barrier functions for safety-critical control of discrete systems with application to bipedal robot navigation." in *Robotics: Science and Systems*, vol. 13. Cambridge, MA, USA, 2017, pp. 1–10.

[22] J. Zeng, B. Zhang, and K. Sreenath, "Safety-critical model predictive control with discrete-time control barrier function," in *2021 American Control Conference (ACC)*. IEEE, 2021, pp. 3882–3889.

[23] T. G. Molnar and A. D. Ames, "Composing control barrier functions for complex safety specifications," *IEEE Control Systems Letters*, 2023.

[24] J. Breeden and D. Panagou, "Compositions of multiple control barrier functions under input constraints," in *2023 American Control Conference (ACC)*. IEEE, 2023, pp. 3688–3695.