Wallbounce : Push wall to navigate with Contact-Implicit MPC

Xiaohan Liu[1*], Cunxi Dai[1*], John Z. Zhang[1], Arun Bishop[1], Zachary Manchester[1] and Ralph Hollis[1]

***Abstract*— In this work, we introduce a framework that enables highly maneuverable locomotion using non-periodic contacts. This task is challenging for traditional optimization and planning methods to handle due to difficulties in specifying contact mode sequences in real-time. To address this, we use a bi-level contact-implicit planner and hybrid model predictive controller to draft and execute a motion plan. We investigate how this method allows us to plan arm contact events on the shmoobot, a smaller ballbot, which uses an inverse mouse-ball drive to achieve dynamic balancing with a low number of actuators. Through multiple experiments we show how the arms allow for acceleration, deceleration and dynamic obstacle avoidance that are not achievable with the mouse-ball drive alone. This demonstrates how a holistic approach to locomotion can increase the control authority of unique robot morpohologies without additional hardware by leveraging robot arms that are typically used only for manipulation. Project website: https://cmushmoobot.github.io/Wallbounce**

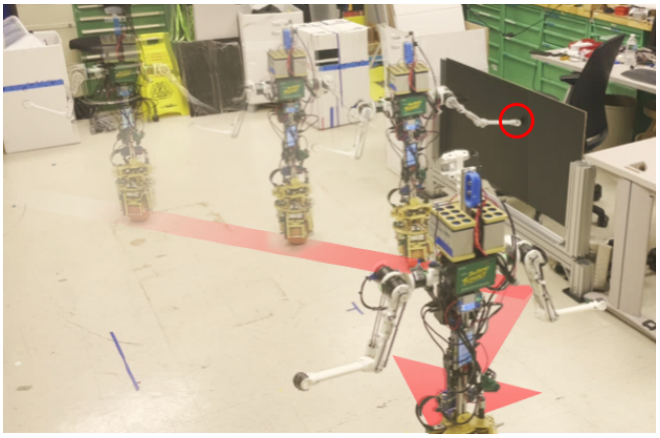


Fig. 1. Time-lapse picture of CMU shmoobot making a sharp turn by pushing wall.

I. INTRODUCTION

Humans and animals possess the incredible natural ability to leverage interactions between all parts of their bodies and the environment to achieve highly agile and dynamic locomotion behaviors. These interactions enable them to increase their control authority and locomotion capabilities beyond what can be achieved with legs alone. For example, parkour athletes use their hands to push off against walls and navigate around obstacles. Inspired by these capabilities, solving complicated locomotion tasks by leveraging diverse contact sources has been a long-standing challenge in robotics research.

Existing literature on multi-contact motion planning and control largely considers locomotion [1] [2] and manipulation [3] as separate research problems. In recent years, the rising interest in generalist robot agents has accelerated the design of robot hardware platforms equipped with both wheeled bases or legs for locomotion and arms for manipulation [4], [5], [6]. This new trend in robot morphology also introduces interesting research questions on how one can take advantage of the addition of robot arms during locomotion to augment the capabilities and robustness of the robot. Despite these increasingly mature human-like robot form factors, systems capable of solving challenging locomotion problems while leveraging upper-body capabilities remain understudied.

Integrating upper limb contacts into a Model Predictive Control (MPC) framework, however, is challenging. MPC often requires a predefined contact schedule for each contact point. For legged locomotion, where contacts are typically periodic, a contact schedule can be generated with heuristics [2], [6]. For upper limb contacts, determining the timing and duration of contact for other parts of the body is difficult. Most of the existing work rely on a hand-crafted contact schedule for upper limbs [7], [8]. Other works use search-based methods to find possible contact strategies [9], [10]. However, these methods are based on kinematics and quasi-static analysis and cannot capture the dynamic effect of the robot.

In this paper, we investigate using end effector contact on the CMU shmoobot (a smaller CMU ballbot [11]) to enhance its balance, locomotion, and navigation capabilities. Shmoobot uses a single ball drive to achieve dynamic balancing with a low number of actuators while remaining maneuverable in tight human spaces. However, this unique morphology limits how quickly Shmoobot can change its momentum, making it less robust when encountering unexpected obstacles while moving or experiencing large disturbances. We explore how we can use the arms the robot would typically use for manipulation to address these limitations during locomotion, increasing shmoobot's robustness and reactivity with no additional hardware.

We propose a bi-level Model-Predictive Control (MPC) framework that enables our robot to discover and utilize upper limb contacts during locomotion. At the higher level, we use contact-implicit optimization to identify potential contact schedules. Then, at a lower level, we deploy a hybrid trajectory optimization with this fixed contact schedule to generate smooth, feasible motion plans. Finally, we implement this framework on the CMU shmoobot platform and demonstrate its capabilities through several hardware

*Equal contribution.

[1]Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA, {xiaohan5, cunxid, ziyangz3, arunbish, zmanches, rhollis}@cs.cmu.edu

experiments.

Our specific contributions are:

1) A bi-level MPC framework that can reason about acyclic contacts and leverage upper limb contacts in locomotion.
2) Deployment and evaluation of the proposed framework on a novel bi-manual service robot that balances on a ball.
3) Experiments demonstrating how upper limb contacts can effectively assist in robot locomotion.

II. Related Works

A. Locomotion With Upper Limb Contacts

Like humans, human-like robots can use their upper limbs to assist with locomotion. Research [12] explored how a humanoid robot can make contact with a wall to prevent falling, while [7], [10] demonstrated how arm contact can help robots traverse challenging terrains. More related research studied how robots can gain acceleration by making contact with the environment. Reference [13] focuses on transferring human wall-pushing skills directly to robots. In [14], the researcher developed a reflex-based controller that can control the moving direction of a robot after contact with a wall. Our work focuses on an optimization-based framework that enables the CMU shmoobot to autonomously leverage upper limb contact without pre-specification during locomotion and navigation.

B. Contact-Implicit Optimization

Optimization-based algorithms can be a powerful tool when planning over contact mode schedules and timings. In particular, contact-implicit optimization (CIO) [15], [16] does not require predefined contact timings or locations, allowing the algorithm to explore different contact patterns. One common method for solving CIO is to formulate contact dynamics as complementarity problems [17]. Research [18] treats the contact dynamics as constraints and solves them with the direct collocation method. A higher-order collocation method [15], [19], solves the problem using HTO to refine the solution for faster convergence. However, due to the non-smooth nature of the contact dynamics, these methods are numerically unstable and take a long time (minutes to hours) to converge [20], making them unpractical during deployment in online MPC settings without specialized solvers [21]. Another approach of solving CIO is by using compliant contact models. Works [22], [23], use a continuous contact flag to control the allowed contact force. Other works [24], [25], use a nonlinear spring-damper system to model contact and solve the CIO problem with DDP-based methods. Compliant contact models make fast, real time CIO possible. However, soft contact models often introduce physical artifacts like force at a distance, which makes the planned trajectory hard to track for the hardware.

III. Background

A. Platform Description

The CMU shmoobot (Fig. 2(a)) is a 1.2 m tall robot that balances on a ball wheel. The robot has a pair of 3-DOF torque-controllable arms mounted onto its body. The ball is actuated by a four-motor Inverse Mouse-Ball Drive mechanism (IMBD) [11]. A pair of actuated opposing rollers drive the ball in each of the two orthogonal motion directions, which allows omnidirectional motion on the floor. A slip ring assembly and 5th actuator allows unlimited yaw rotation of the body. The model makes the following assumptions: (i) there is no slip between the ball and the floor; and (ii) the ball is always in contact with the floor.

B. Symbol and Notations

The body and world coordinate systems are defined in Fig. 2. Quantities in the body frame have a left subscript B. Other quantities are in the world coordinate system. Vectors are bold and lowercase ($\mathbf{a}$, $\boldsymbol{\omega}$), matrices are uppercase (A, Ω), scalars are lowercase and italicized (a, ω).

The operator $[\mathbf{v}]_\times$ converts a vector $\mathbf{v} = [v_1; v_2; v_3] \in \mathbb{R}^3$ into a skew-symmetric 'cross product matrix':

$$[\mathbf{v}]_\times = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}, \tag{1}$$

and we have $\mathbf{v} \times \mathbf{x} = [\mathbf{v}]_\times \mathbf{x}$.

C. MPC overview

Consider that we have an optimal control problem with I modes:

$$\begin{cases} \min\limits_{\mathbf{u}(.)\ x(.)} \ \sum_i \Phi_i(\mathbf{x}(t_{i+1})) + \int_{t_i}^{t_{i+1}} l_i(\mathbf{x}(t), \mathbf{u}(t), t)\, dt \\ \text{s.t. } \mathbf{x}(t_0) = \mathbf{x}_0 & (2\text{ - }1) \\ \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) & (2\text{ - }2) \\ \mathbf{g}_i(\mathbf{x}(t), \mathbf{u}(t), t) = \mathbf{0} & (2\text{ - }3) \\ \mathbf{h}_i(\mathbf{x}(t), \mathbf{u}(t), t) \geq \mathbf{0} & (2\text{ - }4) \\ \text{for } t_i < t < t_{i+1} \text{ and } i \in \{0, 1, \cdots, I-1\} \end{cases}. \tag{2}$$

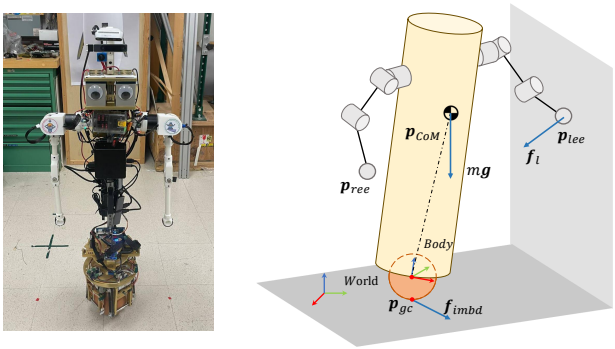
Here, (2-1) is the initial state constraint; (2-2) is the system dynamics constraint; (2-3) and (2-4) are equality and inequality constraints respectively. In our formulation, the active constraints vary as the mode of the system changes. An MPC controller recurrently solves this optimal control problem and searches for an optimal state input trajectory that minimizes the overall stage cost.

IV. System Modeling

A. Coordinate Definition

Since the ball is always in contact with the ground, we have a set of position constraints $p_z^{ball} = r_{ball}, v_z^{ball} = 0$ and $a_z^{ball} = 0$. The system dynamics is modeled with a set of minimal coordinates with implicit dynamics constraints.

As shown in Fig. 2(b), the origin of the robot base frame is defined at the center of the ball wheel. The general coordinate



(a) (b)

Fig. 2. (a) CMU shmoobot. (b) Coordinate systems of Shmoobot.

of the base is then defined as $\mathbf{q}_b = [p_x^{ball}, p_y^{ball}, \psi, \theta, \phi]^T$, where p_x^{ball}, p_y^{ball} are position, ψ, θ, ϕ are ZYX Euler angles of body orientation.

B. Simplified Robot Dynamics

We use single rigid body dynamics to model the shmoobot system. We neglect the mass of the arms and only consider the dynamic effect of the robot body.

Since the base frame is not defined at the center of mass, we need to consider the dynamics coupling between angular and linear terms. From Newton-Euler equations we have:

$$\begin{aligned} \sum \mathbf{f}^i &= m\dot{\mathbf{v}} + m[\mathbf{r}_{com}]_\times \alpha - m[\omega]_\times [\omega]_\times \mathbf{r}_{CoM}, \\ \sum \tau^i + \sum [\mathbf{r}_f^i]_\times \mathbf{f}^i &= \dot{\mathbf{l}}_r - [\mathbf{r}_{com}]_\times m\mathbf{a} + [\omega]_\times I_{inertia}\omega, \end{aligned} \tag{3}$$

where m is the body mass, $I_{inertia}$ is the inertia matrix, $\mathbf{v}$ is the linear velocity; and $\mathbf{l_r}$ is the angular momentum. $\mathbf{f^i}$ and $\tau^\mathbf{i}$ are applied external force and torque respectively. $\mathbf{r}_{CoM}$ is the position vector from the origin of the base frame to the center of mass (CoM). Similarly, $\mathbf{r}_f^i$ is the position vector from the origin to the point where external force $\mathbf{f^i}$ is applied.

In our case, we assume that the system has small angular velocity and angular acceleration. Also, since the ball always remains on the ground, we have $\mathbf{r}_z = 0, \mathbf{v}_z = 0$ and $\mathbf{a}_z = 0$. Then the linear acceleration can be expressed as $\mathbf{a} = \frac{[\sum \mathbf{f}_x^i, \sum \mathbf{f}_y^i, 0]^T}{m}$.

Then we have:

$$m\dot{\mathbf{v}} = \sum [\mathbf{f}_x^i, \mathbf{f}_y^i, 0]^T, \tag{4}$$

$$\dot{\mathbf{l}}_\mathbf{r} = \sum \tau^i + \sum [\mathbf{r}_f^i]_\times \mathbf{f}^i + [\mathbf{r}]_\times [\sum \mathbf{f}_\mathbf{x}^\mathbf{i}, \sum \mathbf{f}_\mathbf{y}^\mathbf{i}, 0]^T. \tag{5}$$

In our case, we only consider the contact force on the end effectors and the ball. Then, the base dynamics of the system can be written as:

$$m\dot{\mathbf{v}} = [\mathbf{f}_{IMBD,x}, \mathbf{f}_{IMBD,y}, 0]^T + \sum_{i=1}^{2} \left[\mathbf{f}_{c,x}^i, \mathbf{f}_{c,y}^i, 0\right]^T, \tag{6}$$

$$\dot{\mathbf{l}}_\mathbf{r} = \sum_{i=1}^{2} \mathbf{r}_{b,c}^i \times \mathbf{f}_c^i + \mathbf{r}_{b,CoM} \times m\mathbf{g} + \tau_{yaw} + \mathbf{r}_{b,CoM} \times m\dot{\mathbf{v}}. \tag{7}$$

Here, $\mathbf{f}_{IMBD}$ is the contact force on the ball. $\mathbf{f}_c^i$ is the contact force on the i-th end effector. $\mathbf{r}_{b,c_i}$ is the position of the i-th end effector w.r.t. the center of the ball; $\mathbf{r}_{b,CoM}$ is the position vector from ball center to the center of mass.

V. Contact Modeling

In this section, we present the contact models used by the controllers. The Contact-Implicit MPC uses a contact-invariant soft-contact model, while the Hybrid MPC uses a linear constraints-based set of contact modes.

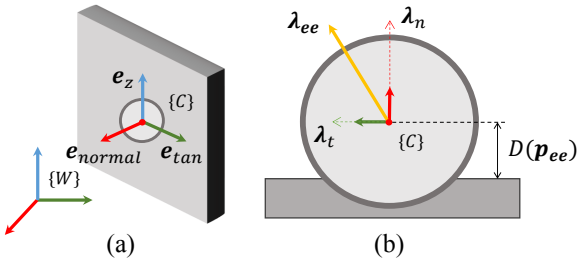
A. Contact Frame Definition

As shown in Fig. 3(a), the orientation of the contact frame is determined by the contact surface. The surface normal $\mathbf{e}_{\text{normal}}$ at a given point $\mathbf{p}$ can be acquired by computing the gradient of the Signed Distance Function (SDF) of the surface:

$$\mathbf{e}_{\text{normal}} = \nabla D(\mathbf{p}). \tag{8}$$

In this paper, we only consider vertical surfaces, so vector $\mathbf{e_z} = [0,0,1]^T$ is always tangent to the surface. Then, we can obtain the other tangent vector by taking the cross product:

$$\mathbf{e}_{tangent} = \mathbf{e}_{normal} \times \mathbf{e}_z. \tag{9}$$



(a) (b)

Fig. 3. (a) Definition of contact frame. (b) Schematic of contact.

B. Soft Contact Model

In contact-implicit MPC, we used a contact-invariant soft contact model. The normal part of the contact force is expressed as a nonlinear function of end-effector position:

$$\lambda_{normal} = f(D(\mathbf{p}_{ee})). \tag{10}$$

Here, $\mathbf{p}_{ee}$ is the position of the end effector, $D(\mathbf{p})$ is the signed distance function of the surface. $f(d)$ is a nonlinear scalar-valued function that increases rapidly when d is smaller than 0, and sticks to 0 when d is larger than 0. There are many choices of the activation function $f(d)$, here we pick

$$f(d) = 0.5 f_{max} \cdot tanh(-\alpha \cdot (d + \beta)) + 0.5 f_{max}, \tag{11}$$

where α and β controls the stiffness of the contact and f_{max} is the maximum allowed normal force. This model implicitly contains the information of contact force limit and is twice differentiable.

The end effector should always be outside the surface, so we have:

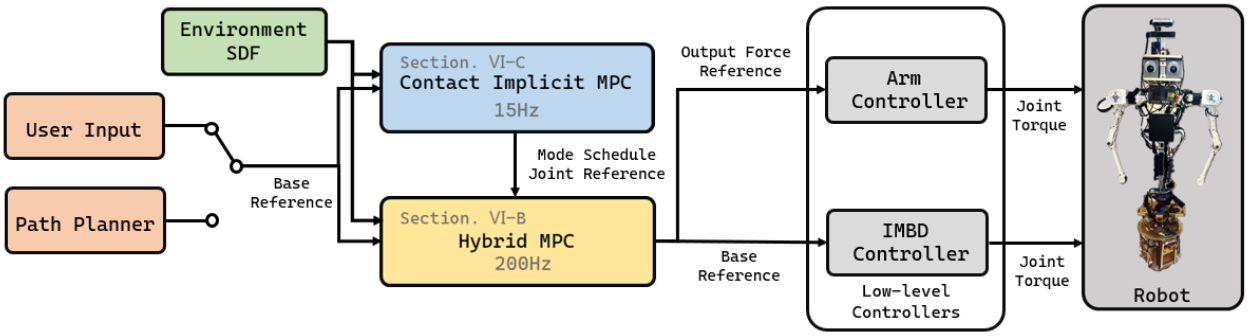


Fig. 4. Control framework diagram. A reference trajectory is first generated by a path planner or sent by the user. A bi-level MPC will calculate an optimal trajectory that tracks the reference. The environment SDF used by the controller needs to be recomputed. The upper level of the controller (the blue block) is a contact-implicit MPC which generates a draft of the motion plan with soft contact models. The lower level of the controller (the yellow block) is a hybrid MPC. It will extract a contact schedule from the motion plan, and refine the trajectory with hard contact models. The low level balancing controller and arm controller will then track the motion plan provided by the hybrid MPC.

$$D(\mathbf{p}_{ee}) \geq 0. \tag{12}$$

Finally, the end effector shouldn't slip on the wall when the contact force is nonzero. This gives a linear complementary constraint:

$$\lambda_{\mathrm{ee}} \dot{\mathbf{p}}_{\mathrm{ee}} = 0. \tag{13}$$

C. Hybrid Contact Model

For hybrid MPC, we use a hybrid contact model. We denote the set of closed contacts by C. Then, if an end effector is in contact, we have the following constraints:

$$\begin{cases} \dot{\mathbf{p}}^{\mathrm{i}}_{ee} = \mathbf{0} \\ D(\mathbf{p}^{\mathrm{i}}_{ee}) = \mathbf{0} \\ \lambda^{\mathrm{i}}_{normal} \cdot \mathbf{e}^{i}_{normal} > \mathbf{0} \\ -\mu\lambda^{i}_{normal} \leq \lambda^{i}_{tan} \leq \mu\lambda^{i}_{normal} \\ -\mu\lambda^{i}_{normal} \leq \lambda^{i}_{z} \leq \mu\lambda^{i}_{normal} \end{cases} \quad \text{if } c_i \in \mathrm{C}. \tag{14}$$

If the end effector is not in contact, we have the following constraints:

$$\begin{cases} \lambda^{\mathrm{i}}_{ee} = \mathbf{0} \\ D(\mathbf{p}^{\mathrm{i}}_{ee}) \geq \mathbf{0} \end{cases} \quad \text{if } c_i \in \overline{\mathrm{C}}. \tag{15}$$

The set of contacts, C, is be obtained from the optimized trajectory of the contact-implicit controller. This will be further discussed in Section. VI.

VI. System Design

In this section, we present our locomotion controller framework. The overall system structure is presented in Fig. 4.

A. Hybrid MPC

1) System Dynamics: The system states $\mathbf{x} \in \mathbb{R}^{16}$ and inputs $\mathbf{u} \in \mathbb{R}^{15}$ are defined as:

$$\mathbf{x} = \left[\mathbf{h}_b^T, \mathbf{q}_b^T, \mathbf{q}_j^T\right]^T, \mathbf{u} = \left[\mathbf{f}_{IMBD}^T, \mathbf{f}_c^T, \mathbf{v}_j^T\right]^T. \tag{16}$$

Here, $\mathbf{q}_b$ is the generalized coordinate of the base. $\mathbf{q}_j$ are the joint positions. $\mathbf{h}_b = \left[m\mathbf{v}^T, \mathbf{l}_r^T\right]^T \in \mathbb{R}^5$ is the collection of linear and angular momentum.

For input $\mathbf{u}$, $\mathbf{f}_{IMBD} = [\mathbf{f}_x, \mathbf{f}_y, \tau_z]$ is the contact force on the ball. Due to the unique property of the IMBD mechanism, there is almost no relative spinning along the z axis of the ball, and we can approximately have $\tau_z = \tau_{yaw}$. $\mathbf{f}_c = [\mathbf{f}_c^1, \mathbf{f}_c^2] \in \mathbb{R}^6$ is the contact force on the two end effectors. $\mathbf{v}_j^T$ are the joint velocities.

Then, from equations (7) we have:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} m\mathbf{v} \\ \mathbf{l_r} \\ \mathbf{q}_b \\ \mathbf{q}_j \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{2}\left[\mathbf{f}^i_{c,x}, \mathbf{f}^i_{c,y}\right]^T + [\mathbf{f}_{IMBD,x}, \mathbf{f}_{IMBD,y}]^T \\ \sum_{i=1}^{2}[\mathbf{r}^i_{b,c}]_\times \mathbf{f}^i_c + [\mathbf{r}_{b,CoM}]_\times m(\mathbf{g} + \dot{\mathbf{v}}) + \tau_{yaw} \\ A_b^{-1}(\mathbf{q}_b)\mathbf{h}_b \\ \mathbf{v}_j \end{bmatrix}. \tag{17}$$

Here, A_b is the centroidal momentum matrix which maps generalized velocities to centroidal momentum. Readers can refer to [26] for more details.

2) Constraints: The contact constraints (14), (15) are described in section V-C. An input limit constraint is also added.

3) Cost: The cost is a quadratic tracking cost to follow a given full state trajectory, including base pose, momentum, and nominal joint positions.

B. Contact-Implicit MPC

1) System Dynamics: The system states $\tilde{\mathbf{x}} \in \mathbb{R}^{16}$ and inputs $\tilde{\mathbf{u}} \in \mathbb{R}^{13}$ are defined as:

$$\tilde{\mathbf{x}} = \left[\mathbf{h}_b^T, \mathbf{q}_b^T, \mathbf{q}_j^T\right]^T, \tilde{\mathbf{u}} = \left[\mathbf{f}_c^T, \mathbf{v}_j^T, \alpha^T\right]^T.$$

Here, $\alpha = \left[\alpha^1_{\mathrm{tangent}}, \alpha^1_{\mathrm{z}}, \alpha^2_{\mathrm{tangent}}, \alpha^2_{\mathrm{z}}\right] \in \mathbb{R}^4$ are the auxiliary input variables that control the tangential parts of the contact force. The other parts of the state and input are consistent with the definitions in the hybrid MPC.

From equation (10) we have:

$$\lambda^i_{normal} = f(D(FK_i(\mathbf{q}_b, \mathbf{q}_j))). \tag{18}$$

Here, $FK_i(\mathbf{q}_b, \mathbf{q}_j)$ is the forward kinematics function that calculates the position of the i-th end effector in world frame.

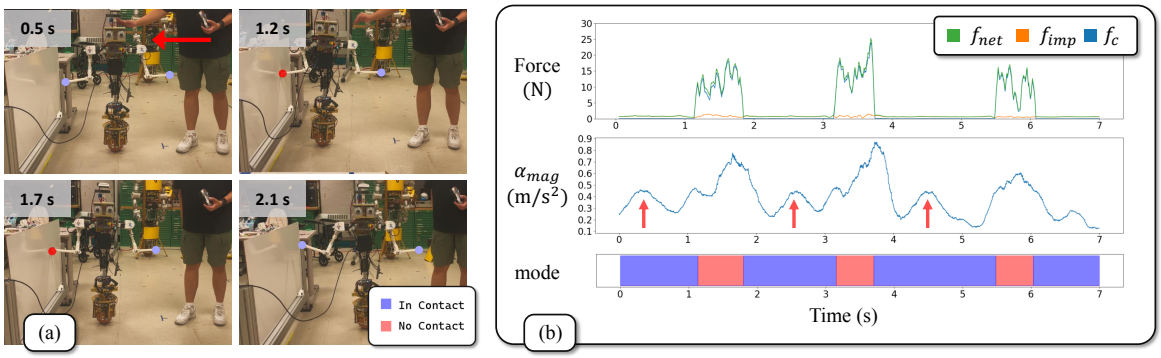


Fig. 5. (a) Key frames for the disturbance rejection experiment. The end effectors are highlighted with blue and red circles. A red circle indicates that the end effector is in contact. At T = 0.5 s, a human operator pushed the robot to the wall. The robot actively used its arm to recover from the disturbance. (b) Force output trajectory, acceleration trajectory and mode sequence of the robot.

λ^i_{normal} is the normal part of contact force on the end effector. The tangential parts of the contact force are controlled by the auxiliary variable α. We have:

$$\lambda^i_{tan} = \alpha^i_{tan} \cdot \lambda^i_{normal}, \tag{19}$$

$$\lambda^i_z = \alpha^i_z \cdot \lambda^i_{normal}. \tag{20}$$

Then, the contact force $\mathbf{f}^i_c$ can be expressed as:

$$\mathbf{f}^i_c = \lambda^i_{\text{normal}} \mathbf{e}^i_{\text{normal}} + \lambda^i_{\text{tangent}} \mathbf{e}^i_{\text{tangent}} + \lambda^i_z \mathbf{e}^i_z. \tag{21}$$

Substitute (21) into (17), and we can have the system dynamics.

2) Constraints: The end effector constraints (12), (13) are described in section V-B. Additionally, we have a friction cone constraint $-\mu \leq \alpha \leq \mu$. All these constraints are handled with penalty methods and treated as parts of the cost function.

3) Cost: Like VI-A, the system cost is a quadratic tracking cost to follow a full state trajectory.

C. Contact Schedule Generation

The contact schedule used by the Hybrid MPC is generated by thresholding the outputs from contact-implicit MPC. With Eqn. 9, we can obtain the normal force trajectories $\lambda^i_{normal}(t)$. The algorithm goes through the force trajectories and add the end effector with $\lambda^i_{normal}(t) > \lambda_{threshold}$ to the active contact set $C(t)$. Here, we use $\lambda_{threshold} = 5$ N. Contacts last shorter than 0.5 second will be neglected.

D. Body control

The Body Controller consists of two parts: a balancing controller and a yaw controller. The balancing controller is a PID controller cascaded with a PD controller that tracks the desired body leaning angle and velocity. The yaw controller is a PID controller that tracks the yaw orientation in world frame.

E. Arm control

The arm controller tracks the end effector position and output force at the same time. The reference end effector position is given by:

$$_B\tilde{\mathbf{p}}^{\mathbf{i}} = {}_BFK_i(\tilde{\mathbf{q}}_j). \tag{22}$$

$_B\tilde{\mathbf{p}}^{\mathbf{i}}_{\mathbf{ee}}$ is the reference end effector position in body frame, and $\tilde{\mathbf{q}}_j$ is the reference joint position. We use a task space impedance controller to track the end effector position. The control law is:

$$_B\mathbf{f}_{imp} = \mathbf{K}_p(_B\tilde{\mathbf{p}}^{\mathbf{i}} - {}_B\mathbf{p}^i) + \mathbf{K}_d(0 - {}_B\dot{\mathbf{p}}^i), \tag{23}$$

where $\mathbf{K}_p$, $\mathbf{K}_d$ are positive definite gain matrices.

Then, the control law used to compute joint torques for the i-th arm is:

$$\tau^{\mathbf{i}} = \mathbf{J}^{iT}\left[{}_B\mathbf{f}^i_{imp} + {}_B\mathbf{f}^i_c\right], \tag{24}$$

where $\mathbf{J}^i$ is the Jacobian matrix of i-th end effector, and $\mathbf{K}_p$, $\mathbf{K}_d$ are positive definite gain matrices.

VII. Experiments

A. Controller Implementation

The nonlinear optimal control problem is implemented in C++ and solved by the SLQ solver provided by the ETH OCS2 [27] toolbox. The software uses the Eigen3 linear algebra library [28]. All the dynamics and constraints are implemented with CppAD and can be auto differentiated.

The controller is deployed on a robot onboard computer with Intel Core i7-1165G7 CPU. The state estimation of the base pose is provided by a T265 tracking camera. Both MPCs have a 1 s planning horizon. The hybrid MPC requires 1.98 ms solve time on average, while the contact-implicit MPC requires 10.99 ms on average. However, to save computation power for other components (navigation, obstacle detection) and ensure software stability, the contact-implicit MPC is restricted to updates at 15 Hz, and the low-level hybrid MPC updates at 200 Hz.

B. Contact Model

In this experiment, we compared the end-effector and force output trajectories generated by a single-level (using only CI-MPC) and bi-level (using Hybrid MPC for trajectory correction) MPC. The time between 0.35 s and 1.7 s was identified as a valid contact period. The Hybrid MPC further refined the trajectory based on this constraint. As shown in Fig. 6, the top plot is the end-effector velocity. During contact, the Hybrid MPC keeps the end-effector velocity close to zero, while the raw output from CI-MPC exhibits

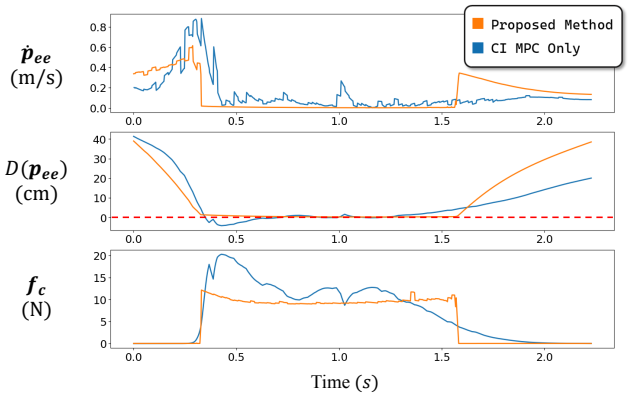


Fig. 6. The planned end effector velocity (top), position (middle), and force (bottom) trajectories from the proposed bi-level MPC (orange) and CI MPC (blue).

noticeable relative sliding. The middle plot displays the distance between the end-effector and the wall. The raw output from CI-MPC has a wall penetration of up to 5 cm, whereas the corrected end-effector trajectory stays precisely on the wall. The bottom plot shows the force output trajectory during this period. Comparing it with the position trajectory reveals that CI-MPC begins outputting force even before the end-effector makes contact with the wall.

It is important to note that the infeasible trajectories produced by the soft CI-MPC can be reduced by tuning contact model parameters. However, to guarantee feasible motion plans online, we find it necessary to apply the hybrid MPC in our framework.

C. Disturbance Rejection

We first test the robot's ability to reject external disturbances by pushing against the wall. The robot was placed 0.5 meters away from the wall and commanded to stay in place. During the test, an operator pushes the robot toward the wall. The time-lapse sequence of the experiment is shown in Fig. 5(a).

As can be seen, when the robot approached the wall, it stretched its arm and used its end effector to push itself back to its original position. Fig. 5(b) shows the forces, acceleration, and planned contact sequence from three consecutive trials. The first plot shows the expected force output of the robot's right arm during the experiment. It can be observed that the output quickly reached approximately 20 N when contact occurred. The dominant part of the force came from the output of the hybrid MPC. The task-space impedance controller only has a limited contribution. The second plot shows the magnitude of the robot's acceleration during the process. The peak acceleration marked with red arrows was caused by the disturbance applied by the operator. While the higher peaks resulted from the robot pushing against the wall. After contact ended, the robot's acceleration rapidly decreased.

D. Obstacle Avoidance

In this experiment, we demonstrated the robot's capability to quickly avoid obstacles by pushing against a wall. We use

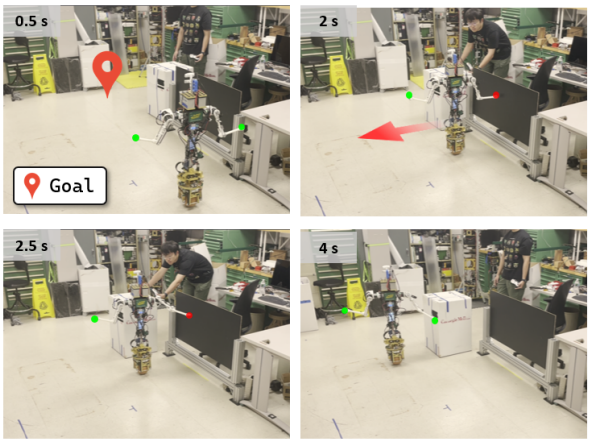


Fig. 7. Key frames for the obstacle avoidance experiment. The red highlighted point is the goal location. At T = 2 s, a obstacle occurred right in front of the robot. The robot pushed the wall to avoid it.

an external path planner to provide a reference trajectory to the MPC controller. When an obstacle is detected, the planner will quickly generate a collision-free trajectory. The robot uses a Realsense D435i depth camera to detect obstacles. As shown in Fig. 7, the robot was commanded to move to a point 3 m ahead. An operator quickly pushed a box to block the robot on its way. Upon detecting the obstacle, our robot successfully pushed against the wall and avoided the obstacle. Note that the acceleration required to avoid the sudden obstacle, in this case, is challenging and dangerous with the IMBD drive wheel alone, making the upper-limb contact force necessary to successfully avoid the obstacle.

VIII. Conclusion

In this paper, we proposed a bi-level MPC framework on the CMU shmoobot platform that can utilize non-periodic contacts without predefined contact sequences in locomotion tasks. The proposed framework can provide an optimal trajectory that satisfies hard contact constraints at real-time rates. We validated our approach on a CMU shmoobot, a dual-arm mobile base robot that balances on a ball. The robot shows the ability to utilize contacts to quickly accelerate, decelerate, and avoid dynamic obstacles. The proposed framework can also be deployed on other robotic systems with manipulators. However, in this paper, we only considered contact between two end effectors and vertical walls. Future work could incorporate advanced collision detection libraries to account for whole-body contact. However, we only considered vertical walls. Future work could include surfaces with different orientations, and incorporate legged locomotion into the same framework. Moreover, the algorithm's performance on long-horizon tasks needs to be investigated.

ACKNOWLEDGMENT

This work was supported in part by NSF grant CNS-1629757 and Amazon. The authors would like to thank Zhongyu Li, Guanqi He, and Yuntian Zhao for the valuable discussions.

REFERENCES

[1] R. Grandia, F. Jenelten, S. Yang, F. Farshidian, and M. Hutter, "Perceptive locomotion through nonlinear model-predictive control," *IEEE Transactions on Robotics*, vol. 39, no. 5, pp. 3402–3421, 2023.

[2] J. Di Carlo, P. M. Wensing, B. Katz, G. Bledt, and S. Kim, "Dynamic locomotion in the mit cheetah 3 through convex model-predictive control," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1–9.

[3] M. T. Mason, J. K. Salisbury, and J. K. Parker, "Robot Hands and the Mechanics of Manipulation," *Journal of Dynamic Systems, Measurement, and Control*, vol. 111, no. 1, pp. 119–119, 03 1989. [Online]. Available: https://doi.org/10.1115/1.3153010

[4] R. Shu and R. Hollis, "Development of a humanoid dual arm system for a single spherical wheeled balancing mobile robot," in *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, 2019, pp. 499–504.

[5] Z. Fu, T. Z. Zhao, and C. Finn, "Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation," in *Conference on Robot Learning (CoRL)*, 2024.

[6] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch, R. Diethelm, S. Bachmann, A. Melzer, and M. Hoepflinger, "Anymal - a highly mobile and dynamic quadrupedal robot," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 38–44.

[7] M. Murooka, M. Morisawa, and F. Kanehiro, "Centroidal trajectory generation and stabilization based on preview control for humanoid multi-contact motion," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 8225–8232, 2022.

[8] J.-P. Sleiman, F. Farshidian, M. V. Minniti, and M. Hutter, "A unified mpc framework for whole-body dynamic locomotion and manipulation," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4688–4695, 2021.

[9] E. Jelavic, K. Qu, F. Farshidian, and M. Hutter, "Lstp: Long short-term motion planning for legged and legged-wheeled systems," *IEEE Transactions on Robotics*, vol. 39, no. 6, pp. 4190–4210, 2023.

[10] S. Tonneau, A. Del Prete, J. Pettré, C. Park, D. Manocha, and N. Mansard, "An efficient acyclic contact planner for multiped robots," *IEEE Transactions on Robotics*, vol. 34, no. 3, pp. 586–601, 2018.

[11] U. Nagarajan, G. Kantor, and R. Hollis, "The ballbot: An omnidirectional balancing mobile robot," *The International Journal of Robotics Research*, vol. 33, no. 6, pp. 917–930, 2014.

[12] S. Wang and K. Hauser, "Realization of a real-time optimal control strategy to stabilize a falling humanoid robot with hand contact," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 3092–3098. [Online]. Available: https://ieeexplore.ieee.org/document/8460500/

[13] C. Bauer and N. S. Pollard, "Human-informed robot agility: Understanding human pushing interactions for skill transfer to humanoids," in *2023 IEEE-RAS 22nd International Conference on Humanoid Robots (Humanoids)*, 2023, pp. 1–8.

[14] C. Bauer, D. Bauer, A. Allaire, C. G. Atkeson, and N. Pollard, "Learning to navigate by pushing," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 171–177.

[15] Z. Manchester, N. Doshi, R. J. Wood, and S. Kuindersma, "Contact-implicit trajectory optimization using variational integrators," *The International Journal of Robotics Research*, vol. 38, no. 12-13, pp. 1463–1476, 2019.

[16] M. Posa, C. Cantu, and R. Tedrake, "A direct method for trajectory optimization of rigid bodies through contact," *The International Journal of Robotics Research*, vol. 33, pp. 69 – 81, 2014. [Online]. Available: https://api.semanticscholar.org/CorpusID:6532910

[17] M. Anitescu and F. A. Potra, "Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems," *Nonlinear Dynamics*, vol. 14, pp. 231–247, 1997. [Online]. Available: https://api.semanticscholar.org/CorpusID:7393680

[18] C. Mastalli, I. Havoutis, M. Focchi, D. G. Caldwell, and C. Semini, "Hierarchical planning of dynamic movements without scheduled contact sequences," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 4636–4641.

[19] M. R. Turski, J. Norby, and A. M. Johnson, "Staged contact optimization: Combining contact-implicit and multi-phase hybrid trajectory optimization," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 2376–2383.

[20] J. Z. Zhang, S. Yang, G. Yang, A. L. Bishop, S. Gurumurthy, D. Ramanan, and Z. Manchester, "Slomo: A general system for legged robot motion imitation from casual videos," *IEEE Robotics and Automation Letters*, vol. 8, no. 11, pp. 7154–7161, 2023.

[21] S. Le Cleac'h, T. A. Howell, S. Yang, C.-Y. Lee, J. Zhang, A. Bishop, M. Schwager, and Z. Manchester, "Fast contact-implicit model predictive control," *IEEE Transactions on Robotics*, vol. 40, pp. 1617–1629, 2024.

[22] I. Mordatch, Z. Popović, and E. Todorov, "Contact-invariant optimization for hand manipulation," in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA '12. Goslar, DEU: Eurographics Association, 2012, p. 137–144.

[23] I. Mordatch, E. Todorov, and Z. Popović, "Discovery of complex behaviors through contact-invariant optimization," *ACM Trans. Graph.*, vol. 31, no. 4, jul 2012. [Online]. Available: https://doi.org/10.1145/2185520.2185539

[24] M. Neunert, M. Stäuble, M. Giftthaler, C. D. Bellicoso, J. Carius, C. Gehring, M. Hutter, and J. Buchli, "Whole-body nonlinear model predictive control through contacts for quadrupeds," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1458–1465, 2018.

[25] M. Neunert, F. Farshidian, A. W. Winkler, and J. Buchli, "Trajectory optimization through contacts and automatic gait discovery for quadrupeds," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1502–1509, 2017.

[26] D. E. Orin, A. Goswami, and S.-H. Lee, "Centroidal dynamics of a humanoid robot," *Autonomous Robots*, vol. 35, pp. 161–176, 2013. [Online]. Available: https://api.semanticscholar.org/CorpusID:15473645

[27] F. Farshidian *et al.*, "OCS2: An open source library for optimal control of switched systems," [Online]. Available: https://github.com/leggedrobotics/ocs2.

[28] G. Guennebaud, B. Jacob *et al.*, "Eigen v3," http://eigen.tuxfamily.org, 2010.