From Object Pushing to Self Pushing: Whole-Body Control for Ballbots

Xiaohan Liu

CMU-RI-TR-25-62

June 25, 2025

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics.

To the Machine Spirit, may its circuits forever hum in harmony.

Abstract

Dynamically stable mobile robots, like ballbots, are agile but highly sensitive to manipulator-induced disturbances. This feature makes loco-manipulation of dynamically stable mobile robots a tightly coupled whole-body control problem. This thesis investigates how manipulators can be leveraged not only for interacting with objects, but also for locomotion.

First, we investigate effective wheelchair maneuvering using a dynamically balancing mobile manipulator. Wheelchairs are a type of nonholonomic cart system, maneuvering such systems with mobile manipulators (MM) is challenging mostly due to the following reasons: 1) These systems feature nonholonomic constraints and considerably varying inertial parameters that require online identification and adaptation. 2) These systems are widely used in human-centered environments, which demand the MM to operate in potentially crowded spaces while ensuring compliance for safe physical human-robot interaction (pHRI). We propose a control framework that plans whole-body motion based on quasi-static analysis to maneuver heavy nonholonomic carts while maintaining overall compliance. We validated our approach experimentally by maneuvering a wheelchair with a bimanual mobile manipulator, the CMU ballbot. The experiments demonstrate the proposed framework is able to track desired wheelchair velocity with loads varying from 11.8 kg to 79.4 kg at a maximum linear velocity of 0.45 m/s and angular velocity of 0.3 rad/s. Furthermore, we verified that the proposed method can generate human-like motion smoothness of the wheelchair while ensuring safe interactions with the environment.

Second, we explore how upper limb contacts can assist robot locomotion. Utilizing upper limb contacts 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.

Acknowledgments

I would like to express my sincere gratitude to my advisor, Prof. Ralph Hollis, for giving me the opportunity to work on Ballbot, Shmoo, and maglev haptics. It has been an honor to work with you. I've learned so much—not just about robotics, but also about life.

I also want to thank my committee members, Prof. Zachary Manchester and Prof. Changliu Liu, for their insightful feedback and patient mentorship. I got to know you through your classes. The courses on optimal control and safe robotics offered invaluable insights and greatly supported my research.

To Cunxi—you are the most important collaborator and friend I've had so far in my life. We've been working together for six years, and it was you who introduced me to Ralph. Without you, my life at CMU would have been completely different. (And hey, I'm using your actual first name here, not calling you Jimmy, because you probably won't get to use it much in the future LOL.)

I'm also grateful to John Z. Zhang and Arun Bishop, who helped with the wall-pushing paper. Without their support, the paper wouldn't have turned out as well as it did. I'd also like to thank Roberto Shu, who has always been friendly and willing to help. His assistance and guidance were crucial—without him, we wouldn't have been able to turn on Ballbot. What he accomplished years ago laid the foundation for our work.

To my parents: you have always been my biggest supporters. You stood behind almost every decision I made and gave me the chance to follow my dreams and ambition. I'm always proud to be your son.

I feel incredibly lucky to have had the chance to do research at RI for two years. Now, as I transition from academia to industry, a new world awaits. The experiences I've had will definitely help me in the future—and maybe one day, I'll return for a Ph.D.

Funding

The works discussed in this thesis were supported in part by NSF grant CNS-1629757 and Amazon.

Contents

Chapter 1

Introduction

Robots can be broadly classified by their mobility and stability characteristics. Fixed-base manipulators have an immovable base, and thus are unaffected by reaction forces from their manipulators. In contrast, mobile robots must manage base disturbances. Statically stable mobile robots (such as a four-wheeled platform with a low center of gravity)[1][9][14] can maintain stability through a wide support polygon and can passively resist tipping to some extent. Dynamically stable mobile robots[27][49] rely on active balancing to stay upright. This dynamic stability allows the robots to be agile, but also makes them highly sensitive to disturbances. Even modest forces or motions on the robot manipulator can induce significant base reactions.

This sensitivity makes locomanipulation tasks on dynamically stable platforms more challenging. Any external force exerted by the manipulator, for example, pushing a door or lifting an object, will be transmitted through the robot structure and can perturb its overall balance. Balance, locomotion, and manipulation are highly coupled in such systems[30]. Therefore, loco-manipulation on dynamically balancing robots is a whole-body coordination problem, where every manipulation action has immediate consequences on stability and navigation.

The inherent coupling between manipulation and locomotion, though potentially problematic if not handled properly, can be utilized to improve mobility. A dynamically stable robot can use its manipulator not only to interact with objects but also to enhance its own locomotion. Recent research has begun to leverage this idea. By deliberately making contact with the environment, a robot manipulator can generate additional forces that support movement and balance [49]. This capability offers a distinct advantage over fixed-base and statically stable robots.

This thesis builds on that insight : moving the focus from object pushing to self-pushing. We develop and experimentally validate whole-body controllers that (i) reject manipulator-induced disturbances to maintain balance, and (ii) deliberately exploit manipulator forces to enhance ballot's locomotion capabilities. The main structure of the paper is summarized as follows:

Chapter. 2 introduces a whole-body control framework for wheelchair maneuvering with ballbot. Chapter. 3 introduces a whole body controller framework that can utilize upper limb contacts in locomotion tasks with contact implicit MPC. Chapter. 4 concludes the thesis with a summary of the contributions and presents potential directions for future research.

Chapter 2

Object Pushing : Wheelchair Manipulation with ballbot

2.1 Abstract

In this chapter, we present a control framework to effectively maneuver wheelchairs with a dynamically stable mobile manipulator. Wheelchairs are a type of nonholonomic cart system, maneuvering such systems with mobile manipulators (MM) is challenging mostly due to the following reasons: 1) These systems feature nonholonomic constraints and considerably varying inertial parameters that require online identification and adaptation. 2) These systems are widely used in human-centered environments, which demand the MM to operate in potentially crowded spaces while ensuring compliance for safe physical human-robot interaction (pHRI). We propose a control framework that plans whole-body motion based on quasi-static analysis to maneuver heavy nonholonomic carts while maintaining overall compliance. We validated our approach experimentally by maneuvering a wheelchair with a bimanual mobile manipulator, the CMU ballbot. The experiments demonstrate the proposed framework is able to track desired wheelchair velocity with loads varying from 11.8 kg to 79.4 kg at a maximum linear velocity of 0.45 m/s and angular velocity of 0.3 rad/s. Furthermore, we verified that the proposed method can generate human-like motion smoothness of the wheelchair while ensuring safe interactions with the environment.

2.2 Introduction

Recent advancements in mobile manipulators have greatly enhanced their utility for assisting humans. A critical aspect is the ability of MMs to operate nonholonomic cart systems, such as shopping carts, luggage carts or hospital beds. These usually have one or two caster wheels and a set of fixed wheels that introduce nonholonomic constraints to their motion. They are essential for transporting cargo or people in environments centered around human activities, including supermarkets, hospitals, and construction sites. Among these, pushing a wheelchair to a specified location is vital for individuals with mobility issues, particularly in places like public transport and healthcare centers. Using robots for wheelchair assistance can significantly reduce a caregiver's burden and enhance mobility for wheelchair users. When pushing the wheelchair, the robot needs to maneuver the wheelchair with smooth motion while ensuring safe interaction with others surrounding it.

Previous work has been done in maneuvering nonholonomic systems with various platforms. [13, 31, 47] used bipedal humanoid robots to maneuver heavy objects or carts by computing zero momentum points (ZMP) of interaction with the object and foot placement planning. [33] studied how to change the robot's posture and body leaning angle based on kinematics to maximize force exertion. While being able to exert large forces via configuration change, bipedal robots face challenges from the inherent discreteness in their locomotion. This can lead to non-smooth movements of the object being pushed if not carefully controlled, undesirable for wheelchair pushing as it may compromise passenger comfort and has reduced energy efficiency on flat surfaces compared to wheeled robots.

Multi-wheeled mobile base robots are also widely used in cart-pushing tasks. [35, 37] studied path planning methods for bimanual mobile-based robots when navigating with a cart, which assumed that the cart can be reoriented by the MM as needed. This does not hold for heavy carts such as wheelchairs occupied by a person. [1, 22, 23] proposed trajectory-tracking controllers for non-holonomic cart systems that utilize the change of grasping point on the cart to maneuver with a single robot arm. [10] proposed an improved impedance controller for cart-pushing tasks to enhance wheelchair stability and ride quality. Results in these studies indicate that multi-wheeled mobile base robots are capable of producing smoother movements

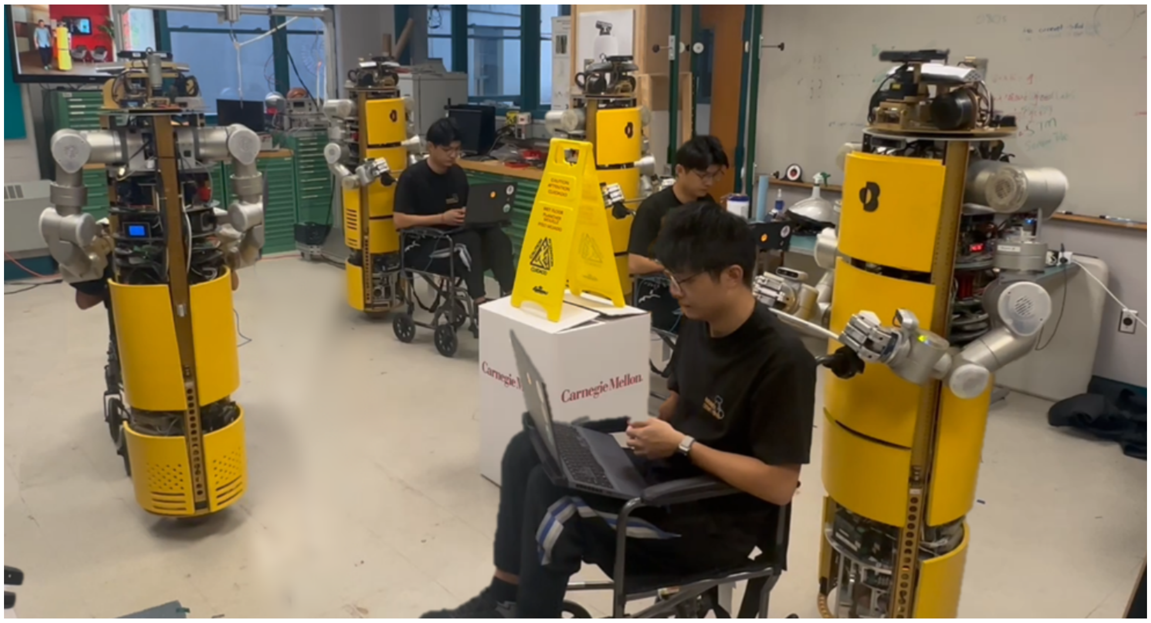


Figure 2.1: Time-lapse picture of the CMU ballbot maneuvering wheelchair around obstacle.

compared to bipedal humanoid robots discussed above. However, a human-height wheeled robot can easily become dynamically unstable if it takes a large impact or accelerates too quickly [16], which led to the adoption of heavy bases with a large footprint in the mentioned studies, hindering mobility in confined areas like hallways such as hallways.

Moreover, previous works lack overall system compliance, which is the compliance in both the arms and the base, either due to hardware limitations or the control methods, key for ensuring safe pHRI in crowded environments.

For wheelchair-pushing tasks, we need to further consider the riding experience of the passenger and the safety of the people around. To address this, our goal is to develop a wheelchair-pushing controller for the CMU ballbot that allows accurate and smooth velocity tracking while maintaining overall system compliance.

The CMU ballbot is a human-sized bimanual robot that operates while balancing on a single spherical wheel, featuring both compliance and omnidirectionality [42]. By leaning its body, the ballbot can change its Center of Mass (CoM) leaning angle to exert large forces such as the required force for human sit-to-stand assistance [38]. The torque-sensing 7-DoF arms provide accurate measurement of the interaction

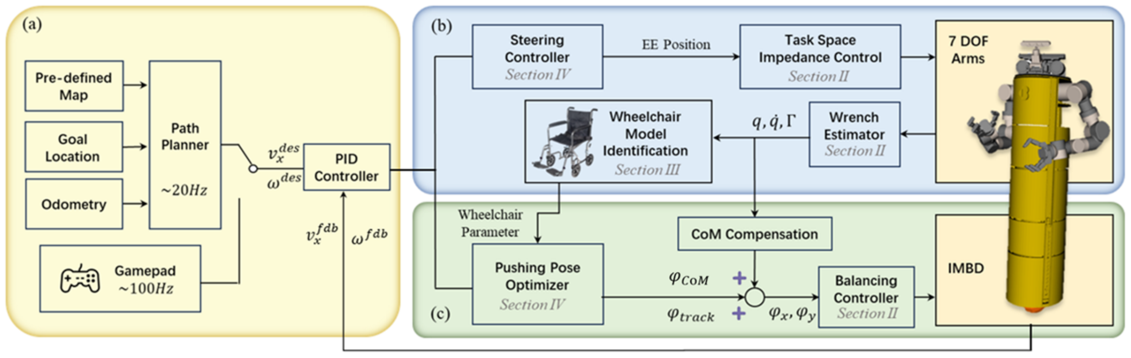


Figure 2.2: Control framework diagram. (a) The reference velocity is first generated by the path planner or sent from a gamepad. The velocity is then tracked by the wheelchair-pushing controller by computing the optimal CoM leaning angle and end-effector position commands. (b) The lower-level task-impedance arm controller then tracks the desired end-effector position commands. A wrench estimator is implemented to provide force estimation that is used for wheelchair model identification. (c) The CoM leaning-angle command is tracked with a balancing controller with CoM compensation.

force [40]. These aspects make the CMU ballot suitable for deployment in caregiving scenarios that require safe pHRI.

In this chapter, we present a control framework for the ballbot to dynamically maneuver a wheelchair as shown in Fig. 2.1, ensuring overall compliance. The controller diagram is shown in Fig. 2.2. The framework's key elements include a pushing pose optimizer and a steering controller, which together plan a whole-body motion, enabling the ballbot to track the desired wheelchair velocity. To identify the parameters in the wheelchair's model, an extended Kalman filter (EKF) is utilized for online system identification (SI). The main contribution of this chapter is a holistic scheme for bimanual dynamic balancing robots to maneuver heavy cart-like systems while ensuring overall compliance.

The remainder of this chapter is constructed as follows: Section 2.3 introduces the modelling of the CMU ballbot. Section 2.4 shows the dynamic model of the wheelchair and online parameter identification. The pushing pose optimizer and steering controller for wheelchair pushing is discussed in Section 2.5. Finally, we show that the proposed method can accurately track velocity commands with various loads by the experiments described in Section 2.6. Finally, conclusions are summarized in

Section 2.7.

2.3 Ballbot Model and Control

This section introduces the ballbot model and implementation of the balancing controller and the arm impedance controller.

2.3.1 Kinematics Model

The frame choices are shown in Fig. 2.3(a), with the origin frame $\{O\}$ centered on the spherical wheel with axes aligned with the inertial frame $\{I\}$, the body frame $\{B\}$ centered at shoulder center with only its z axis aligned with $\{O\}$. The red, green and blue axis in coordinate frames represent x, y and z axis respectively. The arm's task and posture kinematics are defined with respect to body frame $\{B\}$. For both arm, we can write the task relationship between the task coordinates $x \in \mathbb{R}^6$ and the

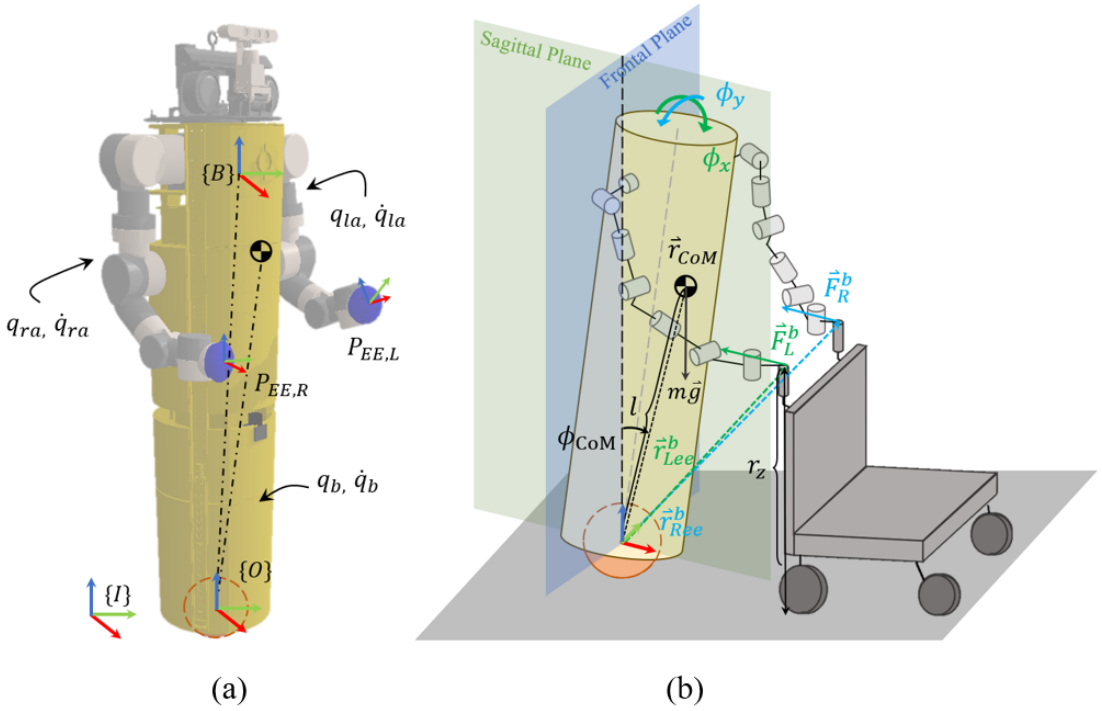


Figure 2.3: (a) The ballbot dynamics model. (b) Quasi-static analysis with forces exerted at the end-effectors.

joint configuration coordinates $q \in \mathbb{R}^{10}$ in the following form:

$$x = \mathbf{FK}(q). \tag{2.1}$$

Here, $q = [q_b, q_a]^T$. $q_b \in \mathbb{R}^3$ is the vector of the body pose defined as $[\phi_x, \phi_y, \phi_z]^T$ w.r.t. $\{O\}$, where ϕ_x, ϕ_y are body leaning angles, and ϕ_z is the body yaw. $q_a \in \mathbb{R}^7$ is the vector of the arm joints. The task coordinate x is defined as $x = [p^o_x, p^o_y, p^o_z, \phi, \theta, \psi]^T$. Where $[p^o_x, p^o_y, p^o_z]^T$ is the position vector w.r.t. $\{O\}$, and $[\phi, \theta, \psi]^T$ are the Z-Y-X Euler angles. The task Jacobian $\boldsymbol{J}(q)$ can then be defined as:

$$\boldsymbol{J}(q) = \frac{\delta \mathbf{FK}(q)}{\delta \boldsymbol{q}} \in \mathbb{R}^{6\times 10}. \tag{2.2}$$

2.3.2 Task-space Impedance Controller

Another aspect of care-giving tasks is to ensure user's safety while ensuring safe physical human-robot interaction with others. Towards this goal, a task-space impedance controller is used for the arm's controller to track the desired end effector motion while providing stable physical interaction. The desired impedance behavior between external force F_{ext} and end-effector position error $e_x = x - x_d$ is that of a mass-spring-damper system of the form:

$$\bar{M}_d \ddot{e}_x + \bar{B}_d \dot{e}_x + \bar{K}_d e_x = F_{ext}. \tag{2.3}$$

The symmetric positive definite matrices $\bar{K}_d, \bar{B}_d$, and $\bar{M}_d$ are the desired stiffness, damping, and inertia matrix, respectively. $\boldsymbol{F}_{\text{ext}}$ is the force exerted at the system. The control law in joint torque space is:

$$\tau^* = \boldsymbol{J}^T \left[\boldsymbol{\Lambda} \ddot{\boldsymbol{x}}_d - \boldsymbol{\Lambda} \overline{\boldsymbol{M}}_d^{-1} \left(\overline{\boldsymbol{B}}_d \dot{\boldsymbol{e}}_x + \overline{\boldsymbol{K}}_d \boldsymbol{e}_x \right) + \boldsymbol{\mu} \right], \tag{2.4}$$

where $\boldsymbol{\Lambda} = \left(JM^{-1}J^T\right)^{-1}$, $\boldsymbol{\mu} = \boldsymbol{\Lambda} \left(JM^{-1} \left(\boldsymbol{h} - \tau_{fric} \right) - \dot{J} \dot{\boldsymbol{q}} \right)$ where $M \in \mathbb{R}^{10\times 10}$ is the mass/inertia matrix, $h \in \mathbb{R}^{10}$ is the vector of Coriolis, centripetal, and gravity forces, and τ_{fric} is the joint friction.

2.3.3 Balancing Controller

The balancing controller is a PID controller cascaded with a PD controller that tracks the desired leaning Center of Mass (CoM) leaning angle ϕ_{CoM} as shown in Fig. 2.3(b). The two arms each have a mass of 12.9 kg, thus posing considerable CoM change to the robot while moving. A CoM compensator is implemented to maintain the CoM to the equilibrium position using body leaning [40].

2.4 Wheelchair Modeling

2.4.1 Wheelchair Dynamics

The wheelchair has two fixed wheels and two caster wheels in the front, since the caster wheels in the front rotate quickly to the direction of motion, we assume that they have a negligible effect on the system's dynamics [2, 6]. The two rear wheels introduce a nonholonomic constraint such that only motion in the wheelchair's current direction is allowed. A way to model this characteristic is by defining the system state as and $q_w = [x, \theta]$ velocity as $\dot{q_w} = [v_x, \omega]$ w.r.t. the wheelchair frame $\{W\}$, which is the center of the two wheel centers. We can then integrate $\dot{q_w}$ and get the

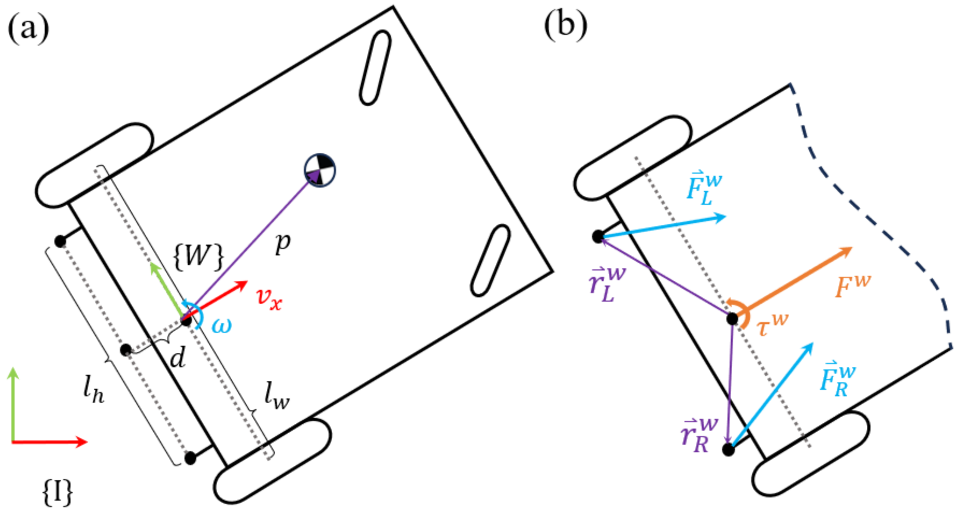


Figure 2.4: Planar wheelchair model. (a) Geometric notations. (b) Input force notations.

wheelchair's position w.r.t. the world frame $\{I\}$. Also due to the same constraint, the input wrench exerted at the handle has only 2 effective DoFs, which are the applied force along the x axis of the cart and torque about the z axis, as any lateral force will be dissipated by the constraints [2]. Finally, we assume that the wheelchair is always on the floor. Thus we can faithfully model the wheelchair's dynamics as a secondary planar system. The kinematic energy of the system w.r.t. frame $\{W\}$ can be calculated by:

$$\begin{aligned} T = \frac{1}{2} m_w v_x^2 - m_w p_x v_x \omega s_\theta - m_w p_y v_x \omega c_\theta \\ + \frac{1}{2} m_w \omega^2 (p_x^2 + p_y^2) + \frac{1}{2} I \omega^2, \end{aligned} \tag{2.5}$$

where s_θ and c_θ are sine and cosine functions of θ. The dynamics equation is given by:

$$\frac{d}{dt}\left(\frac{\partial \mathscr{L}}{\partial \dot{q_i}}\right) - \frac{\partial \mathscr{L}}{\partial q_i} = \Gamma, \tag{2.6}$$

where Γ is the external wrench applied to the system, including the wrench applied by the robot Γ_w and non-conservative force $N_w(\dot{q_w})$ such as the viscous force as the wheels rotate. Then, we can reformulate the dynamics equation as:

$$M_w \ddot{q_w} + C_w(\dot{q_w})\dot{q_w} = \Gamma_w - N_w(\dot{q_w}), \tag{2.7}$$

where $M_w(q_w)$ is the inertia matrix, and $C_w(\dot{q_w})$ is the Coriolis term. In the following, the variables describe the properties of the overall wheelchair system (including additional load and passenger). The formulations can be given as:

$$\begin{aligned} M_w &= \begin{bmatrix} m_w & -m_w\left(p_x s_\theta + p_y c_\theta\right) \\ -m_w\left(p_x s_\theta + p_y c_\theta\right) & I_w + m_w\left(p_x^2 + p_y^2\right) \end{bmatrix}, \\ C_w &= \begin{bmatrix} 0 & -m_w \omega\left(p_y s_\theta - p_x c_\theta\right) \\ 0 & 0 \end{bmatrix}, \quad \Gamma_w = \begin{bmatrix} F^w \\ \tau^w \end{bmatrix}, \\ N_w &= \sigma \begin{bmatrix} v_x \\ \omega \end{bmatrix}, \end{aligned} \tag{2.8}$$

where σ is a positive viscous coefficient that opposes the system's motion, m_w and I_w are the mass and rotational inertia at the wheelchair CoM respectively. The geometric notations are shown in Fig. 2.4 (a), where l_w, l_h is the distance between the two rear wheels and the two handles respectively, d is the x-axis offset between the handles and wheels w.r.t $\{W\}$, and p_x, p_y are the distance of CoM w.r.t. $\{W\}$. The input wrenches are typically exerted at the handles as shown in Fig. 2.4 (b), the mapping between Γ_{input} and Γ_w is :

$$\Gamma_{input} = \begin{bmatrix} \text{Proj}_x \vec{F_L^w} + \text{Proj}_x \vec{F_R^w} \\ \vec{r_L^w} \times \vec{F_L^w} + \vec{r_R^w} \times \vec{F_R^w} \end{bmatrix}. \tag{2.9}$$

In this context, we assume that the friction between the wheel and the floor is always static friction solely arising from the bearing friction in the rear wheels. We can further model the friction coefficient as:

$$\sigma = [\mu N \quad \mu N l_w/2]. \tag{2.10}$$

where μ is the rotational friction coefficient, and N is the force exerted on the wheel. By assuming that the wheelchair's mass m_w is evenly distributed on the four wheels, the force on each wheel is $N = \text{mg}/4$, with $g = 9.8\ \text{m}^2/\text{s}$.

2.4.2 Wheelchair Model Identification

An EKF is used to estimate the unknown parameters, similar to [2]. We need to estimate the mass m_w, CoM position of the system $[p_x\ p_y]^T$ and the friction coefficient μ. We pick a set of parameters that would be linear in the dynamic equations for ease of computation by selecting $\phi = \begin{bmatrix} m_w & m_w p_x & m_w p_y & I + m_w \left(p_x^2 + p_y^2\right) & \sigma \end{bmatrix}^T$, these can then be used to calculate the desired parameters. The EKF state vector then consists of:

$$\hat{\mathbf{x}} = \begin{bmatrix} \hat{q_w}^T & \dot{\hat{q}}_w^T & \hat{\phi}^T \end{bmatrix}^T, \tag{2.11}$$

And the derivative of the state vector is:

$$\dot{\hat{\mathbf{x}}} = \begin{bmatrix} \dot{\hat{q}}_w \\ \ddot{\hat{q}}_w \\ \dot{\hat{\phi}} \end{bmatrix} = \begin{bmatrix} \dot{\hat{q}}_w \\ \hat{M_w}^{-1} \left(\Gamma_w - \hat{C_w}\dot{\hat{q}}_w - \hat{N_w}\right) \\ 0 \end{bmatrix}. \tag{2.12}$$

We used an end-effector wrench estimator developed in previous work [39] to estimate the end-effector wrench Γ based on joint torque sensor readings. Then a simple moving average filter is applied to smooth results. The initial values are set to $\phi_0 = \begin{bmatrix} 60 & 0 & 0 & 30 & 0.001 \end{bmatrix}^T$, and they are updated online at 100 Hz. The result of online estimation during experiment is shown in Section V.

2.5 Wheelchair Pushing Controller

In this part, we propose a pushing pose optimizer based on a quasi-static analysis of the system. The key insight of our controller is to treat the arms as passive spring-like elements and use the body leaning as the dominant factor in velocity control. We also introduce a steering controller that helps find optimal reference positions for end-effectors and minimizes the required leaning angle.

2.5.1 Pushing Pose Optimizer

For pushing control, we are interested in mapping from body leaning angle to the force exerted on the wheelchair. When the ballbot leans, the arms of the robot will be compressed and have force outputs as shown by Eq. (2.3). If the arm controllers share the same stiffness and damping parameters $\bar{K}_d$ and $\bar{B}_d$, they will have comparable compression, thus a similar force output. We validated this effect by measuring the EE displacement during a 2-minute run as shown in Fig. 2.5. The simplification still holds when the length of the arms are not the same. To simplify the control, we assume that the two arms have equal output $\vec{F_L^b} = \vec{F_R^b} = \vec{F^b}$. Since we set the rotational stiffness to be 0, we also assume that there is zero output torque. Then the torque equilibrium at the ball center can be expressed as:

$$-\left(\vec{r_{Lee}^b} + \vec{r_{Ree}^b}\right) \times F^b + \vec{r}_{\text{CoM}} \times m_{robot}\vec{g} - \tau_{robot} = 0. \tag{2.13}$$

Here, $\vec{r_{Lee}^b}$ and $\vec{r_{Ree}^b}$ are the position vector of the end-effectors w.r.t. $\{B\}$ as shown in Fig. 2.3, m_{robot} is the mass of the ballbot, and τ_{robot} is the net torque result from the friction between the ballbot body and its inverse mouseball drive system (IMBD).

The EEs are commanded to maintain constant height at the handle, so we can

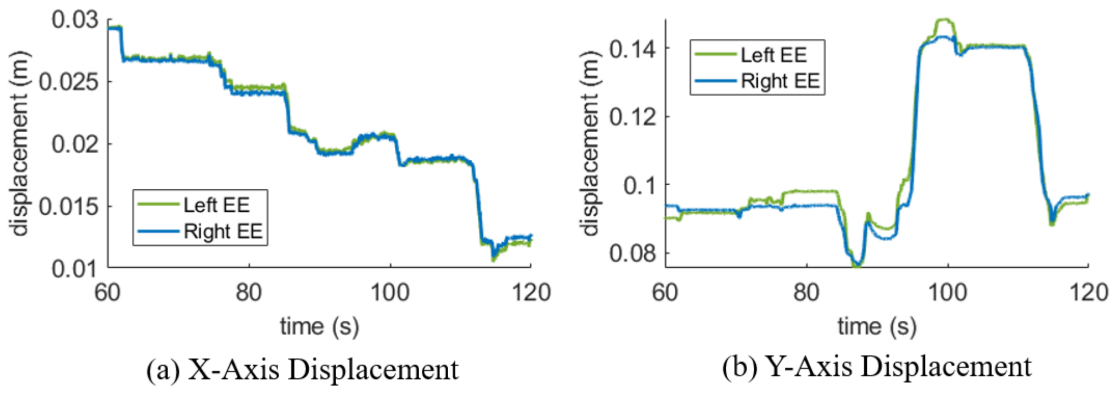


Figure 2.5: End-effector displacement during 2-minute wheelchair maneuvering experiment on (a) x axis and (b) y axis w.r.t. $\{B\}$. The maximum displacement mismatch between the two EEs are 0.011 m on the x axis and 0.052 m on the y axis.

assume that no z-axis force will be exerted. Then, for the x and y axes we have:

$$2F_x^b r_z = m_{robot} gl \sin \phi_x, \quad 2F_y^b r_z = m_{robot} gl \sin \phi_y. \tag{2.14}$$

Here, r_z is the height of the wheelchair handle, and F_x^b and F_y^b are the projection of F^b on the sagittal and frontal plane. Linearizing the equations at $\phi_x, \phi_y = 0$, and we can have a mapping between the output force and body leaning angle:

$$F_x^b = \frac{m_{robot} gl \phi_x}{2r_z}, \qquad F_y^b = \frac{m_{robot} gl \phi_y}{2r_z}. \tag{2.15}$$

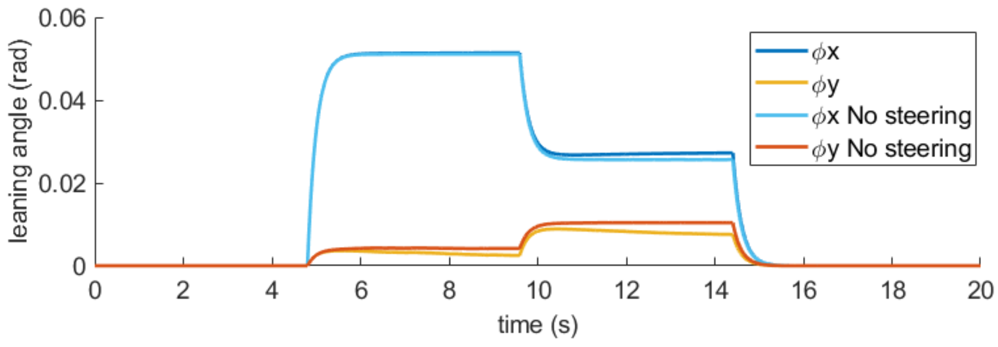


Figure 2.6: Comparison of ballbot CoM leaning angle ϕ_x, ϕ_y calculated by the pushing pose optimizer with and without the steering controller. With the steering controller activated, the required CoM leaning angle becomes smaller, indicating less aggressive body movement.

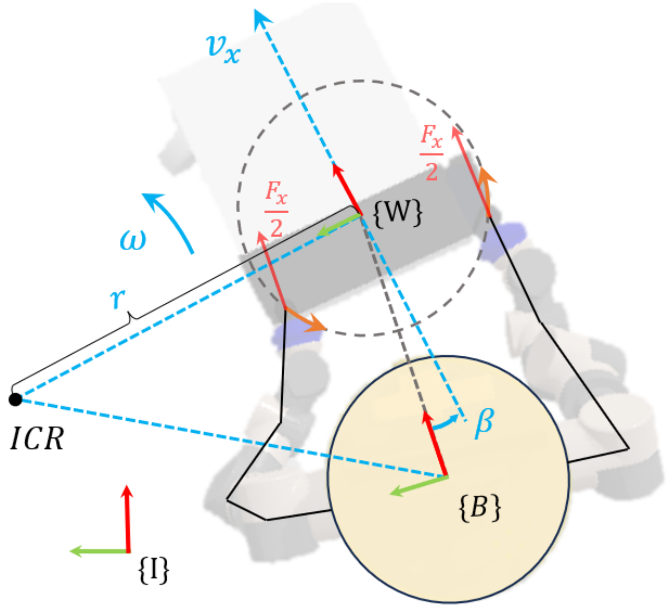


Figure 2.7: Wheelchair steering controller schematic. The end-effectors steer the wheelchair around the rotational center of wheelchair, i.e., the midpoint of the two rear fixed wheels, based on the steering angle β.

Now we move on to the wheelchair system. From Eq. (2.7), we know that when the wheelchair moves at constant velocity $\dot{q_w}$, we will have:

$$C_w(\dot{q_w})\dot{q_w} + N_w(\dot{q_w}) = \Gamma_{input}. \tag{2.16}$$

Then, expressing the input matrix in the ballbot frame we have:

$$\Gamma_{input} = \begin{bmatrix} 2\,\mathrm{Proj}_x\, R_b^w \vec{F^b} \\ (\vec{r_L^w} + \vec{r_R^w}) \times R_b^w \vec{F^b} \end{bmatrix}. \tag{2.17}$$

Here, matrix R_b^w is the rotation matrix that transforms a vector in the ballbot frame to the wheelchair frame $\{W\}$.

Based on Eq. (2.17) , we assume that the output force of the end-effectors is influenced by the leaning angle of the body, ϕ_x and ϕ_y. The system is expected to reach the target velocity $\dot{q_w}$ and satisfy the equilibrium equations Eq. (2.16). Put

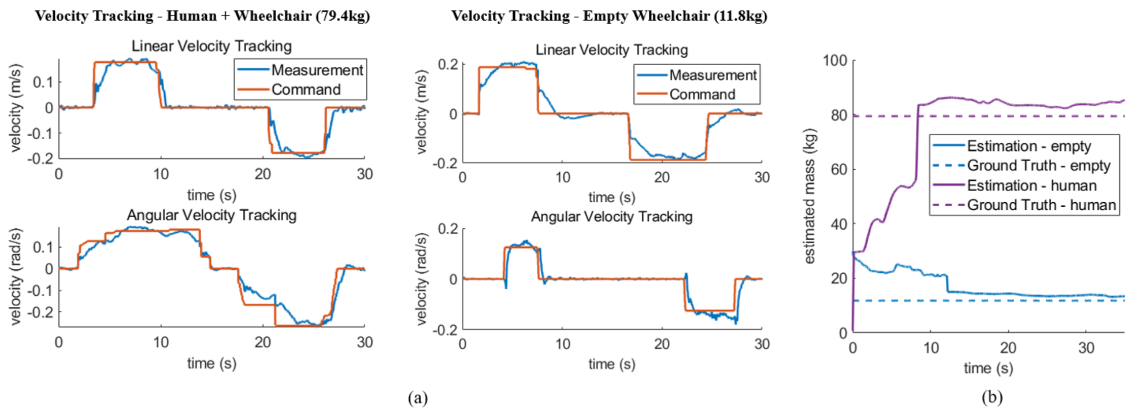


Figure 2.8: (a) Velocity tracking performance. The left figure shows results when a human sits in the wheelchair and the right figure shows results with a empty wheelchair. (b) Online mass estimation with human (79.4 kg in total) and empty cart (11.8 kg).

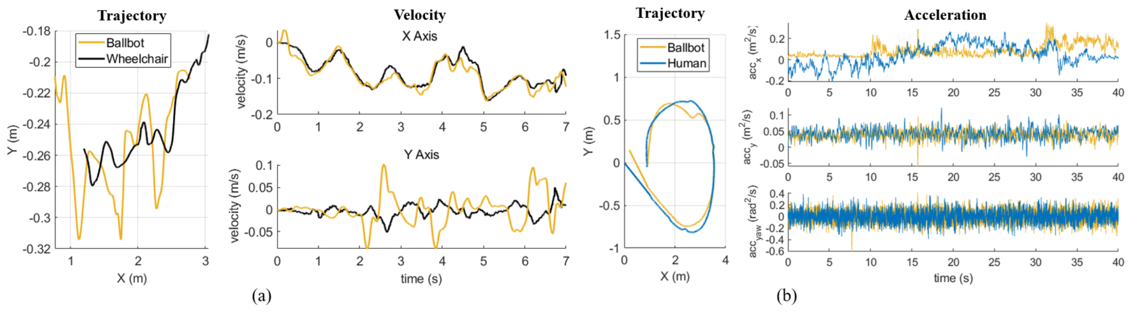


Figure 2.9: (a) The trajectory (left figure) and velocity (right figure) of both the ballbot and the wheelchair when disturbed while pushing. (b) The trajectory (left figure) and acceleration (right figure) of the wheelchair when pushed with the ballbot and human.

these equations in matrix form and we have:

$$\underbrace{C_w(\dot{q_w})\dot{q_w} + N_w(\dot{q_w})}_{\boldsymbol{f}} = \underbrace{\begin{bmatrix} 2m_{robot}gl\,\mathrm{Proj}_x\,R_b^w \\ m_{robot}gl[\vec{r_L^w} + \vec{r_R^w}]_\times R_b^w \end{bmatrix}}_{\mathbf{A}} \boldsymbol{\Phi}, \tag{2.18}$$

where $\Phi = [\phi_x, \phi_y, 0]^T$. $[\vec{a}]_\times$ is the cross-product matrix of $\vec{a}$. For any $\vec{a}, \vec{b} \in \mathbb{R}^3$, we have $[\vec{a}]_\times \vec{b} = \vec{a} \times \vec{b}$.

To calculate the optimal body leaning angle, this problem is formulated as an unconstrained quadratic program problem:

$$\min_{\boldsymbol{f}} (\mathbf{A\Phi} - f)^T Q (\mathbf{A\Phi} - f) + \mathbf{\Phi}^T \mathrm{R} \mathbf{\Phi}, \tag{2.19}$$

where Q, R are positive definite weight matrices. The closed-form solution is:

$$\widehat{\mathbf{\Phi}} = \left(\mathbf{A}^\top \mathbf{Q} \mathbf{A} + \mathbf{R}\right)^{-1} \mathbf{A}^\top \mathbf{Q} \boldsymbol{f}. \tag{2.20}$$

In practice, this is solved online at 100 Hz. At each time step, the controller will update the optimal lean angles. The lean angles are then tracked by the low-level balancing controller.

2.5.2 Steering Controller

We want to minimize the ballbot lateral leaning angle ϕ_y to avoid collision between the arms and the body, while keeping the ability to exert torque on the wheelchair for turning. To address this, we propose a wheelchair steering heuristic that plans desirable EE positions for the pushing pose optimizer such that ϕ_y is minimized.

By positioning the ballbot's body such that the center of $\{W\}$ is on the x axis of $\{B\}$, the z-axis torque exerted on the wheelchair can be controlled with ϕ_x and steering angle β as shown in Fig. 2.7. By steering the wheelchair's direction, we effectively change the z-axis torque exerted on the wheelchair such that the system tracks the desired velocity $\left[v_x^{des}, \omega^{des}\right]$ around the Instantaneous Center of Rotation (ICR). The turning radius can be calculated as $r_{ICP} = v_x^{des}/\omega^{des}$, and steering angle β can be calculated as:

$$\beta = \sin^{-1}\left(\frac{\omega l_w}{4 v_x^{des} d}\right). \tag{2.21}$$

The desired end-effector $\widehat{\vec{r^b_{Lee}}}$ and $\widehat{\vec{r^b_{Ree}}}$ positions can be calculated as:

$$\begin{aligned} \widehat{\vec{r^b_{Lee}}} &= [d, 0]^T - R_z(\widehat{\beta})^T l_w/2, \\ \widehat{\vec{r^b_{Ree}}} &= [d, 0]^T + R_z(\widehat{\beta})^T l_w/2, \end{aligned} \tag{2.22}$$

where R_z is the rotation around the z axis. From this, the angular velocity can be controlled with steering angle and linear velocity. Due to arm workspace constraint, the steering angle is limited as $\beta \in [-35°, 35°]$.

2.6 Experiments

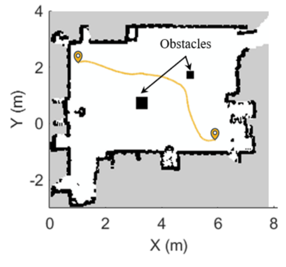
2.6.1 Velocity Tracking with Different Loads

To demonstrate the velocity tracking performance of the proposed planner, we tested with an empty wheelchair (11.8 kg) and a human subject weighing 67.6 kg (79.4 kg in total). The parameters mentioned in Section III are estimated online and updated to the wheelchair model, and the velocity command is sent from a gamepad. The results are shown in Fig. 2.8, where (a) shows linear and angular velocity tracking with different loads, and (b) shows the online estimated mass. When maneuvering an empty cart, the average system response time is 1.6 s for a 0.2 m/s step in linear velocity, and 1.7 s for a 0.15 rad/s step in angular velocity. When maneuvering a wheelchair with a person, the average system response time is 1.1 s for a 0.2 m/s step in linear velocity, and 1.8 s for a 0.18 rad/s step in angular velocity. The online estimation converges in 10.2 s with a 3.2 kg (4%) steady-state error when pushing the wheelchair with a human, and converges in 13.4 s with a 1.1 kg (9.3%) when pushing an empty wheelchair. The maximum velocity achieved in our experiments with a load on a wheelchair (34.6 kg in total) is 0.45 m/s of linear velocity and 0.3 rad/s of angular velocity. Due to the limited space in the room, we did not conduct further tests on the system's maximum speed. Furthermore, the system demonstrated the ability to effectively turn the wheelchair in place, which is important to navigating in narrow spaces such as hallways and elevators.

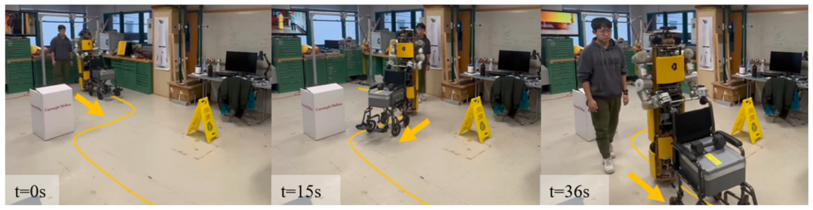
2.6.2 Motion Smoothness

In this experiment, we compared the wheelchair's acceleration when maneuvered by the ballbot against when manuvered by a human as a metric to evaluate motion smoothness [36]. Initially, we remotely operated the ballbot to navigate through the lab around an obstacle as shown in Fig. 2.1, followed by a human attempting to replicate the same path. An Intel Realsense T265 sensor is mounted on the wheelchair to record the trajectory and acceleration data as shown in Fig. 2.9 (b). We assessed the wheelchair's motion smoothness using the acceleration norm as a metric, with a lower acceleration norm suggesting gentler accelerations or decelerations, thereby indicating smoother movements.

The data showed that the ballbot's maneuvering resulted in 25.3% higher acceleration norm on the x axis, 21.7% on the y axis, and 19% on the z axis w.r.t. $\{B\}$ compared to human maneuvering. This result showed that our approach can generate human-like motion smoothness of the wheelchair, which is essential for ensuring passenger's comfort.



(a)



(b)

Figure 2.10: Navigation across lab with predefined obstacles. (a) Map with predefined obstacles. The markers indicate the start position (left marker) and goal position (right marker), and the yellow line shows the actual trajectory of the ballbot. (b) Time-lapse picture of the experiment. Yellow arrow indicates ballbot's velocity direction.

2.6.3 Physical Compliance

Another important goal of the proposed controller is the ability to ensure safe pHRI with overall compliance. To validate this, we manually disturbed the ballbot's body while it is tracking a linear velocity command of 0.1 m/s. The velocity and position of both the ballbot and wheelchair are measured to analyze how the impact is transmitted to the wheelchair. As suggested in Fig. 2.9 (a), the x-axis velocity of the wheelchair closely mirrors that of the ballbot. On the y axis, the disturbance is more evidently weakened due to overall compliance, resulting in a less aggressive velocity change of the wheelchair compared to the ballbot. Furthermore, we tested the overall compliance of the system by measuring the minimum force required to move the wheelchair by pushing on ballbot. The measured required force is 22.7 N on the x axis and 12.3 N on the y axis w.r.t. $\{B\}$ for the system to move, showing that the overall compliance of the system can ensure safety for people that might collide with the system, and can be manually stopped with small amount of force.

2.6.4 Navigating with the Wheelchair

A vital task for wheelchair maneuvering is to navigate it to the desired position while avoiding obstacles. In this experiment, we integrated the proposed controller with the ROS navigation framework [19]. The task is to navigate the wheelchair with a 34.6 kg load across the lab in a known map and avoid predefined obstacles as shown in Fig. 2.10 (a). The planner replans at 20 Hz frequency online and sends real-time velocity commands to the controller. Acceleration constraints were added to the planner to generate feasible commands for the wheelchair-pushing system, avoiding drastic movements. Fig. 2.10 (c) shows the time-lapse picture of this experiment. The system successfully reached the given goal location without collision. This experiment shows the controller's ability to effectively track real-time velocity commands and compatibility with the existing navigation pipeline.

2.7 Conclusion

We present a control framework that enables the CMU ballbot to maneuver a wheelchair while maintaining balance. The core idea is to utilize whole-body motion to maneuver while maintaining overall compliance. The proposed method is evaluated in real hardware experiments, showing that the approach can perform desirable maneuvers while ensuring smooth wheelchair motion and safe pHRI. In future work, autonomous affordance detection will allow grasping point identification. It will also be necessary to evaluate the ballbot pushing a wheelchair up and down ADA-compliant ramps [46]. Furthermore, it is likely that the proposed framework can be applied to similar non-holonomic cart systems such as wheelbarrows and shopping carts.

Chapter 3

Self Pushing: Push wall to navigate with Contact-Implicit MPC

3.1 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 [11][7] and manipulation [20] 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 [41], [9],[14]. 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

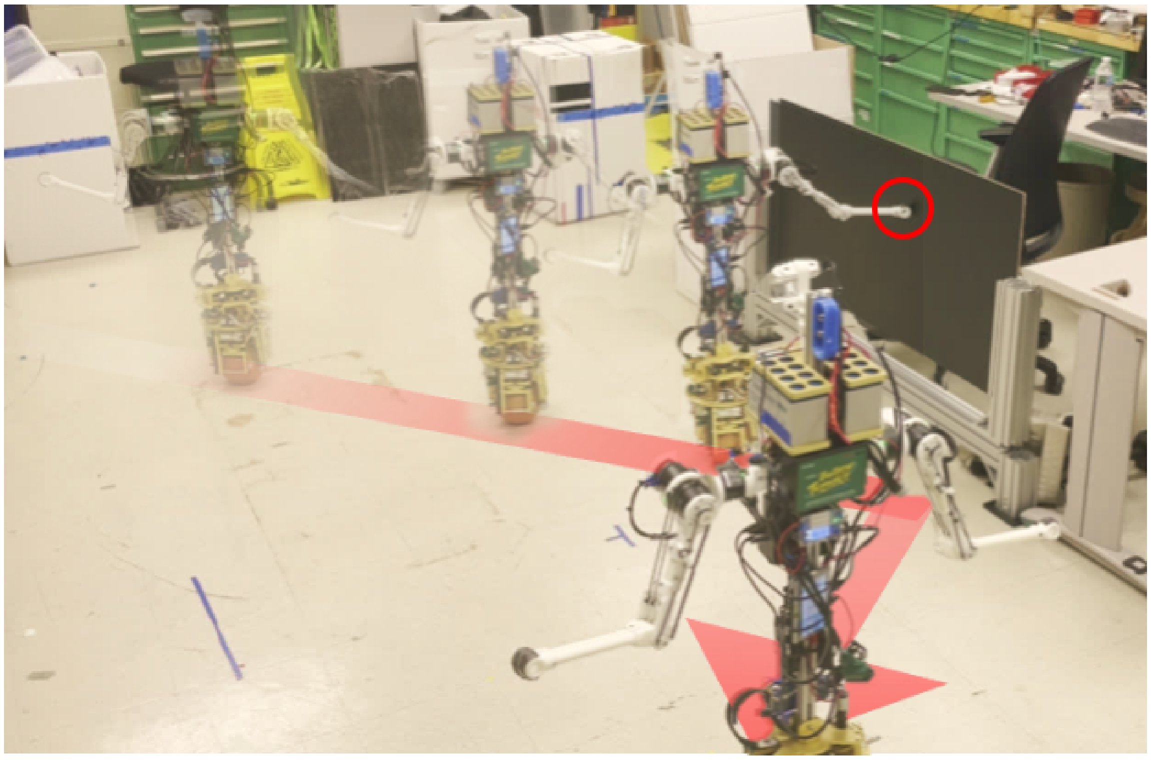


Figure 3.1: Time-lapse picture of CMU shmoobot making a sharp turn by pushing wall.

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 [7, 14]. 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 [26, 43]. Other works use search-based methods to find possible contact strategies [15, 44]. However, these methods are based on kinematics and quasi-static analysis and cannot capture the dynamic effect of the robot.

In this chapter, we investigate using end effector contact on the CMU shmoobot (a smaller CMU ballbot [27]) 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 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.

3.2 Related Works

3.2.1 Locomotion With Upper Limb Contacts

Like humans, human-like robots can use their upper limbs to assist with locomotion. Research [48] explored how a humanoid robot can make contact with a wall to prevent falling, while [26, 44] 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 [4] focuses on transferring human wall-pushing skills directly to robots. In [5], 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.

3.2.2 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)[18, 34] 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[3]. Research [21] treats the contact dynamics as constraints and solves them with the direct collocation method. A higher-order collocation method [18, 45], 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 [50], making them unpractical during deployment in online MPC settings without specialized solvers [17]. Another approach of solving CIO is by using compliant contact models. Works [24, 25], use a continuous contact flag to control the allowed contact force. Other works [28, 29], 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.

3.3 Background

3.3.1 Platform Description

The CMU shmoobot (Fig. 3.2(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)[27]. 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.

3.3.2 Symbol and Notations

The body and world coordinate systems are defined in Fig. 3.2. Quantities in the body frame have a left subscript B. Other quantities are in the world coordinate system. Vectors are bold and lowercase ($\mathbf{a}$, ω), 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':

$$[\mathbf{v}]_\times = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}, \tag{3.1}$$

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

3.3.3 MPC overview

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

$$\begin{cases} \min\limits_{\mathbf{u}(.)\ x(.)} \quad \sum_i \Phi_i(\mathbf{x}(t_{i+1})) + \displaystyle\int_{t_i}^{t_{i+1}} l_i(\mathbf{x}(t), \mathbf{u}(t), t)\, dt \\ \text{s.t. } \mathbf{x}(t_0) = \mathbf{x}_0 & (3.2 \text{ - } 1) \\ \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) & (3.2 \text{ - } 2) \\ \mathbf{g}_i(\mathbf{x}(t), \mathbf{u}(t), t) = \mathbf{0} & (3.2 \text{ - } 3) \\ \mathbf{h}_i(\mathbf{x}(t), \mathbf{u}(t), t) \geq \mathbf{0} & (3.2 \text{ - } 4) \\ \text{for } t_i < t < t_{i+1} \text{ and } i \in \{0, 1, \cdots, I-1\} \end{cases}. \tag{3.2}$$

Here, (3.2-1) is the initial state constraint; (3.2-2) is the system dynamics constraint; (3.2-3) and (3.2-4) are equality and inequality constraints respectively. In our formulation, the active constraints vary as the contact 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.

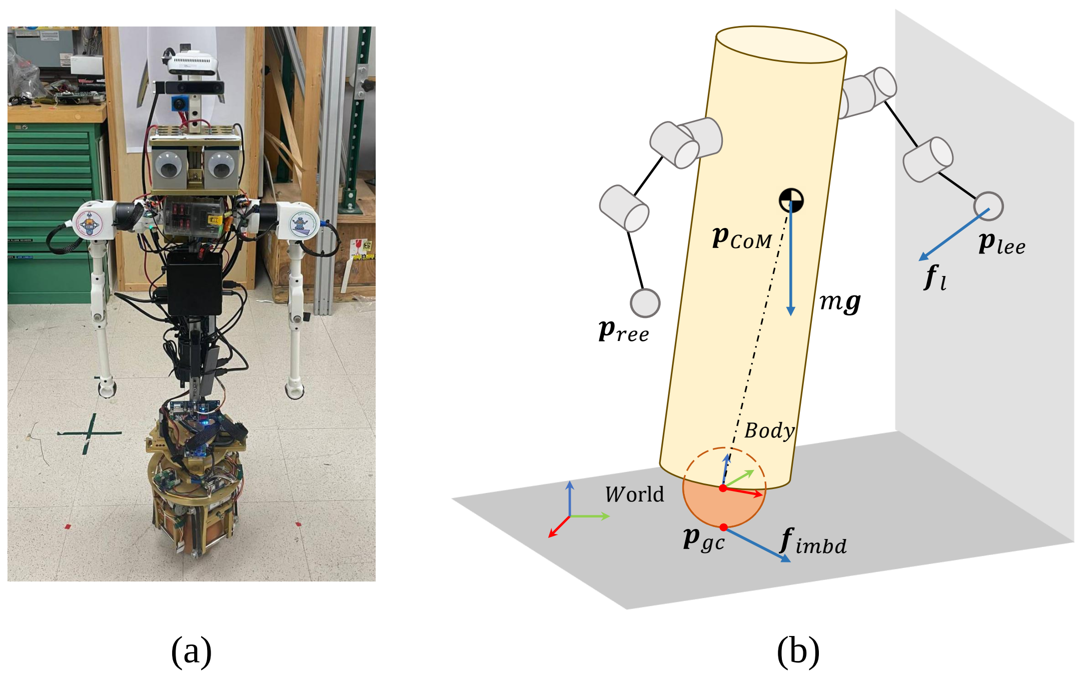


Figure 3.2: (a) CMU shmoobot. (b) Coordinate systems of Shmoobot.

3.4 System Modeling

3.4.1 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. 3.2(b), the origin of the robot base frame is defined at the center of the ball wheel. The general coordinate 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.

3.4.2 Simplified Robot Dynamics

We use single rigid body dynamics to model the shmoobot system. Most of the arm's mass is concentrated in its shoulder, while the linkages are comparatively lighter. We neglect the mass of the arm linkages 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:

$$\begin{aligned} \sum \mathbf{f}^i &= m\dot{\mathbf{v}} + m[\mathbf{r}_{com}]_\times \dot{\omega} - m[\boldsymbol{\omega}]_\times [\boldsymbol{\omega}]_\times \mathbf{r}_{CoM}, \\ \sum \boldsymbol{\tau}^i + \sum [\mathbf{r}_f^i]_\times \mathbf{f}^i &= \dot{\mathbf{l}}_r - [\mathbf{r}_{com}]_\times m\mathbf{a} + [\omega]_\times I_{inertia}\omega, \end{aligned} \tag{3.3}$$

where m is the body mass, $I_{inertia}$ is the inertia matrix, $\mathbf{v}$ is the linear velocity, ω is the angular 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:

$$m\dot{\mathbf{v}} = \sum [\mathbf{f}_x^i, \mathbf{f}_y^i, 0]^T, \tag{3.4}$$

$$\dot{\mathbf{l}}_{\mathbf{r}} = \sum \boldsymbol{\tau}^i + \sum [\mathbf{r}_f^i]_\times \mathbf{f}^i + [\mathbf{r}]_\times [\sum \mathbf{f_x^i}, \sum \mathbf{f_y^i}, 0]^T. \tag{3.5}$$

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:

$$m\dot{\mathbf{v}} = [\mathbf{f}_{IMBD,x}, \mathbf{f}_{IMBD,y}, 0]^T + \sum_{i=1}^{2} \left[\mathbf{f}_{c,x}^i, \mathbf{f}_{c,y}^i, 0\right]^T, \tag{3.6}$$

$$\dot{\mathbf{l}}_{\mathbf{r}} = \sum_{i=1}^{2} \mathbf{r}_{b,c}^i \times \mathbf{f}_c^i + \mathbf{r}_{b,CoM} \times m\mathbf{g} + \tau_{yaw} + \mathbf{r}_{b,CoM} \times m\dot{\mathbf{v}}. \tag{3.7}$$

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. τ_{yaw} is the input torque along the yaw axis. $\mathbf{r}_{b,c_i}$ 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.

3.5 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.

3.5.1 Contact Frame Definition

As shown in Fig. 3.3(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:

$$\mathbf{e}_{\text{normal}} = \nabla D(\mathbf{p}). \tag{3.8}$$

In this work, 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:

$$\mathbf{e}_{tangent} = \mathbf{e}_{normal} \times \mathbf{e}_z. \tag{3.9}$$

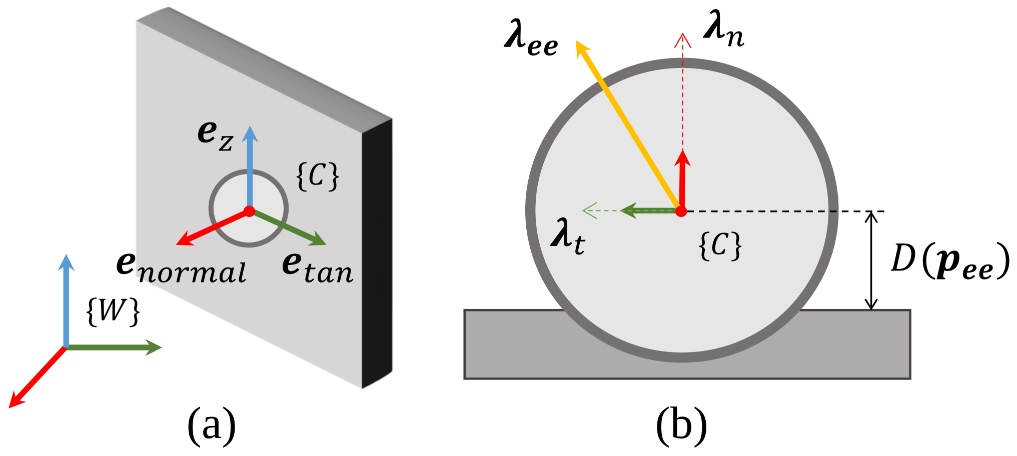


Figure 3.3: (a) Definition of contact frame. (b) Schematic of contact.

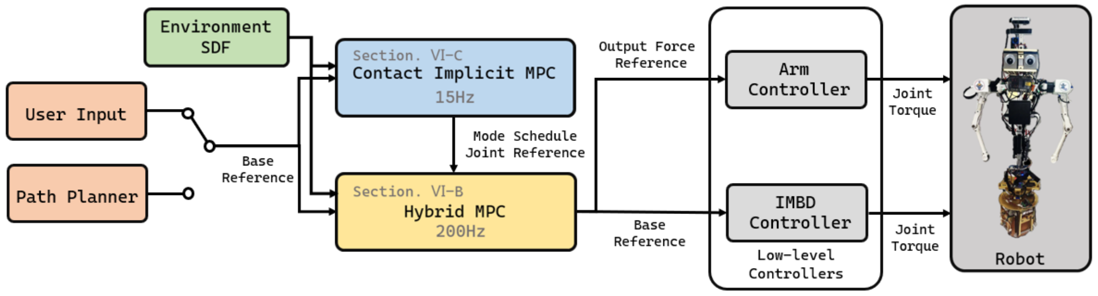


Figure 3.4: Control framework diagram. A reference trajectory is first generated by a path planner or sent by the user. A bi-level MPC will calculate an optimal trajectory that tracks the reference. The environment SDF used by the controller needs to be recomputed. The upper level of the controller (the 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.

3.5.2 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:

$$\lambda_{normal} = f(D(\mathbf{p}_{ee})). \tag{3.10}$$

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

$$f(d) = 0.5f_{max} \cdot tanh(-\alpha \cdot (d + \beta)) + 0.5f_{max}, \tag{3.11}$$

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:

$$D(\mathbf{p}_{ee}) \geq 0. \tag{3.12}$$

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

$$\lambda_{\text{ee}}\dot{\mathbf{p}}_{\text{ee}} = 0. \tag{3.13}$$

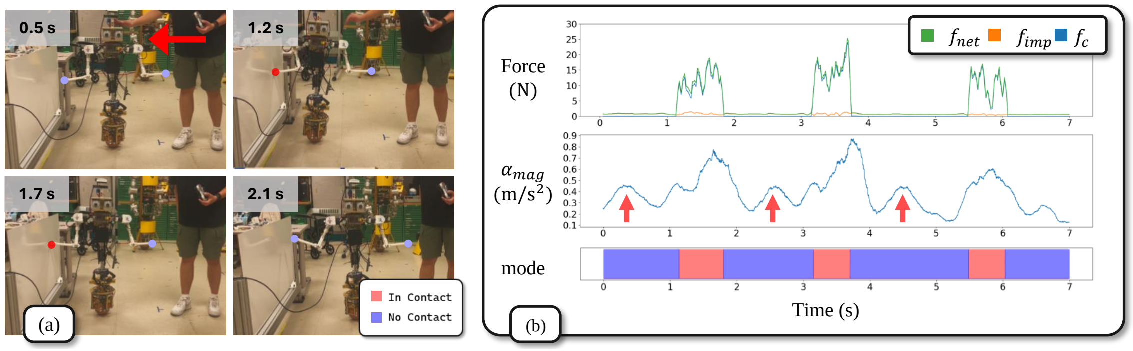


Figure 3.5: (a) Key frames for the disturbance rejection experiment. The end effectors are highlighted with blue and red circles. A red circle indicates that the end effector is in contact. At T = 0.5 s, a human operator pushed the robot to the wall. The robot actively used its arm to recover from the disturbance. (b) Force output trajectory, acceleration trajectory and mode sequence of the robot.

3.5.3 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:

$$\begin{cases} \dot{\mathbf{p}}^{\text{i}}_{ee} = \mathbf{0} \\ D(\mathbf{p}^{\text{i}}_{ee}) = \mathbf{0} \\ \lambda^{\text{i}}_{normal} \cdot \mathbf{e}^{i}_{normal} > \mathbf{0} \\ -\mu\lambda^{i}_{normal} \leq \lambda^{i}_{tan} \leq \mu\lambda^{i}_{normal} \\ -\mu\lambda^{i}_{normal} \leq \lambda^{i}_{z} \leq \mu\lambda^{i}_{normal} \end{cases} \quad \text{if } c_i \in \text{C}. \tag{3.14}$$

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

$$\begin{cases} \lambda_{ee}^{\mathrm{i}} = \mathbf{0} \\ D(\mathbf{p}_{ee}^{\mathrm{i}}) \geq \mathbf{0} \end{cases} \quad \text{if } c_i \in \overline{\mathrm{C}}. \tag{3.15}$$

The set of contacts, C, is be obtained from the optimized trajectory of the contact-implicit controller. This will be further discussed in Section. 3.6.

3.6 System Design

In this section, we present our locomotion controller framework. The overall system structure is presented in Fig. 3.4.

3.6.1 Hybrid MPC

System Dynamics

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

$$\mathbf{x} = \left[\mathbf{h}_b^T, \mathbf{q}_b^T, \mathbf{q}_j^T\right]^T, \mathbf{u} = \left[\mathbf{f}_{IMBD}^T, \mathbf{f}_c^T, \mathbf{v}_j^T\right]^T. \tag{3.16}$$

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 (3.7) we have:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} m\mathbf{v} \\ \mathbf{l_r} \\ \mathbf{q}_b \\ \mathbf{q}_j \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{2}\left[\mathbf{f}_{c,x}^i, \mathbf{f}_{c,y}^i\right]^T + [\mathbf{f}_{IMBD,x}, \mathbf{f}_{IMBD,y}]^T \\ \sum_{i=1}^{2}[\mathbf{r}_{b,c}^i]_\times \mathbf{f}_c^i + [\mathbf{r}_{b,CoM}]_\times m(\mathbf{g} + \dot{\mathbf{v}}) + \tau_{yaw} \\ A_b^{-1}(\mathbf{q}_b)\mathbf{h}_b \\ \mathbf{v}_j \end{bmatrix}. \tag{3.17}$$

Here, A_b is the centroidal momentum matrix which maps generalized velocities to centroidal momentum. Readers can refer to [32] for more details.

Constraints

The contact constraints (3.14), (3.15) are described in section 3.5.3. An input limit constraint is also added.

Cost

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

3.6.2 Contact-Implicit MPC

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, $\alpha = \left[\alpha^1_{\text{tangent}}, \alpha^1_{\text{z}}, \alpha^2_{\text{tangent}}, \alpha^2_{\text{z}}\right] \in \mathbb{R}^4$ 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 (3.10) we have:

$$\lambda^i_{normal} = f(D(FK_i(\mathbf{q}_b, \mathbf{q}_j))). \tag{3.18}$$

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. 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:

$$\lambda^i_{tan} = \alpha^i_{tan} \cdot \lambda^i_{normal}, \tag{3.19}$$

$$\lambda^i_{z} = \alpha^i_{z} \cdot \lambda^i_{normal}. \tag{3.20}$$

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

$$\mathbf{f}_c^i = \lambda^i_{\text{normal}}\mathbf{e}^i_{\text{normal}} + \lambda^i_{\text{tangent}}\mathbf{e}^i_{\text{tangent}} + \lambda^i_z\mathbf{e}^i_{\text{z}}. \tag{3.21}$$

Substitute (3.21) into (3.17), and we can have the system dynamics.

Constraints

The end effector constraints (3.12), (3.13) are described in section 3.5.2. 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.

Cost

Like 3.6.1, the system cost is a quadratic tracking cost to follow a full state trajectory.

3.6.3 Contact Schedule Generation

The contact schedule used by the Hybrid MPC is generated by thresholding the outputs from contact-implicit MPC. With Eqn. 3.10, we can obtain the normal force trajectories $\lambda^i_{normal}(t)$. The algorithm goes through the force trajectories and adds 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.

3.6.4 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.

3.6.5 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:

$$_B\tilde{\mathbf{p}}^{\mathbf{i}}_{\mathbf{ee}} = {}_BFK_i(\tilde{\mathbf{q}}_j). \tag{3.22}$$

$_\mathbf{B}\tilde{\mathbf{p}}^{\mathbf{i}}_{\mathbf{ee}}$ 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:

$$_B\mathbf{f}_{imp} = \mathbf{K}_p({}_B\tilde{\mathbf{p}}^{\mathbf{i}}_{\mathbf{ee}} - {}_B\mathbf{p}^i_{ee}) + \mathbf{K}_d(0 - {}_B\dot{\mathbf{p}}^i_{ee}), \tag{3.23}$$

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:

$$\tau^{\mathbf{i}} = \mathbf{J}^{iT}\left[{}_B\mathbf{f}^i_{imp} + {}_B\mathbf{f}^i_c\right], \tag{3.24}$$

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.

3.7 Experiments

3.7.1 Controller Implementation

The nonlinear optimal control problem is implemented in C++ and solved by the SLQ solver provided by the ETH OCS2 [8] toolbox. The software uses the Eigen3 linear algebra library [12]. 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.

3.7.2 Contact Model

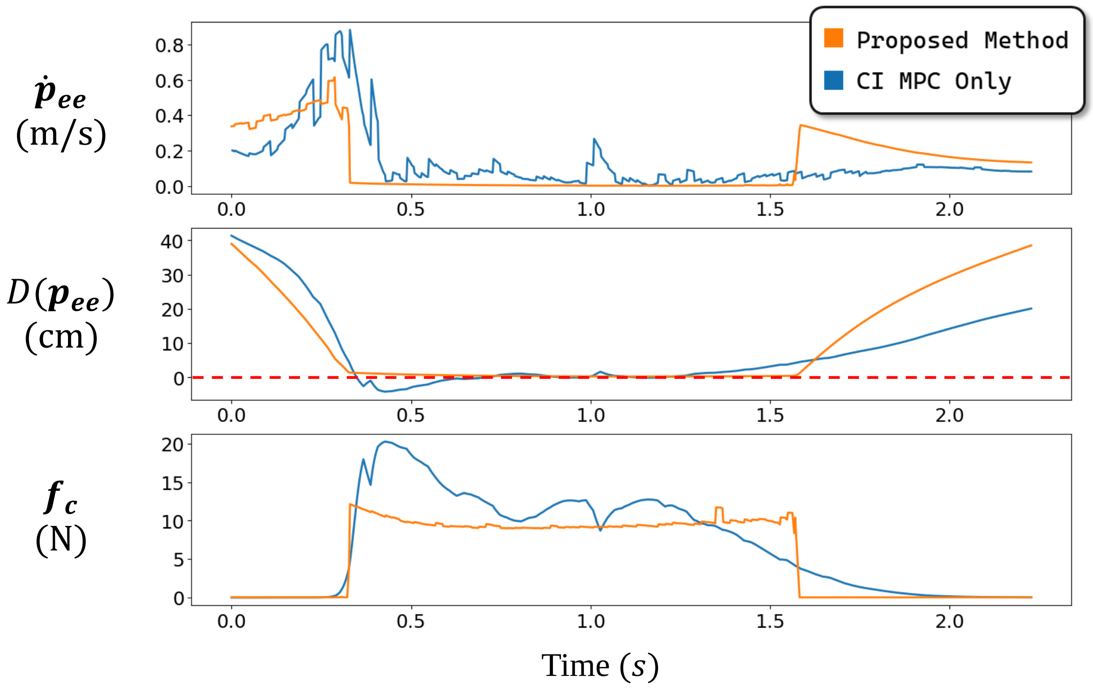


Figure 3.6: The planned end effector velocity (top), position (middle), and force (bottom) trajectories from the proposed bi-level MPC (orange) and CI MPC (blue).

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. 3.6, 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 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.

3.7.3 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. 3.5(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. 3.5(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.

3.7.4 Obstacle Avoidance

In this experiment, we demonstrated the robot's capability to quickly avoid obstacles by pushing against a wall. We use 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. 3.7, 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

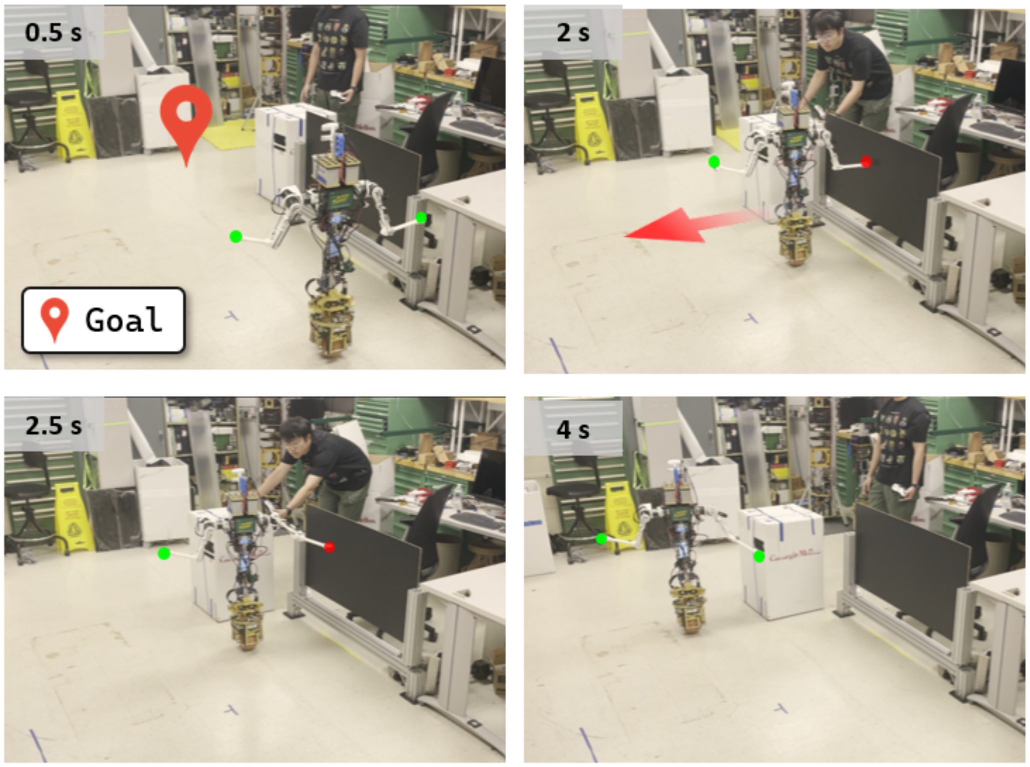


Figure 3.7: Key frames for the obstacle avoidance experiment. The red highlighted point is the goal location. At T = 2 s, an obstacle occurred right in front of the robot. The robot pushed the wall to avoid it.

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.

3.8 Conclusion

In this chapter, 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. In this chapter, we focused solely on contact between two end effectors and vertical walls. Future work could integrate advanced collision detection libraries to account for whole-body contact. Additionally, expanding the framework to include surfaces with varying orientations and incorporating legged locomotion could further enhance the approach. Moreover, the algorithm's performance on long-horizon tasks needs to be investigated.

Chapter 4

Conclusions

This thesis investigated whole-body control strategies for dynamically stable mobile robots. We focus on the tight coupling between locomotion and manipulation. Unlike statically stable or fixed-base platforms, dynamically balancing robots, such as ballbot, cannot treat their manipulators and bases as separate subsystems. Even modest manipulator forces can induce large reaction torques at the base. If not compensated properly, it can even quickly tip the robot over. In manipulation tasks, end-effector forces can destabilize the robot and must be carefully coordinated. In contrast, during locomotion, those end-effector forces can be intentionally applied to interact with the environment and enhance mobility.

The thesis presents two novel whole-body control frameworks. The first is a whole-body controller for wheelchair maneuvering, which enables a ballbot to maintain balance while pushing a human-occupied wheelchair. The second is a bi-level contact implicit MPC controller that can allow shmoobot to utilize manipulator contacts in locomotion tasks.

Future work can explore more deeply integrated controllers where manipulation and locomotion are not just coordinated but mutually reinforced. For example, in microgravity, an astronaut may propel themself by throwing a heavy tool. In this case, the force used to throw the object also pushes his body in the opposite direction. The momentum generated by the manipulation can be reused to help locomotion. Similarly, robots could learn to plan actions that simultaneously achieve both manipulation and locomotion goals.

Realizing such behaviors will require predictive dynamics models that capture the fully coupled robot–object dynamics, like inertia transfer, complex contacts, and different surface features. This also requires optimizing over longer horizon than today's MPC can tractably handle.

Bibliography

[1] Sergio Aguilera and Seth Hutchinson. Control of cart-like nonholonomic systems using a mobile manipulator. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6801–6808, Detroit, MI, USA, October 2023. IEEE. ISBN 978-1-66549-190-7. doi: 10.1109/IROS55552.2023.10342088. 1, 2.2

[2] Sergio Aguilera, Muhammad Ali Murtaza, Jonathan Rogers, and Seth Hutchinson. Modeling and inertial parameter estimation of cart-like nonholonomic systems using a mobile manipulator. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3073–3079, London, United Kingdom, May 2023. IEEE. ISBN 9798350323658. doi: 10.1109/ICRA48891.2023.10161076. 2.4.1, 2.4.2

[3] Mihai Anitescu and Florian A. Potra. Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems. *Nonlinear Dynamics*, 14:231–247, 1997. URL `https://api.semanticscholar.org/CorpusID:7393680`. 3.2.2

[4] Cornelia Bauer and Nancy 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)*, pages 1–8, 2023. doi: 10.1109/Humanoids57100.2023.10375185. 3.2.1

[5] Cornelia Bauer, Dominik Bauer, Alisa Allaire, Christopher G. Atkeson, and Nancy Pollard. Learning to navigate by pushing. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 171–177, 2022. doi: 10.1109/ICRA46639.2022.9812194. 3.2.1

[6] Guy Campion and Woojin Chung. Wheeled rob 17. wheeled robots. *Part B*. 2.4.1

[7] Jared Di Carlo, Patrick M. Wensing, Benjamin Katz, Gerardo Bledt, and Sangbae 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)*, pages 1–9, 2018. doi: 10.1109/IROS.2018.8594448. 3.1

[8] Farbod Farshidian et al. OCS2: An open source library for optimal control of switched systems. [Online]. Available: https://github.com/leggedrobotics/ocs2. 3.7.1

[9] Zipeng Fu, Tony Z. Zhao, and Chelsea Finn. Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation. In *Conference on Robot Learning (CoRL)*, 2024. 1, 3.1

[10] Yusuke Fujimoto and Toshiyuki Murakami. An improvement method of compliance control in pushing operation by mobile manipulator. In *IECON 2010 - 36th Annual Conference on IEEE Industrial Electronics Society*, pages 1447–1452, Glendale, AZ, USA, November 2010. IEEE. ISBN 978-1-4244-5225-5. doi: 10.1109/IECON.2010.5675469. 2.2

[11] Ruben Grandia, Fabian Jenelten, Shaohui Yang, Farbod Farshidian, and Marco Hutter. Perceptive locomotion through nonlinear model-predictive control. *IEEE Transactions on Robotics*, 39(5):3402–3421, 2023. doi: 10.1109/TRO.2023.3275384. 3.1

[12] Gaël Guennebaud, Benoît Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010. 3.7.1

[13] Louis Hawley and Wael Suleiman. Control strategy and implementation for a humanoid robot pushing a heavy load on a rolling cart. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4997–5002, September 2017. doi: 10.1109/IROS.2017.8206382. 2.2

[14] Marco Hutter, Christian Gehring, Dominic Jud, Andreas Lauber, C. Dario Bellicoso, Vassilios Tsounis, Jemin Hwangbo, Karen Bodie, Peter Fankhauser, Michael Bloesch, Remo Diethelm, Samuel Bachmann, Amir Melzer, and Mark Hoepflinger. Anymal - a highly mobile and dynamic quadrupedal robot. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 38–44, 2016. doi: 10.1109/IROS.2016.7758092. 1, 3.1

[15] Edo Jelavic, Kaixian Qu, Farbod Farshidian, and Marco Hutter. Lstp: Long short-term motion planning for legged and legged-wheeled systems. *IEEE Transactions on Robotics*, 39(6):4190–4210, 2023. doi: 10.1109/TRO.2023.3302239. 3.1

[16] T.B. Lauwers, G.A. Kantor, and R.L. Hollis. A dynamically stable single-wheeled mobile robot with inverse mouse-ball drive. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 2884–2889, 2006. doi: 10.1109/ROBOT.2006.1642139. 2.2

[17] Simon Le Cleac'h, Taylor A. Howell, Shuo Yang, Chi-Yen Lee, John Zhang, Arun Bishop, Mac Schwager, and Zachary Manchester. Fast contact-implicit model predictive control. *IEEE Transactions on Robotics*, 40:1617–1629, 2024. doi: 10.1109/TRO.2024.3351554. 3.2.2

[18] Zachary Manchester, Neel Doshi, Robert J Wood, and Scott Kuindersma. Contact-implicit trajectory optimization using variational integrators. *The International Journal of Robotics Research*, 38(12-13):1463–1476, 2019. 3.2.2

[19] Eitan Marder-Eppstein, Eric Berger, Tully Foote, Brian Gerkey, and Kurt Konolige. The office marathon: Robust navigation in an indoor office environment. In *2010 IEEE International Conference on Robotics and Automation*, pages 300–307, 2010. doi: 10.1109/ROBOT.2010.5509725. 2.6.4

[20] Matthew T. Mason, J. Kenneth Salisbury, and Joey K. Parker. Robot Hands and the Mechanics of Manipulation. *Journal of Dynamic Systems, Measurement, and Control*, 111(1):119–119, 03 1989. ISSN 0022-0434. doi: 10.1115/1.3153010. URL `https://doi.org/10.1115/1.3153010`. 3.1

[21] Carlos Mastalli, Ioannis Havoutis, Michele Focchi, Darwin G. Caldwell, and Claudio Semini. Hierarchical planning of dynamic movements without scheduled contact sequences. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4636–4641, 2016. doi: 10.1109/ICRA.2016.7487664. 3.2.2

[22] Nandagopal Methil and Ranjan Mukherjee. Pushing and steering wheelchairs using a holonomic mobile robot with a single arm. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5781–5785, Beijing, China, October 2006. IEEE. ISBN 978-1-4244-0258-8 978-1-4244-0259-5. doi: 10.1109/IROS.2006.282387. 2.2

[23] Nandagopal S. Methil and Ranjan Mukherjee. Adaptive control of a wheelchair-pushing holonomic robot subject to input constraints. In *Unmanned Systems Technology X*, volume 6962, pages 629–640. SPIE, April 2008. doi: 10.1117/12.777213. 2.2

[24] Igor Mordatch, Zoran Popović, and Emanuel Todorov. Contact-invariant optimization for hand manipulation. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '12, page 137–144, Goslar, DEU, 2012. Eurographics Association. ISBN 9783905674378. 3.2.2

[25] Igor Mordatch, Emanuel Todorov, and Zoran Popović. Discovery of complex behaviors through contact-invariant optimization. *ACM Trans. Graph.*, 31 (4), jul 2012. ISSN 0730-0301. doi: 10.1145/2185520.2185539. URL `https://doi.org/10.1145/2185520.2185539`. 3.2.2

[26] Masaki Murooka, Mitsuharu Morisawa, and Fumio Kanehiro. Centroidal trajectory generation and stabilization based on preview control for humanoid multi-contact motion. *IEEE Robotics and Automation Letters*, 7(3):8225–8232, 2022. doi: 10.1109/LRA.2022.3186515. 3.1, 3.2.1

[27] Umashankar Nagarajan, George Kantor, and Ralph Hollis. The ballbot: An omnidirectional balancing mobile robot. *The International Journal of Robotics Research*, 33(6):917–930, 2014. 1, 3.1, 3.3.1

[28] Michael Neunert, Farbod Farshidian, Alexander W. Winkler, and Jonas Buchli. Trajectory optimization through contacts and automatic gait discovery for quadrupeds. *IEEE Robotics and Automation Letters*, 2(3):1502–1509, 2017. doi: 10.1109/LRA.2017.2665685. 3.2.2

[29] Michael Neunert, Markus Stäuble, Markus Giftthaler, Carmine D. Bellicoso, Jan Carius, Christian Gehring, Marco Hutter, and Jonas Buchli. Whole-body nonlinear model predictive control through contacts for quadrupeds. *IEEE Robotics and Automation Letters*, 3(3):1458–1465, 2018. doi: 10.1109/LRA.2018.2800124. 3.2.2

[30] Shunichi Nozawa, Toshiaki Maki, Mitsuharu Kojima, Shigeru Kanzaki, Kei Okada, and Masayuki Inaba. Wheelchair support by a humanoid through integrating environment recognition, whole-body control and human-interface behind the user. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1558–1563, September 2008. doi: 10.1109/IROS.2008.4650903. 1

[31] Shunichi Nozawa, Yohei Kakiuchi, Kei Okada, and Masayuki Inaba. Controlling the planar motion of a heavy object by pushing with a humanoid robot using dual-arm force control. In *2012 IEEE International Conference on Robotics and Automation*, pages 1428–1435, St Paul, MN, USA, May 2012. IEEE. ISBN 978-1-4673-1405-3 978-1-4673-1403-9 978-1-4673-1578-4 978-1-4673-1404-6. doi: 10.1109/ICRA.2012.6224884. 2.2

[32] David E. Orin, Ambarish Goswami, and Sung-Hee Lee. Centroidal dynamics of a humanoid robot. *Autonomous Robots*, 35:161–176, 2013. URL `https://api.semanticscholar.org/CorpusID:15473645`. 3.6.1

[33] Matteo Parigi Polverini, Arturo Laurenzi, Enrico Mingo Hoffman, Francesco Ruscelli, and Nikos G. Tsagarakis. Multi-contact heavy object pushing with a centaur-type humanoid robot: Planning and control for a real demonstrator. *IEEE Robotics and Automation Letters*, 5(2):859–866, April 2020. ISSN 2377-3766. doi: 10.1109/LRA.2020.2965906. 2.2

[34] Michael Posa, Cecilia Cantu, and Russ Tedrake. A direct method for trajectory optimization of rigid bodies through contact. *The International Journal of Robotics Research*, 33:69 – 81, 2014. URL `https://api.semanticscholar.org/CorpusID:6532910`. 3.2.2

[35] Jonathan Scholz, Sachin Chitta, Bhaskara Marthi, and Maxim Likhachev. Cart pushing with a mobile manipulation system: Towards navigation with moveable

objects. In *2011 IEEE International Conference on Robotics and Automation*, pages 6115–6120, 2011. doi: 10.1109/ICRA.2011.5980288. 2.2

[36] Martin Schulze, Friedrich Graaf, Lea Steffen, Arne Roennau, and Rüdiger Dillmann. A trajectory planner for mobile robots steering non-holonomic wheelchairs in dynamic environments. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3642–3648. IEEE. ISBN 9798350323658. doi: 10.1109/ICRA48891.2023.10161082. URL `https://ieeexplore.ieee.org/document/10161082/`. 2.6.2

[37] Martin Schulze, Friedrich Graaf, Lea Steffen, Arne Roennau, and Rüdiger Dillmann. A trajectory planner for mobile robots steering non-holonomic wheelchairs in dynamic environments. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3642–3648, London, United Kingdom, May 2023. IEEE. ISBN 9798350323658. doi: 10.1109/ICRA48891.2023.10161082. 2.2

[38] Michael Shomin, Jodi Forlizzi, and Ralph Hollis. Sit-to-stand assistance with a balancing mobile robot. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3795–3800, May 2015. doi: 10.1109/ICRA.2015.7139727. 2.2

[39] Roberto Shu. An agile and dexterous balancing mobile manipulator robot. 2.4.2

[40] Roberto Shu and Ralph 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)*, pages 499–504. IEEE. ISBN 978-1-5386-7630-1. doi: 10.1109/Humanoids43949.2019.9034999. URL `https://ieeexplore.ieee.org/document/9034999/`. 2.2, 2.3.3

[41] Roberto Shu and Ralph 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)*, pages 499–504, 2019. doi: 10.1109/Humanoids43949.2019.9034999. 3.1

[42] Roberto Shu and Ralph Hollis. Momentum based whole-body optimal planning for a single-spherical-wheeled balancing mobile manipulator. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3221–3226, Prague, Czech Republic, September 2021. IEEE. ISBN 978-1-66541-714-3. doi: 10.1109/IROS51168.2021.9636752. 2.2

[43] Jean-Pierre Sleiman, Farbod Farshidian, Maria Vittoria Minniti, and Marco Hutter. A unified mpc framework for whole-body dynamic locomotion and manipulation. *IEEE Robotics and Automation Letters*, 6(3):4688–4695, 2021. doi: 10.1109/LRA.2021.3068908. 3.1

[44] Steve Tonneau, Andrea Del Prete, Julien Pettré, Chonhyon Park, Dinesh Manocha, and Nicolas Mansard. An efficient acyclic contact planner for

multiped robots. *IEEE Transactions on Robotics*, 34(3):586–601, 2018. doi: 10.1109/TRO.2018.2819658. 3.1, 3.2.1

[45] Michael R. Turski, Joseph Norby, and Aaron 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)*, pages 2376–2383, 2023. doi: 10.1109/IROS55552.2023.10342296. 3.2.2

[46] Bhaskar Vaidya, Michael Shomin, Ralph Hollis, and George Kantor. Operation of the ballbot on slopes and with center-of-mass offsets. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2383–2388, 2015. doi: 10.1109/ICRA.2015.7139516. 2.7

[47] Jean Chagas Vaz and Paul Oh. Material handling by humanoid robot while pushing carts using a walking pattern based on capture point. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9796–9801, May 2020. doi: 10.1109/ICRA40945.2020.9196872. 2.2

[48] Shihao Wang and Kris 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)*, pages 3092–3098. IEEE, 2018. ISBN 978-1-5386-3081-5. doi: 10.1109/ICRA.2018.8460500. URL https://ieeexplore.ieee.org/document/8460500/. 3.2.1

[49] Chong Zhang, Wenli Xiao, Tairan He, and Guanya Shi. Wococo: Learning whole-body humanoid control with sequential contacts, 2024. URL https://arxiv.org/abs/2406.06005. 1

[50] John Z. Zhang, Shuo Yang, Gengshan Yang, Arun L. Bishop, Swaminathan Gurumurthy, Deva Ramanan, and Zachary Manchester. Slomo: A general system for legged robot motion imitation from casual videos. *IEEE Robotics and Automation Letters*, 8(11):7154–7161, 2023. doi: 10.1109/LRA.2023.3313937. 3.2.2