2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Feedback Whole-Body Control of Wheeled Inverted Pendulum Humanoids Using Operational Space

Muhammad Ali Murtaza[1],Vahid Azimi[1] and Seth Hutchinson[1]

***Abstract*— We present a hierarchical framework for trajectory optimization and optimal feedback whole-body control of wheeled inverted pendulum (WIP) humanoid robot. The framework extends rapidly exponentially stabilizing control Lyapunov functions (RES-CLF) to operational space for controlling WIP humanoid robots while utilizing a hierarchical framework to compute an optimal policy. The upper level of the hierarchy encodes locomotion tasks, while the lower level incorporates the full system dynamics, including manipulation tasks to be performed. The framework computes an optimal policy directly in the operational space. Thus it avoids computing inverse kinematics or inverse dynamics explicitly. The framework can handle torque and task constraints while guaranteeing exponential convergence and min-norm control from RES-CLF. The efficacy of the framework is demonstrated on 18 degrees of freedom (DoF) WIP humanoid robot, Golem Krang, and 7 DoF planar WIP humanoid robot.**

I. INTRODUCTION

Wheeled Inverted Pendulum (WIP) robots have received significant attention recently. These robots share a number of attributes with their bipedal robot counterparts. For example, both can dynamically adjust their centers of mass, and hence can lift and deal with heavy payloads. An advantage for WIP robots is that their wheels enable fast and efficient locomotion, which remains a challenging task for bipedal robots. For this reason, WIP robots have been applied in variety of applications such as the Segway personal transporter [1], [2], WIP-based transporters [3], [4], WIP-based wheelchairs [5], and WIP-based humanoid robots [6]–[8]. WIP humanoids may also include one or more redundant manipulators, enabling them to perform manipulation tasks, potentially involving heavy payloads that require large forces.

For all of their advantages, however, WIP humanoids face a unique set of challenges. They have highly nonlinear, unstable, under-actuated dynamics. Most of the literature deals with this problem by considering the upper manipulator control as separate to the control of the lower body. The lower body is considered as a simplified model with one link, and control policy is designed for this system [9], [10]. A significant drawback to this approach is that the upper body motion is not exploited to compensate for the pitch changes induced by the forward motion of the wheels. This becomes significant if the robot is tasked with safety-critical or precision-critical tasks, or if there is a sudden change in center of mass (COM) of the robot due to the introduction of large transient forces. Hence there is a need for a unified approach to address locomotion and manipulation tasks. The unified whole-body approach has been demonstrated on several WIP humanoid robots [8], [11]–[13]; however, most of these robots have single manipulator [12], [13] and those with double manipulators have at most 10 degrees of freedom [11]. In contrast, our proposed feedback whole-body control is applicable to system with many more DoF and we demonstrate this on Golem Krang with full 18-DoF [14].

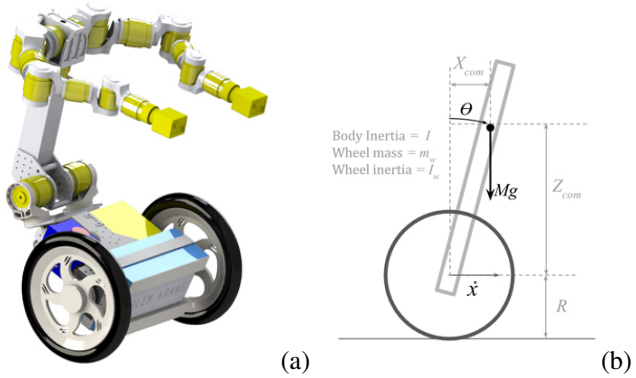

Fig. 1: WIP Humanoid Robot Golem Krang [14].

In this paper, we present an approach to trajectory optimization and feedback control for a WIP humanoid robot that builds on previous work reported by Zafar [8], [15] for the robot Golem Krang shown in Figure 1. In Zafar's approach, differential dynamic programming (DDP) was used at the high level to optimize a trajectory for a simplified system model, and a corresponding control was derived using an MPC-based approach [8] as well as an LQR-based approach [15]. At the lower level, a QP-based approach was used to compute joint accelerations satisfying different tasks. Then these joint accelerations were applied to isolated manipulator dynamics, which were obtained by elimination of the wheel dynamics from the overall system dynamic model. It combined different tasks by assigning different weights in the objective function of QP. Adding weights to the objective function restricts the search space for the nonlinear programming solver, and it may even produce an unrealistic solution with the wrong set of weights [16]. In case of multiple tasks to be simultaneously satisfied, choosing the correct weights is not trivial [17]. This also meant re-tuning of the weights if the robot is assigned a different task. Besides, there is no guarantee regarding the exponential stability of the system.

Our work extends the work of [8], [15] in two ways. First, we formulate the high-level control problem as a

[1]Muhammad Ali Murtaza, Vahid Azimi and Seth Hutchinson are with the Institute of Robotics and Intelligent Machines at the Georgia Institute of Technology, Atlanta, GA, 30332, USA. email: mamurtaza@gatech.edu, vahid.azimi@gatech.edu, seth@gatech.edu

time varying LQR problem. Our approach is inspired by [18], which proposed LQR-trees comprised of sparse LQR stabilizing trajectories, and which provided performance guarantees by verifying regions of attraction using sum-of-squares (SOS) programming to certify Lyapunov functions. Second, we formulate the low-level control problem using rapidly exponentially stabilizing control Lyapunov function (RES-CLF) [19]. RES-CLF computes point-wise min-norm control and ensures exponential convergence. Therefore, it is also more robust than traditional feedback linearized controllers [20] and has been applied to control bipedal robotics [21]. However, RES-CLF computes control in the joint space and, therefore, cannot be applied to redundant robotic manipulators without explicitly computing inverse dynamics. We extend this formulation to whole-body control for WIP humanoids with redundant manipulators using operational state-space. The formulation can satisfy torque constraints along with task constraints without specifying any weights to the task while computing point-wise min-norm control and ensuring exponential convergence. This implies that WIP humanoid can perform multiple tasks without any changes to the original formulation. To demonstrate the efficacy of our approach, we report results for an 18 DoF WIP humanoid Golem Krang. We also compare our approach with results reported in [8] on the planar model of Golem Krang.

The remainder of paper is organized as follow. Section II discusses system model of the simplified robot as well as complete robot. Section III explains feedback motion policy in hierarchical control framework, as well as input/output feedback linearization in operational space. Results and analysis are presented in section IV on 7-DOF planar humanoid robot model and 18-DoF 3-D Golem Krang. Finally, paper is concluded in section VI.

II. System Modeling

To perform whole-body control of a WIP humanoid, we assume a hierarchical methodology as described in [8] consisting of two hierarchy: low-level hierarchy, which performs control on a complete model, and a high-level hierarchy, which performs planning on a simplified model of the robot. The complete dynamical modeling of our target system is derived in [15]. This section briefly discusses the hierarchical methodology and modeling for the sake of clarity.

A. Full System Dynamic Model

Our WIP humanoid robot is highly redundant manipulator designed on differential drive wheeled robot. We refer the first link of the robot as the base link and the rest of the complete structure except the wheels is referred to as the body. The full 3-D dynamic model can then be represented by the differential equation:

$$M(q)\begin{bmatrix}\ddot{x}\\ \ddot{\psi}\\ \ddot{q}\end{bmatrix} + H(q,\dot{q}) - \Gamma_{fric} = B\tau \tag{1}$$

where $H(q,\dot{q}) = C(q,\dot{q})\begin{bmatrix}\dot{x} & \dot{\psi} & \dot{q}\end{bmatrix}^T + G(q)$ for notation simplicity. Here M is the inertia matrix, C is the Coriolis matrix, G is the vector related to gravity related term, Γ_{fric} is the friction torque, $\dot{x}$ and $\ddot{x}$ are the base heading velocity and acceleration, $\dot{\psi}$ and $\ddot{\psi}$ is the robot spin velocity and acceleration, $q, \dot{q}$ and $\ddot{q}$ are the angular position, angular velocity and angular acceleration associated with each joint with respect to the reference frame, and $\tau = [\tau_L\ \tau_R\ \tau_2 \cdots\ \tau_n]^T$ is the vector of torques acting on the joints, where τ_L and τ_R are the wheel torques. B is given as

$$B = \begin{bmatrix}\bar{B} & 0_{3\times n}\\ 0_{(n-1)\times 2} & I_{n-1}\end{bmatrix}, \qquad \bar{B} = \begin{bmatrix}\frac{1}{R} & \frac{1}{R}\\ \frac{-L}{2R} & \frac{L}{2R}\\ -1 & -1\end{bmatrix} \tag{2}$$

where L and R are the distance between wheel center and radius of the wheels respectively. For the case of Golem Krang, wheel motors, τ_L and τ_R, are responsible for both the pitch angle as well as position of the robot, hence it is under actuated in nature. In addition, there is a reaction torque of the wheel motors, since the motors are directly mounted on the link. We can then define $\tau_1 = -(\tau_L + \tau_R)$, $\tau_0 = \frac{L}{2R}(\tau_L - \tau_R)$ and $\Gamma = [\tau_1\ \cdots\ \tau_n]^T$ to write system dynamics as

$$M(q)\begin{bmatrix}\ddot{x}\\ \ddot{\psi}\\ \ddot{q}\end{bmatrix} + H(q,\dot{q}) - \Gamma_{fric} = \begin{bmatrix}-\frac{\tau_1}{R}\\ \tau_0\\ \Gamma\end{bmatrix} \tag{3}$$

For a complete derivation, please see [8], [15].

In order to apply operational space control on our WIP humanoid, we isolate manipulator dynamics and this results in [8]

$$\mathcal{M}\ddot{q} + \kappa\left(H(q,\dot{q}) - \Gamma_{fric}\right) = \Gamma \tag{4}$$

where $\mathcal{M}$ is the modified inertia matrix and $\kappa\left(H(q,\dot{q}) - \Gamma_{fric}\right)$ is the modified Coriolis and gravity related term. The complete derivation and detail is in [7], [15]. It has been shown in [8] that the driving force of $\ddot{x}$ is the body-weight torque and the parameters $(\dot{q}, \ddot{q})$ capture that motion. Hence we can approximate it as a simplified model.

B. Simplified Model

The high-level hierarchy utilizes a simplified model of WIP humanoid. The simplified model is considered to be a wheeled inverted pendulum with just one link, shown in Figure 1(b). It can also be defined as a cart pole model, where a rigid body has to balance itself on the wheel or cart. We utilized the same model given in [15], where state space of the simplified robot is given as $X = \begin{bmatrix}x_0\ y_0\ \psi\ \theta\ \dot{x}\ \dot{\psi}\ \dot{\theta}\end{bmatrix}^T$. Here (x_0, y_0) represents the position of the midpoint of the simplified robot in the world, $\dot{x}$ represents the heading velocity, θ and $\dot{\theta}$ are the inclination position and velocity, and ψ and $\dot{\psi}$ are the spin angle and velocity of the simplified model with respect to the world frame. The state space for the dynamics can be written as $\dot{X} = f(X,u)$, and we will refer to it as such from now on. Here u represents the inclination acceleration, and the complete derivation can be found in [15]. The center of the mass trajectory of the simplified model is then given by

$$\begin{bmatrix}X_{COM}\\ Y_{COM}\\ Z_{COM}\end{bmatrix} = \begin{bmatrix}x_o + d\sin(\theta)\cos(\psi)\\ y_o + d\sin(\theta)\sin(\psi)\\ R + d\cos(\theta)\end{bmatrix} \tag{5}$$

where d is the center of mass of the body link of the simplified model in the body frame, θ is the angle of inclination and ψ is the spin of the robot in the world frame.

The simplified model is also underactuated in nature. We assume frame 0 is the frame attached to the base link, defined such that the origin of the frame is located at the midpoint of the two wheels of WIP with its x-axis always along the heading direction. COM of the body link of the simplified robot is computed by adding the product of mass and COM of each link, represented in frame 0, and divide it by the total mass of all the links. If the upper body is compensating to fix the orientation and position of the end-effector, it will change the center of mass of the simplified model. Therefore the corresponding desired joint angle may also be needed to be readjusted. We will, therefore, define a feedback optimal control strategy that takes this into account in the next section.

III. Feedback Motion Control

In this section, we first define optimal motion planning in the offline setting, followed by a low-level controller and finally, feedback motion control in the high-level controller. The objective of the control design is to perform manipulation and locomotion tasks in safety-critical settings.

A. Optimal Motion Planning

The primary purpose of optimal motion planning is to design feasible motions for the COM of a simplified model. The planned trajectory should reach the destination while satisfying state constraints and torque constraints. State constraints include balancing or any additional safety constraint, and input constraint may include torque limit on wheels. For this purpose, simplified dynamics are utilized for designing optimal trajectories. The optimal control problem with the initial condition, $X(0)$, is formulated as follow

$$\begin{aligned} \underset{u}{\text{minimize}} \quad & J(u) = \int_0^{t_f} \left(u^T u\right) dt \\ \text{subject to :} \quad & \dot{X} = f(X,u), \\ & X(t_f) = X^{des}, \qquad |u(t)| \leq u_{max} \end{aligned} \tag{6}$$

where $X^{des} = \begin{bmatrix} x^{des} & y^{des} & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T$. Equation (6) is solved by first sampling the time into n_s discrete time steps. It is then solved by using either nonlinear optimization toolbox or any indirect method, like differential dynamic programming (DDP) or Pontryagin's maximum principle (PMP). There is, in general, a trade-off between using a densely sampled trajectory to get better a solution and computation time in both methods. The generated optimal trajectory serves as a reference to the low-level controller for the locomotion tasks.

B. Low Level Control

For safety-critical or precision critical tasks, we often require the precise position of the end-effectors. Hence tasks defined in joint space formulation may not be suitable, and it is often desired to design control laws directly in the operational space [22]. Each task z_τ can be performed by some configuration of joints. We can then transform joint velocity, $\dot{q}$, to task velocity, $\dot{z}_\tau$, through Jacobian matrix $J_\tau(q)$, given as

$$\dot{z}_\tau = J_\tau(q)\dot{q}. \tag{7}$$

Similarly task acceleration, $\ddot{z}_\tau$ is given as

$$\ddot{z}_\tau = \dot{J}_\tau(q)\dot{q} + J_\tau(q)\ddot{q}. \tag{8}$$

We can write $\ddot{q}$ in terms of $J_\tau(q), \dot{J}_\tau(q)$ and $\ddot{z}_\tau$ as

$$\ddot{q} = J_\tau^\dagger(q)\left[\ddot{z}_\tau - \dot{J}_\tau(q)\dot{q}\right] \tag{9}$$

where $J_\tau^\dagger(q)$ is the pseudo-inverse of the task Jacobian. In addition, we also have system dynamics given by Eq. (4). Substituting $\ddot{q}$ in Eq. (4) yields [23]

$$\mathcal{M}J_\tau^\dagger(q)\left[\ddot{z}_\tau - \dot{J}_\tau(q)\dot{q}\right] + \kappa\left(H(q,\dot{q}) - \Gamma_{fric}\right) = \Gamma. \tag{10}$$

Eq. (10) can then be written as

$$\begin{aligned} \ddot{z}_\tau &= \left(\mathcal{M}J_\tau^\dagger(q)\right)^{-1}\left[\Gamma - \kappa\left(H(q,\dot{q}) - \Gamma_{fric}\right)\right] + \dot{J}_\tau(q)\dot{q} \\ &= f(q,\dot{q}) + g(q)\Gamma \end{aligned} \tag{11}$$

where

$$\begin{aligned} f(q,\dot{q}) &= \dot{J}_\tau(q)\dot{q} - \left(\mathcal{M}J_\tau^\dagger(q)\right)^{-1}\kappa\left(H(q,\dot{q}) - \Gamma_{fric}\right) \\ g(q) &= \left(\mathcal{M}J_\tau^\dagger(q)\right)^{-1} \end{aligned}$$

Eq. (11) defines the task dynamics in terms of system dynamics. We will use it to define the notion of operational state space and design our controller based on it.

1) Operational Space: We can define operational space in terms of system dynamics, jacobian and torques. Let $y^{\tau_i} = [z_{\tau_i}, \dot{z}_{\tau_i}]$ represents the operational state space associated with task $\tau_i, i \in \{1, 2, \cdots, N\}$, where N are the total number of tasks. We can then write operational state space for task τ_i as follow

$$\begin{aligned} \dot{y_1}^{\tau_i} &= J_{\tau_i}(q)\dot{q} \\ \dot{y_2}^{\tau_i} &= f(q,\dot{q}) + g(q)\Gamma \end{aligned} \tag{12}$$

where $\dot{y_2}^{\tau_i}$ is derived using Eq. (11). We can then define virtual output $y_v^{\tau_i}$, also referred as virtual constraints, for each task τ_i to modulate the robot to perform the desired task [19]. They are termed as virtual because they are implemented through feedback control rather than through physical constraints. We can then apply input/output feedback linearization control on each virtual output to drive them to zero [24], [25]. Each virtual output is of the form

$$y_v^{\tau_i} = z_{\tau_i} \tag{13}$$

Then the time derivative of $y_v^{\tau_i}$ results in

$$\begin{aligned} \dot{y}_v^{\tau_i} &= \dot{y_1}^{\tau_i} \\ \ddot{y}_v^{\tau_i} &= \dot{y_2}^{\tau_i} = \underbrace{f(q,\dot{q})}_{L_{f_v}^\gamma} + \underbrace{g(q)}_{L_{g_v}L_{f_v}^{\gamma-1}}\Gamma \end{aligned} \tag{14}$$

where $\gamma \in \{1, 2\}$ is the relative degree,. Then we can drive virtual outputs to desired task state by defining the control law of the form [24]:

$$\Gamma = \left(L_{g_v}L_{f_v}^{\gamma-1}\right)^{-1}\left[\sum_{i=1}^{N}\left(-\left(L_{f_v}^\gamma\right)_{\tau_i} + \mu_i\right)\right] \tag{15}$$

where Γ is vector of torques defines in section II-A, $\left(L_{g_v}L_{f_v}^{\gamma-1}\right)$ is the decoupling matrix for all the tasks, L

is the Lie derivative and $\left(L_{f_v}^{\gamma}\right)_{\tau_i}$ is the Lie derivative along vector field $y_v^{\tau_i}$. If there are no torque constraints, then μ_i can be defined as [24], [25]

$$\mu_i = -K_{p_v}^{\tau_i} e^{\tau_i} - K_{d_v}^{\tau_i} \dot{e}^{\tau_i} \tag{16}$$

where $e^{\tau_i} = z_{\tau_i} - z_{\tau_i}^d$, $z_{\tau_i}^d$ is the desired task configuration for task τ_i, $K_{p_v}^{\tau_i}$ and $K_{d_v}^{\tau_i}$ are the gains associated with task state space τ_i. This formulation was designed from an input/output feedback linearization perspective with a generic constraint. It also assumes $L_{g_v} L_{f_v}^{\gamma-1}$ to be square, which may require introducing additional outputs. However, we can attain the same result, similar to Eq. (15), without any additional constraints by considering Eq. (10) and applying input of the form:

$$\Gamma = M J_{aug}^{\dagger} \left(\ddot{z}_{aug}^{des} - \dot{J}_{aug} \dot{q} \right) + \kappa \left(H(q, \dot{q}) - \Gamma_{fric} \right) \tag{17}$$

where $J_{aug} = [J_{\tau_1} \cdots J_{\tau_N}]^T$, $J_{aug}^{\dagger}$ is the pseudo inverse of the J_{aug}, and $\ddot{z}_{aug}^{des} = \left[z_{\tau_1}^d \cdots z_{\tau_N}^d \right]$ is the desired task configuration for all tasks. Applying this input to the system dynamics, given by Eq. (11), yields

$$\ddot{z}_{\tau_i} = \ddot{z}_{\tau_i}^{des} \quad \forall\ \tau_i, i \in \{1, \cdots, N\} \tag{18}$$

However, if we have torque constraints or have some requirement relating to convergence e.g. exponential convergence, then we can modify Eq. (18) as

$$\ddot{z}_{\tau_i} = \ddot{z}_{\tau_i}^{des} + \mu_i \tag{19}$$

where μ is an auxiliary control to incorporate torque limits and ensure exponential convergence. The resulting torque then takes the following form

$$\Gamma = M J_{aug}^{\dagger} \left(\ddot{z}_{aug}^{des} + \mu - \dot{J}_{aug} \dot{q} \right) + \kappa \left(H(q, \dot{q}) - \Gamma_{fric} \right) \tag{20}$$

where $\mu = [\mu_1 \ \cdots \ \mu_N]^T$. We can then reformulate it as rapidly exponentially stabilizing control Lyapunov function using quadratic programming (RES-CLF-QP) to ensure exponential convergence [19].

2) Rapidly Exponentially Stabilizing Control Lyapunov Function: Rapidly Exponentially stabilizing control Lyapunov functions, presented in [19], provide a method to simultaneously satisfy torque constraints and achieve exponential stability. First, we define a vector $\eta_i := [e^{\tau_i}; \dot{e}^{\tau_i}]^T$ for task τ_i. Then assuming that the preliminary input associated with Eq. (20) has been applied, we can write Eq. (4) as following linear system, also known as normal form in control literature [24], [25]:

$$\dot{\eta}_i = F_{\tau_i} \eta_i + G_{\tau_i} \mu_i \tag{21}$$

where F_{τ_i} and G_{τ_i} take the following form:

$$F_{\tau_i} = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix}, \quad G_{\tau_i} = \begin{bmatrix} 0 \\ I \end{bmatrix} \tag{22}$$

If the operational state space associated with task τ_i is of relative degree 1, then $\eta_i := [e^{\tau_i}]$ and F_{τ_i} and G_{τ_i} take form:

$$F_{\tau_i} = \begin{bmatrix} 0 \end{bmatrix}, \quad G_{\tau_i} = \begin{bmatrix} 1 \end{bmatrix}. \tag{23}$$

We can then transform Eq. (21) into RES-CLF-QP by method in [19]. For this, we first define control Lyapunov function of the form $V^{\tau_i}(\eta_i) = \eta_i^T P_{\tau_i} \eta_i$, where $P_{\tau_i} = P_{\tau_i}^T \succ 0$ is obtained by solving the algebraic Riccati equation:

$$F_{\tau_i}^T P_{\tau_i} + P_{\tau_i} F_{\tau_i} - P_{\tau_i} G_{\tau_i} G_{\tau_i}^T P_{\tau_i} = -Q_{\tau_i} \tag{24}$$

where $Q_{\tau_i} = Q_{\tau_i}^T \succ 0$. Then the time derivative of $V^{\tau_i}(\eta_i)$ yields:

$$\dot{V}^{\tau_i}(\eta_i) = L_F V^{\tau_i}(\eta_i) + L_G V^{\tau_i}(\eta_i) \mu_i \tag{25}$$

where L_F and L_G are lie derivatives along F_{τ_i} and G_{τ_i} and have the following values:

$$\begin{aligned} L_F V^{\tau_i}(\eta_i) &= \eta_i^T \left(F_{\tau_i}^T P_{\tau_i} + P_{\tau_i} F_{\tau_i} \right) \eta_i \\ L_G V^{\tau_i}(\eta_i) &= 2 \eta_i^T P_{\tau_i} G_{\tau_i} \end{aligned} \tag{26}$$

Rapidly exponential stability is then achieved by satisfying the following inequality condition on $V(\eta)$:

$$L_F V^{\tau_i}(\eta_i) + L_G V^{\tau_i}(\eta_i) \mu \leq -\frac{\lambda_{min}(Q_{\tau_i})}{\lambda_{max}(P_{\tau_i})} V^{\tau_i}(\eta_i) \tag{27}$$

where $\lambda_{min}(Q_{\tau_i})$ and $\lambda_{max}(P_{\tau_i})$ are the minimum and maximum eigen values of Q_{τ_i} and P_{τ_i} matrices respectively. It is shown in [19] that applying such input to robotics system results in exponential convergence. We can then design a quadratic programming based optimization problem to satisfy this constraint at each instant of time and any additional torque constraint optimally as follows:

$$\begin{aligned} &\mu^* = \arg\min_{\mu} \ \ \mu^T \mu \\ &\text{subject to :} \\ &M J_{aug}^{\dagger} \left(\ddot{z}_{\tau}^{des} + \mu - \dot{J}_{aug} \dot{q} \right) + \kappa \left(H(q, \dot{q}) - \Gamma_{fric} \right) \leq u_m \\ &- M J_{aug}^{\dagger} \left(\ddot{z}_{\tau}^{des} + \mu - \dot{J}_{aug} \dot{q} \right) - \kappa \left(H(q, \dot{q}) - \Gamma_{fric} \right) \leq u_m \\ &L_F V^{\tau_i}(\eta_i) + L_G V^{\tau_i}(\eta_i) \mu_i \leq -\frac{\lambda_{min}(Q_{\tau_i})}{\lambda_{max}(P_{\tau_i})} V^{\tau_i}(\eta_i) \\ &\forall \ \ \tau_i, \quad i = [1, \cdots, N] \end{aligned} \tag{28}$$

where $\mu = [\mu_1 \ \cdots \ \mu_N]^T$ and u_m is the vector indicating maximum torque limit for each joint. The RES-CLF-QP optimization problem is solved for a single time step to generate torques and computes point-wise min-norm control. It computes a local optimal solution satisfying the constraints. Task prioritization and QP formulation feasibility has been briefly discussed in section V.

Resulting solution of RES-CLF-QP changes the joint positions, hence changing the center of mass of simplified robot model. Therefore, we need a feedback tracking controller to compensate for deviation in planned trajectory.

C. Time-Varying LQR

The objective of the low-level controller is to follow the trajectory of the base angle computed by high-level control on a simplified model. However, the dynamics of the simplified model changes with the upper body compensating for the end effector. That means that a feedback law must also compensate for the corresponding trajectory. The optimal trajectory generated, discussed in section III-A, serves as a reference trajectory over finite time interval $[0, t_f]$.

Let us represent nominal trajectory as $X_0(t) : [0, T] \rightarrow \mathbb{R}^n$ and $u_0 : [0, T] \rightarrow \mathbb{R}^m$ be the nominal controller for the simplified WIP model. Let $\bar{X}(t) = X(t) - X_0(t)$ and $\bar{u}(t) = u(t) - u_0(t)$ represent new states and inputs defining the deviation from the nominal state and nominal input, then we can define the system dynamics in new coordinates as

$$\begin{aligned}\dot{\bar{X}}(t) &= \dot{X}(t) - \dot{X}_0(t) = f(X, u) - [f(X_0), u_0)] \\ &= f(X_0 + \bar{X}, u_0 + \bar{u}) - f(X_0, u_0)\end{aligned} \tag{29}$$

The dynamical equation can then be approximated as

$$\dot{\bar{X}}(t) \approx A(t)\bar{X}(t) + B(t)\bar{u}(t) \tag{30}$$

where $A(t)$ and $B(t)$ are time varying matrices given as

$$A(t) = \frac{\partial f}{\partial X}\Big|_{\substack{X(t)=X_0(t) \\ u(t)=u_0(t)}}, B(t) = \frac{\partial f}{\partial u}\Big|_{\substack{X(t)=X_0(t) \\ u(t)=u_0(t)}} \tag{31}$$

We can then design a quadratic regulator cost to drive the error to zero as

$$\begin{aligned}J(\bar{X}, \bar{u}, t) = &\int_t^{t_f} \left[\bar{X}^T(t) Q \bar{X}(t) + \bar{u}^T(t) R \bar{u}(t)\right] dt \\ &+ \bar{X}^T(t_f) Q_f \bar{X}(t_f)\end{aligned} \tag{32}$$

where Q and Q_f are positive semi-definite and symmetric matrices and R is a positive definite symmetric matrix. The optimal cost-to-go takes the form

$$J^*(\bar{X}, t) = \bar{X}^T(t) S(t) \bar{X}(t) \tag{33}$$

where $S(t)$ is computed by solving the differential Riccati Equation given by

$$\dot{S} = -\left(A^T S + SA - SBR^{-1}B^T S + Q\right) \tag{34}$$

with boundary condition $S(t_f) = Q_f$. The time varying gain $K(t)$ is then given as

$$K(t) = R^{-1} B^T(t) S(t) \tag{35}$$

and the optimal control policy is then given as

$$u(t) = u_0 - K(t)\bar{X}(t). \tag{36}$$

TVLQR gains are computed by linearizing the simplified model along the nominal trajectory. However, COM of the simplified dynamic model is changed due to change in joint position to compensate the end-effector. This deviation from the original model can be seen as a disturbance. However, LQR gain will only be able to compensate for the changing dynamics if the state lies in its region of attraction. Region of attraction is the set of points that asymptotically converge to the origin. In principle, we can compute the region of attraction using sum of squares (SOS) based approaches [18], [26]. We discuss this further in section V.

Fig. (2) shows the complete architecture comprising of optimal motion and feedback controller in whole body control framework.

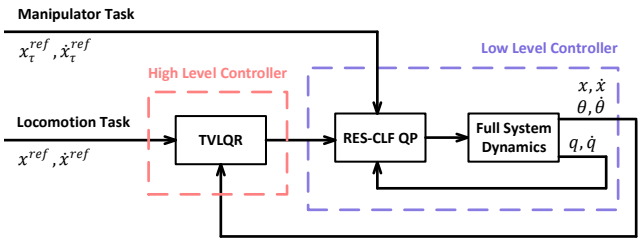

Fig. 2: Block Diagram of Low Level and High Level Controller, comprising of RES-CLF-QP and TVLQR respectively.

IV. Results

The presented framework is applied to the 18 DoF WIP humanoid robot, Golem Krang, using Dynamic Animation and Robotics Toolkit (DART) physics engine [27]. The objectives of the robot were as follows: a) Move the left end-effector to the target position while keeping the right end-effector fixed at a particular position and maintaining the balance of the robot. The left end-effector movement should also maintain orientation during the movement. b) Carry a cup on the tray from the initial position to the desired goal position such that the cup do not fall from the tray. The snapshots of Golem Krang carrying a tray with a cup is given in Fig. (3). Moreover, the complete video of both tasks is submitted as a video submission as well as available online [28], and it shows that the robot is successfully able to perform both tasks without any change in formulation in the low-level hierarchy.

For the sake of illustration, we also applied the framework on 7-DOF planar robot, i.e., a robot with six serial joints attached to the wheels and compared it with [8]. It is a planar version of the Golem Krang, shown in Figure 1. The objective is to maintain end-effector at a fixed position and orientation relative to the body frame, while the robot moves from the position, $x = 0$, to the position, $x = 5$. Initial values of state for complete robot are $\left[x(0)\ q(0)^T\ \dot{x}\ \dot{q}(0)^T\right]^T = \left[0\ -24.3^o\ 56.2^0\ 138.7^o\ -21.2^o\ -21.2^o\ -21.2^o\ 0_{1\times7}\right]$. The initial state for the simplified model was $\left[x\ \theta\ \dot{x}\ \dot{\theta}\right] = 0_{4\times1}$.

For optimal motion planning, we used differential dynamic programming to yield an initial optimal trajectory [29]. For both DDP as well as TVLQR, we assign high value to the final step and nominal value to intermediate steps. The time step for TVLQR was set to 0.1 sec, while low-level QP operated at 0.01 sec. The final time, t_f, was set to 20 sec, but the task was completed in 6 secs. This is due to the high cost associated with the final state.

Tasks for the low-level controller were the following: a) follow the reference trajectory $\ddot{\theta}^{ref}$ generated by the high-level controller, b) maintain the end-effector position while moving, c) maintain the orientation of the end-effector. To compare it with [8], we defined $\ddot{z}_{\tau_i}^{des}$ to take the form of Eq. (16) with the same gains as of [8]. End-effector desired position and orientation were set to some initial value, and

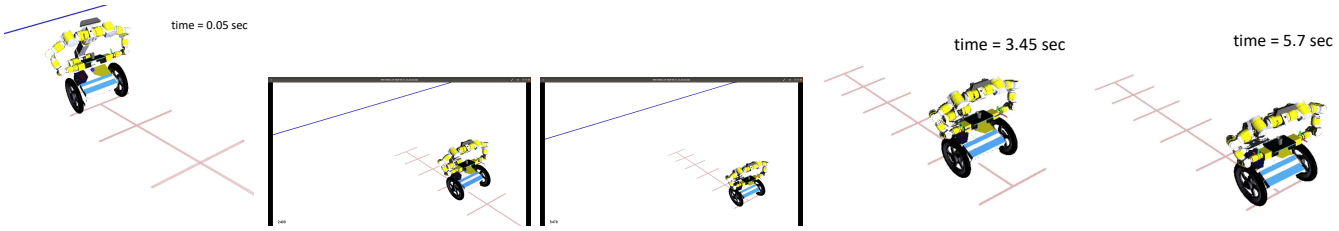

Fig. 3: Snapshots of 18-DoF Golem Krang carrying a tray with a cup at five different instants.

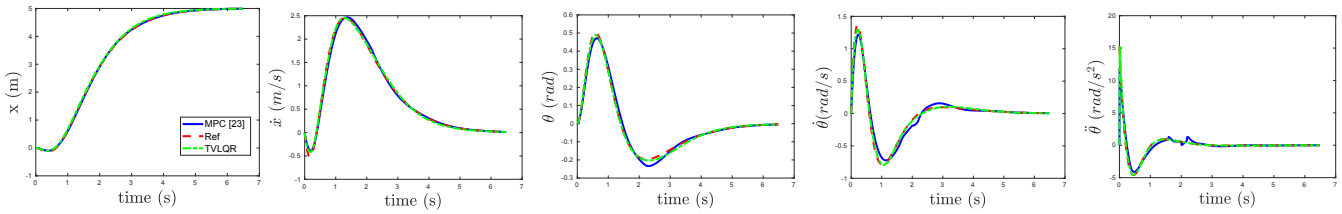

Fig. 4: Reference and State Trajectory of Simplified Model showing (a) Robot Position x, (b) COM Angle θ, (c) Robot Speed $\dot{x}$ (d) COM Angular Speed $\dot{\theta}$ (e) COM Angular Acceleration $\ddot{\theta}$ respectively from left to right. Blue line represents MPC, red line represents the nominal reference trajectory and green line represents TVLQR.

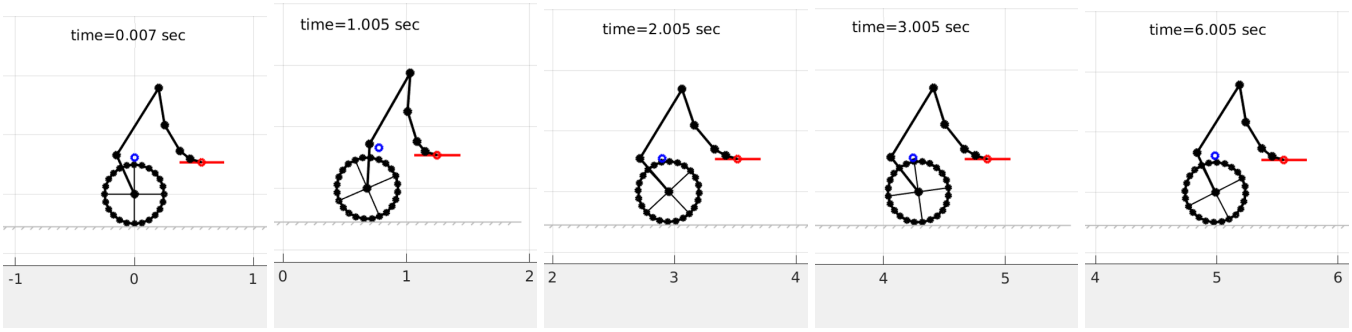

Fig. 5: Snapshots of the full body at 6 different instant. Blue dot represents the COM of robot and red line represents the tray attached to the end-effector at fixed orientation.

their corresponding velocity and accelerations were set to zero. Similarly, $\theta^{ref}, \dot{\theta}^{ref}$ and $\ddot{\theta}^{ref}$ were generated by high-level controller.

Fig. (4) shows the trajectories of the simplified model generated by the high-level controller and actual trajectory of the simplified model after low-level RES-CLF-QP yields the result on the complete model. It also compares the trajectories generated by [8]. Although we see that TVLQR and RES-CLF-QP yield smooth trajectories and better tracking, the improvement by the TVLQR and RES-CLF-QP is minimal as compared to [8] for the same gains. However, the strength of the formulation lies in its more general setting with the ability to accommodate more tasks without any weights re-tuning and its theoretical guarantees. This is evident from the 3D simulation, where we perform two tasks without changing the formulation in the low-level hierarchy. Complete simulation results are shown in video submission supplementing this submission and available online at https://youtu.be/MK95Pv_f21A.

V. Discussion

For this paper, we computed region of attraction using SOS for just the boundary points at the time, $t = 0$, and time, $t = t_f$, to save computational effort. In general, we want to compute a bounded region around nominal trajectory that is an inner approximation of the true region of attraction [18]. We can therefore define a sub-level set $B(\rho, t) = X|\widetilde{V}(X) \leq \rho\}$ such that

$$X(t) \in B(\rho, t), X \neq 0 \implies \widetilde{V}(X) > 0, \dot{\widetilde{V}}(X) < 0 \quad (37)$$

where $\widetilde{V} = \bar{X}^T \bar{S} \bar{X}$ and $\bar{S}$ is the normalized solution of differential Riccati equation, S, given by Eq. (34). The time derivative of $\widetilde{V}$ is given as

$$\dot{\widetilde{V}} = 2\bar{X}^T \bar{S} \hat{f}\left(X_0 + \bar{X}, u_0 - K\bar{X}\right) \quad (38)$$

where $\hat{f}$ is a polynomial approximation of f using Taylor expansion. Then we can compute largest region $B(\rho, t)$ by solving the following optimization problem [18]:

$$\begin{aligned} &\max\ \rho \qquad \text{subject to} \\ &-\dot{\widetilde{V}} + N(\bar{X})\left(\widetilde{V} - \rho\right) \quad SOS \\ &N(\bar{X}) \quad SOS \end{aligned} \quad (39)$$

where $N(\bar{X})$ is the non-negative Lagrange multiplier term. The above optimization problem is not convex in general since it is bilinear in decision variable, $N(\bar{X})$ and ρ. However, we can compute the solution by alternating between the

two set by fixing one and computing other until convergence is achieved [17]. Due to the computational complexity of SOS solvers, the region of attraction cannot be done in real-time. However, we can compute the region of attraction for each instant of time in an offline manner and store them in a look up table for verification.

RES-CLF-QP framework can also be extended to define task prioritization in case of conflicting tasks. This can be done by adding a relaxation term to the task with low priority in the QP formulation, given by Eq. (28). This ensures that RES-CLF-QP always yields a solution [30]. It can also be extended to incorporate limits on the joint velocity and incorporate safety by designing the control barrier function in operational space. We have also assumed that all the states of the robot are observable and provided by the physics engine. However, in the real world, some of the states like the position of the robot may need to be estimated. In that case, we will need observers to be incorporated in the control framework.

VI. CONCLUSIONS

We presented a feedback whole-body control framework, where TVLQR provided feedback and disturbance were quantified by the idea of the region of attraction. We also proposed the RES-CLF-QP framework based on operational state space. RES-CLF-QP framework offers pointwise min-norm control and ensures exponential convergence. It also circumvents the weights for joining different objectives. The framework offers a generalized approach to combine and control different tasks for complex robots. The framework was performed on a scaled-down 7-DoF planar robot and 18-DoF complete robot, and analysis was performed with the previous work of [8].

ACKNOWLEDGMENT

For some aspects of this project, we used code that was developed in conjunction with the research described in [15].

REFERENCES

[1] D. Kamen, "The segway® personal transporter (pt), the first selfbalancing, zero emissions personal transportation vehicle," 2001.

[2] D. Voth, "Segway to the future [autonomous mobile robot]," *IEEE Intelligent Systems*, vol. 20, no. 3, pp. 5–8, 2005.

[3] P. Petrov and M. Parent, "Dynamic modeling and adaptive motion control of a two-wheeled self-balancing vehicle for personal transport," in *13th International IEEE Conference on Intelligent Transportation Systems*. IEEE, 2010, pp. 1013–1018.

[4] L. Vermeiren, A. Dequidt, T. M. Guerra, H. Rago-Tirmant, and M. Parent, "Modeling, control and experimental verification on a two-wheeled vehicle with free inclination: An urban transportation system," *Control Engineering Practice*, vol. 19, no. 7, pp. 744–756, 2011.

[5] Y. Takahashi, S. Ogawa, and S. Machida, "Front wheel raising and inverse pendulum control of power assist wheel chair robot," in *IECON'99. Conference Proceedings. 25th Annual Conference of the IEEE Industrial Electronics Society (Cat. No. 99CH37029)*, vol. 2. IEEE, 1999, pp. 668–673.

[6] S. Jeong and T. Takahashi, "Wheeled inverted pendulum type assistant robot: inverted mobile, standing, and sitting motions," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2007, pp. 1932–1937.

[7] M. Zafar and H. I. Christensen, "Whole body control of a wheeled inverted pendulum humanoid," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2016, pp. 89–94.

[8] M. Zafar, S. Hutchinson, and E. A. Theodorou, "Hierarchical optimization for whole-body control of wheeled inverted pendulum humanoids," in *2019 IEEE International Conference on Robotics and Automation*, 2019.

[9] C.-C. Tsai, H.-C. Huang, and S.-C. Lin, "Adaptive neural network control of a self-balancing two-wheeled scooter," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 4, pp. 1420–1428, 2010.

[10] M. Stilman, M. Zafar, C. Erdogan, P. Hou, S. Reynolds-Haertle, and G. Tracy, "Robots using environment objects as tools the macgyverparadigm for mobile manipulation," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 2568–2568.

[11] G. Zambella, G. Lentini, M. Garabini, G. Grioli, M. G. Catalano, A. Palleschi, L. Pallottino, A. Bicchi, A. Settimi, and D. Caporale, "Dynamic whole-body control of unstable wheeled humanoid robots," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3489–3496, 2019.

[12] E. Guizzo and E. Ackerman, "Boston dynamics officially unveils its wheel-leg robot: best of both worlds," *Retrieved February*, vol. 19, p. 2018, 2017.

[13] M. V. Minniti, F. Farshidian, R. Grandia, and M. Hutter, "Whole-body mpc for a dynamically stable mobile manipulator," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3687–3694, 2019.

[14] M. Stilman, J. Olson, and W. Gloss, "Golem krang: Dynamically stable humanoid robot for mobile manipulation," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 3304–3309.

[15] M. Zafar, "Whole body control of wheeled inverted pendulum humanoids," Ph.D. dissertation, Georgia Institute of Technology, 2019.

[16] A. W. Winkler, C. D. Bellicoso, M. Hutter, and J. Buchli, "Gait and trajectory optimization for legged systems through phase-based end-effector parameterization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1560–1567, 2018.

[17] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[18] R. Tedrake, I. R. Manchester, M. Tobenkin, and J. W. Roberts, "Lqr-trees: Feedback motion planning via sums-of-squares verification," *The International Journal of Robotics Research*, vol. 29, no. 8, pp. 1038–1052, 2010.

[19] A. D. Ames, K. Galloway, K. Sreenath, and J. W. Grizzle, "Rapidly exponentially stabilizing control lyapunov functions and hybrid zero dynamics," *IEEE Transactions on Automatic Control*, vol. 59, no. 4, pp. 876–891, 2014.

[20] S. Kolathaya and A. D. Ames, "Parameter to state stability of control lyapunov functions for hybrid system models of robots," *Nonlinear Analysis: Hybrid Systems*, vol. 25, pp. 174–191, 2017.

[21] K. Galloway, K. Sreenath, A. D. Ames, and J. W. Grizzle, "Torque saturation in bipedal robotic walking through control lyapunov function-based quadratic programs," *IEEE Access*, vol. 3, pp. 323–332, 2015.

[22] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 1, pp. 43–53, 1987.

[23] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot modeling and control*. John Wiley & Sons, 2020.

[24] H. K. Khalil, "Nonlinear systems," *Upper Saddle River*, 2002.

[25] M. A. Henson and D. E. Seborg, "Feedback linearizing control," in *Nonlinear process control*. Prentice-Hall, 1997, vol. 4, pp. 149–231.

[26] P. A. Parrilo, "Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization," Ph.D. dissertation, California Institute of Technology, 2000.

[27] J. Lee, M. X. Grey, S. Ha, T. Kunz, S. Jain, Y. Ye, S. S. Srinivasa, M. Stilman, and C. K. Liu, "Dart: Dynamic animation and robotics toolkit." *J. Open Source Software*, vol. 3, no. 22, p. 500, 2018.

[28] "RES-CLF-QP hierarchical framework for whole-body control of WIP humanoid robots," 2020. [Online]. Available: https://youtu.be/MK95Pv{_}f21A

[29] Y. Tassa, N. Mansard, and E. Todorov, "Control-limited differential dynamic programming," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 1168–1175.

[30] A. D. Ames, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs with application to adaptive cruise control," in *53rd IEEE Conference on Decision and Control*. IEEE, 2014, pp. 6271–6278.