2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

OTTO: Dynamics and Control of Wheeled Bipedal Jumping Robot

Paweekorn Buasakorn, Supachai Vongbunyong and Kitti Thamrongaphichartkul*

***Abstract*— This paper presents *OTTO*, a study of jumping and terrain traversal for a 6-DOF wheeled bipedal robot that addresses the limitations of purely wheeled or legged locomotion when navigating complex and challenging environments. We develop a robot model based on a 3D wheeled inverted pendulum (WIP) system equipped with torso degrees of freedom. A unified framework integrates LQR-based controllers with jumping strategies to enable effective terrain traversal. The system features a novel flight posture controller that exploits wheels as reaction wheels to actively control aerial posture without predetermined activation timing. Additionally, we implement a comprehensive jumping cycle using trajectory optimization that encompasses a terminal trajectory planner for takeoff and landing phases, as well as a flight phase planner. We execute various comprehensive experiments to assess the capabilities of these behaviors utilizing the Gazebo ROS2 simulation environment. The complete system architecture is evaluated through simulation trials of a prototype wheeled-bipedal robot, illustrating the viability and robustness of the proposed approach for terrain locomotion and jumping.**

I. Introduction

Wheeled bipedal robots (WBRs) have gained significant popularity due to their ability to rapidly maneuver by combining the advantages of both locomotion modes. Wheels enable fast and high efficiency traversal on planar surfaces, while legs provide the capability to traverse challenging topographies through differential leg extension. When each side operates at different extension levels, the robot can maintain its zero moment point while maneuvering on uneven terrain, such as when each side encounters different floor elevations. Another major advantage of combining legs with wheels is the implementation of ballistic locomotion strategies. These aerial maneuvers enable the robot to overcome discrete instantaneous obstacles that exceed the wheel radius, such as ascending stairs or traversing elevated barriers.

A. Related Work

This research was inspired by Boston Dynamics' Handle robot [1] a WBR deployed in real-world applications, though detailed technical documentation remains unavailable. ETH's Ascento [2], [3] introduces a spring-loaded mechanism with four actuators but sacrifices head pitch control during locomotion. Five-bar mechanisms [4], [5] effectively address this limitation but require increased spatial footprint. Recent highly compact designs, including SKATER [6] and Diablo [7], employ parallelogram mechanisms [8] to achieve both pitch control and spatial efficiency, leading to our selection of this design for *OTTO*.

All authors are members of CARVER Robotic System, Institute of Field Robotics, Thailand, Thailand. Contact: paweekorn.pb@gmail.com, supachai.von@kmutt.ac.th, kitti.tham@kmutt.ac.th

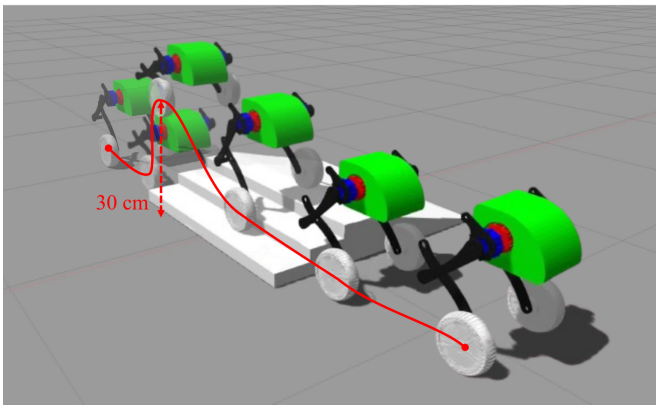

Fig. 1. *OTTO* Robot simulation showing jumping over stairs and landing on different floor levels using a simplified serial-link leg mechanism. The robot reaches 30 cm at its apex state. The terrain features individual step heights of 5 cm. Accompanying video for various terrain experiments is available: https://www.youtube.com/watch?v=ynG1KE$_7U2w$

While reinforcement learning methods [4], [9], [10] face challenges related to unpredictability, generalization of neural networks, and intensive training requirements, model-based approaches remain essential. SKATER employs MPC-assisted Whole-Body Control (WBC), whereas Diablo demonstrates faster response using LQR-based direct torque control. LQR controllers offer robustness, interpretability, and ease of tuning. We extend previous 2D modeling approaches [7], [11]-[13] by implementing a 3D WIP model equipped with hip motors, which facilitates a more straightforward and efficient LQR controller design for the yaw task.

For jumping strategies, existing work [14]-[16] focuses on leg retraction and takeoff planning. However, this approach cannot deal with disturbances during flight. Without prior planning in the take-off phase, the conservation of angular momentum could cause the robot to land outside the controller's operational range. We propose repurposing the locomotion wheels as reaction wheels during aerial phases to actively control pendulum angles and prevent undesirable landing configurations. This approach eliminates the need for planning and can handle disturbances during flight.

B. Contribution

Our main contributions can be summarized as follows:

- We develop an LQR controller with equilibrium point analysis that obtains knowledge from a 3D WIP model equipped with torso dynamics to ensure the robot converges to equilibrium under force disturbances. This is integrated with a motion controller to enable *OTTO* to navigate through uneven terrain. (Section II-III)

- We present a Jumping Cycle Planner using trajectory optimization based on quadratic programming to plan the leg extension/contraction cycle, including take-off, flight, and touch-down phases. (Section IV)
- We purpose a novel posture controller for *OTTO* during the aerial phase, utilizing the existing wheels as reaction wheels. This approach eliminates the need for pre-computed posture trajectories during take-off, preventing unwanted angular momentum and disturbances that could cause *OTTO* to land outside the controller's operational range. (Section V)

We demonstrate the performance through simulation studies in Section VI, and present conclusions and future research directions in Section VII.

II. Modeling

In Section III, we employ an LQR controller requiring mathematical system dynamics. We simplify by neglecting the knee joint, yielding a 3D Wheel Inverted Double Pendulum (3D-WIDP) model using Lagrangian formulation. The dynamics are derived from coordinate frames shown in Figure 2: ground frame $\mathcal{G}$, control frame $\mathcal{W}$ (midpoint between wheels), pendulum frame $\mathcal{P}$, and torso frame $\mathcal{T}$.

With the coordinate frames established, the equations can be derived using the Lagrangian method. For the 3D-WIDP, the kinetic energy of the wheels can be written as:

$$T_W = \frac{1}{2}\dot{\mathbf{r}}_w^T m_w \dot{\mathbf{r}}_w + \frac{1}{2}\dot{\rho}_w^T \mathbf{I}_w \dot{\rho}_w \tag{1}$$

where $\dot{\mathbf{r}}_w \in \mathbb{R}^3$ is the wheel linear velocity vector, $\mathbf{I}_w \in \mathbb{R}^{3\times3}$ is the wheel inertia tensor, and $\dot{\rho}_w \in \mathbb{R}^3$ is the wheel angular velocity vector defined as:

$$\dot{\mathbf{r}}_w = [\dot{x}_w, \dot{y}_w, 0]^T, \quad \dot{\rho}_w = [0, \dot{x}_w/r, \dot{\varphi}]^T \tag{2}$$

The kinetic energy of the pendulum frame and torso frame can be expressed as:

$$T_{PT} = \frac{1}{2}\dot{\mathbf{r}}_p^T m_p \dot{\mathbf{r}}_p + \frac{1}{2}\dot{\rho}_p^T \mathbf{I_p} \dot{\rho}_p + \frac{1}{2}\dot{\mathbf{r}}_t^T m_t \dot{\mathbf{r}}_t + \frac{1}{2}\dot{\rho}_t^T \mathbf{I_t} \dot{\rho}_t \tag{3}$$

where m_p, m_t are pendulum and torso masses, $\mathbf{I}_p, \mathbf{I}_t \in \mathbb{R}^{3\times3}$ are inertia tensors around CoM_1, CoM_2, $\mathbf{r}_p, \mathbf{r}_t \in \mathbb{R}^3$ are position vectors, and $\rho_p, \rho_t \in \mathbb{R}^3$ are angular velocities.

By considering longitudinal wheel-ground contact without lateral slip, we derive velocity expressions for system components. Under this assumption, velocity vectors $\dot{\mathbf{r}}_p$ and $\dot{\mathbf{r}}_t$ for pendulum and torso frames can be expressed as:

$$\begin{aligned}
\dot{\mathbf{r}}_t &= \begin{bmatrix} \dot{x}_c + L\dot{\theta}c_\theta c_\varphi - L\dot{\varphi}s_\theta s_\varphi + l\dot{\beta}c_\varphi c_\beta - l\dot{\varphi}s_\beta s_\varphi \\ \dot{y}_c + L\dot{\theta}c_\theta s_\varphi + L\dot{\varphi}s_\theta c_\varphi + l\dot{\varphi}s_\beta c_\varphi + l\dot{\beta}s_\varphi c_\beta \\ -L\dot{\theta}s_\theta - l\dot{\beta}s_\beta \end{bmatrix} \\
\dot{\mathbf{r}}_p &= \begin{bmatrix} \dot{x}_c + \frac{1}{2}L(\dot{\theta}c_\theta c_\varphi - \dot{\varphi}s_\theta s_\varphi) \\ \dot{y}_c + \frac{1}{2}L(\dot{\theta}c_\theta s_\varphi - \dot{\varphi}s_\theta c_\varphi) \\ -\frac{1}{2}L\dot{\theta}s_\theta \end{bmatrix}
\end{aligned} \tag{4}$$

where L is the leg length from the wheel frame to the torso frame, l is the torso length from the torso frame to CoM2, r is wheel radius, θ is pendulum tilt angle, β is torso rotation angle, and φ is the yaw angle.

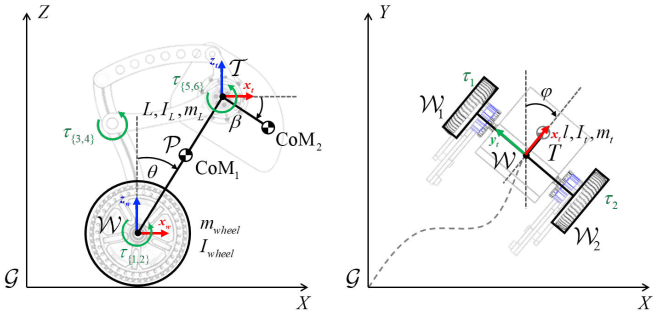

Fig. 2. Coordinate systems with simplified dynamics neglecting the knee joint of *OTTO*. Left: **side view** with pendulum angle and torso angle . Right: **top view** with yaw angle and wheel configuration.

Similarly, the angular velocity of pendulum frame $\dot{\rho}_p$ and torso frame $\dot{\rho}_t$ can be expressed as

$$\dot{\rho}_p = \begin{bmatrix} -\dot{\varphi}\sin(\theta) \\ \dot{\theta} \\ \dot{\varphi}\cos(\theta) \end{bmatrix}, \dot{\rho}_t = \begin{bmatrix} -\dot{\varphi}\sin(\beta) \\ \dot{\beta} \\ \dot{\varphi}\cos(\beta) \end{bmatrix} \tag{5}$$

The total kinetic energy combines contributions from both pendulums and wheels:

$$T_t = T_{PT} + 2T_W \tag{6}$$

The gravitational potential energy accounts for both pendulum and torso positions:

$$V = m_p g L\cos(\theta) + m_t g(L\cos(\theta) - l\sin(\beta)) \tag{7}$$

Using the Lagrangian method with defined kinetic and potential energies, the equations of motion are:

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}_i}\right) - \frac{\partial \mathcal{L}}{\partial \mathbf{q}_i} = \mathbf{Q}_i \tag{8}$$

where $\mathcal{L} = T_t - V$ defines the complete system dynamics with generalized coordinates $\mathbf{q}_i = \{x_w, \theta, \beta, \varphi\}$ and corresponding generalized external forces $\mathbf{Q}_i = \{F_x, \tau_\theta, \tau_\beta, \tau_\varphi\}$. To transform left and right wheel actuator torques τ_l and τ_r to these generalized forces, the mapping is:

$$F_x = \frac{\tau_r + \tau_l}{r}, \quad \tau_\theta = -F_x r, \quad \tau_\varphi = \frac{d}{2r}(\tau_r - \tau_l) \tag{9}$$

The viscous friction forces opposing motion in both the wheel and pendulum are modeled as:

$$F_x = \frac{2\mu\dot{x}_w}{r^2} - \frac{2\mu\dot{\theta}}{r}, \quad \tau_\theta = -\frac{2\mu\dot{x}_w}{r} + 2\mu\dot{\theta} \tag{10}$$

where d is wheelbase length, μ is viscous friction coefficient, and ω_l, ω_r are left and right wheel angular velocities.

3D-WIDP model derived in (8) functions as manipulator system, with its motion governed by dynamic equations. The system's behavior is characterized by:

$$\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{D}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \mathbf{B}\mathbf{u} \tag{11}$$

where $\mathbf{q}$ represents the generalized coordinates, $\mathbf{H}$ is the mass inertia matrix, $\mathbf{C}$ accounts for Coriolis and centrifugal motion effects, $\mathbf{G}$ represents gravitational forces, $\mathbf{B}$ is the input matrix that transforms wheel control inputs into generalized external forces using relations in (9), and $\mathbf{D}$ represents damping friction forces from (10).

III. Stance Posture Controller

A. LQR-Based Controller

The LQR-based controller is designed to manage three primary tasks for the *OTTO* robot: stabilization, yaw control, and torso rotation. Since LQR is a linear control technique, we must first linearize the nonlinear dynamics from equation (11) around a fixed equilibrium point. We define the system state vector and control inputs as follows:

$$\hat{\mathbf{x}} = \begin{bmatrix} x_w & \theta & \beta & \varphi & \dot{x}_w & \dot{\theta} & \dot{\beta} & \dot{\varphi} \end{bmatrix}^T \tag{12}$$

$$\mathbf{u} = \begin{bmatrix} \tau_l & \tau_r & \tau_\beta \end{bmatrix}^T \tag{13}$$

The nonlinear system is linearized around the desired equilibrium point or control law (locally) ξ_0 defined as

$$\xi_0 = \begin{bmatrix} e & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T \tag{14}$$

where e represent small deviations from the nominal equilibrium state. Using standard linearization techniques for dynamic systems [18], the linearized system dynamics can be expressed in the canonical state-space form:

$$\dot{\hat{\mathbf{x}}} = \mathbf{A}\hat{\mathbf{x}} + \mathbf{B}\mathbf{u} \tag{15}$$

The system matrices $\mathbf{A}$ and $\mathbf{B}$ are derived by computing the Jacobians of the nonlinear dynamics with respect to the state and control variables, respectively, given by:

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{H}^{-1}\frac{\partial \mathbf{G}}{\partial \mathbf{q}} & -\mathbf{H}^{-1}\left(\mathbf{C}+\mathbf{D}\right) \end{bmatrix} \mathbf{B} = \begin{bmatrix} \mathbf{0} \\ \mathbf{H}^{-1}\mathbf{B} \end{bmatrix} \tag{16}$$

To make the state $\hat{\mathbf{x}}$ converge to ξ_0, the error vector is defined as $\hat{\mathbf{e}} = \hat{\mathbf{x}} - \xi_0$. Subsequently, full state feedback control $\mathbf{u} = -\mathbf{K}_{3\times 8}\hat{\mathbf{e}}$ is then applied, where $\mathbf{K}_{3\times 8}$ is the gain matrix. The equilibrium can be written as

$$(\mathbf{A} - \mathbf{B}\mathbf{K})^{-1}\mathbf{B}k_\sigma e = \begin{bmatrix} a & 0 & b & c & 0 & 0 & 0 & 0 \end{bmatrix}^T \tag{17}$$

where $k_\sigma = k_1 + k_9 + k_{17}$, with k_1, k_9, and k_{17} being elements of the gain matrix $\mathbf{K}_{3\times 8}$, and a, b, c are non-zero constants that represent the steady-state values for the forward position, torso rotation, and yaw angle, respectively. Since these constants are non-zero, the three states exhibit persistent offsets from equilibrium ξ_0.

To eliminate this steady-state error [19], an auxiliary variable $\mathbf{x}_s$ is introduced to provide integral action by accumulating the system errors, where $\dot{\mathbf{x}}_s = \begin{bmatrix} 0 - x_w & 0 - \beta & 0 - \varphi \end{bmatrix}$. The extended state space can be written as:

$$\begin{bmatrix} \dot{\hat{\mathbf{x}}} \\ \dot{\mathbf{x}}_s \end{bmatrix} = \underbrace{\left[\begin{array}{c|c} \mathbf{A} & \mathbf{0}_{8\times 3} \\ \hline -\mathbf{E} & \mathbf{0}_{3\times 3} \end{array}\right]}_{\mathbf{A_e}} \underbrace{\begin{bmatrix} \hat{\mathbf{x}} \\ \mathbf{x}_s \end{bmatrix}}_{\mathbf{x}_e} + \underbrace{\begin{bmatrix} \mathbf{B} \\ \mathbf{0}_{3\times 3} \end{bmatrix}}_{\mathbf{B_e}} \mathbf{u} \tag{18}$$

Let $\mathbf{E} \in \mathbb{R}^{3\times 8}$ be a selection matrix formed by vertically stacking the corresponding standard basis vectors: $\mathbf{E} = \begin{bmatrix} \mathbf{e}_1^T & \mathbf{e}_3^T & \mathbf{e}_4^T \end{bmatrix}^T$ where each $\mathbf{e}_i^T \in \mathbb{R}^{1\times 8}$ is a row vector with a one in the i-th position and zeros elsewhere. The equilibrium point of extended state space will be:

$$(\mathbf{A}_e - \mathbf{B}_e\mathbf{K}_{3\times 11})^{-1}\mathbf{B}_e k_\sigma e = \begin{bmatrix} \mathbf{0}_{1\times 8} & a & b & c \end{bmatrix}^T \tag{19}$$

Equation (19) shows that with integral action, all states $\hat{\mathbf{x}}$ can now converge to the desired equilibrium point at steady state, eliminating the offsets even under disturbances.

The optimal feedback gain matrix $\mathbf{K}_{3\times 11}$ is computed using discrete-time linear quadratic regulator (LQR) design methodology. The LQR approach iteratively solves the corresponding discrete-time algebraic Riccati equation [17]:

$$\mathbf{A}_e^T\mathbf{S}\mathbf{A}_e - \mathbf{S} - \mathbf{A}_e^T\mathbf{S}\mathbf{B}_e(\mathbf{B}_e^T\mathbf{S}\mathbf{B}_e + \mathbf{R})^{-1}\mathbf{B}_e^T\mathbf{S}\mathbf{A}_e + \mathbf{Q} = \mathbf{0} \tag{20}$$

where $\mathbf{S}$ is the infinite horizon solution of the Riccati equation, and $\mathbf{Q}$ and $\mathbf{R}$ are the symmetric positive definite state and control weighting matrices, respectively. The optimal feedback gain is then obtained as:

$$\mathbf{K}_{3\times 11} = (\mathbf{B}_e^T\mathbf{S}\mathbf{B}_e + \mathbf{R})^{-1}\mathbf{B}_e^T\mathbf{S}\mathbf{A}_e \tag{21}$$

B. Gait Controller

The gait controller addresses model mismatch in the 3D-WIDP balance model. Since it employs a 2D second pendulum representation, it fails to capture 3D torso dynamics. Differential wheel torques for steering create unmodeled yaw moments causing leg separation $\Delta x = x_w^l - x_w^r$, where x_w^l and x_w^r are the x-positions of left and right frames relative to control frame $\mathcal{W}$ from forward kinematics. To mitigate this, a PD controller regulates wheel positions:

$$\tau_{5,6} = \tau_\beta \pm \left(k_p(\Delta x^* - \Delta x) + k_d(\Delta \dot{x}^* - \Delta \dot{x}^*)\right) \tag{22}$$

where Δx^* is the desired leg separation, and ± corresponds to τ_5 and τ_6 respectively. This strategy maintains leg spacing by counteracting yaw disturbances, ensuring stability during steering while preserving gait patterns.

C. Height Controller

The goal of the height controller is to adjust L by modulating knee torque for leg length control. The actuation torque from the height controller can be written as:

$$\tau_H = \mathbf{J}^T(m_t g) + \Big(k_p(L^* - L) + k_d(\dot{L}^* - \dot{L})\Big) \tag{23}$$

where $\mathbf{J}$ is the Jacobian matrix for gravity compensation due to torso mass, and L^*, $\dot{L}^*$ are the desired leg length and its rate, commanded by the user during stance. It is equal to $L(t)$ during the jump process (Section VI).

D. Lean Angle Controller

To handle uneven terrain where each wheel encounters different heights, a PD controller is used to regulate the lean angle and prevent *OTTO* from tipping outside the support polygon. The control input is added to the knee torque from the height controller τ_H, resulting in:

$$\tau_{3,4} = \tau_H \pm \Big(k_p(\Psi^* - \Psi) + k_d(\dot{\Psi}^* - \dot{\Psi})\Big) \tag{24}$$

where Ψ^* is the desired lean angle, and the $\pm$ corresponds to τ_3 and τ_4, respectively. The appropriate lean angle to counteract centrifugal forces during high-speed turns can be determined using moment equilibrium analysis. This yields the desired lean angle as:

$$\Psi^* = \arctan\left(\frac{\dot{x}_w\dot{\varphi}}{g}\right) \tag{25}$$

This ensures the robot maintains balance during high-speed turns by leaning into the curve to counteract forces.

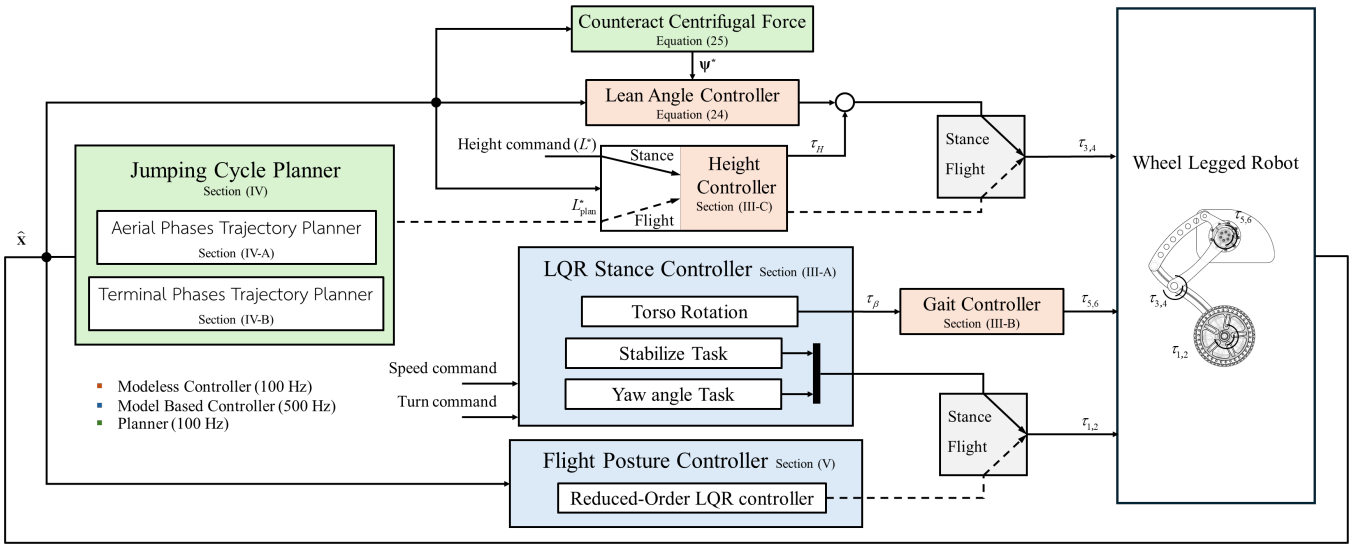

Fig. 3. The overall system architecture of *OTTO* shows how each component described in this paper is connected. The red block computes the input torque to the leg using a model-less controller running at 100 Hz. The blue block is an LQR-based controller that pre-computes the torque applied to the hip and computes the torque applied to the wheel, running at 500 Hz. The green block is a planner that computes the leg length, running at 100 Hz.

IV. Jumping Cycle Planner

As shown in Fig. 3, the jumping cycle planner manages leg extension during the jump. It consists of the Terminal Phases Trajectory Planner and the Aerial Phases Trajectory Planner. The terminal planner generates leg trajectories for take-off and landing, triggered by a jump command and ending after take-off. The aerial planner computes leg trajectories to reach the desired length at the apex and before landing. Once the aerial plan is complete, the landing trajectory from the terminal planner is executed.

A. Terminal Phases Trajectory Planner

Jumping transforms vertical kinetic energy into potential energy, with jump height determined by the vertical take-off velocity, which depends on the rate of leg extension. To model this, we represent the leg-length trajectory as an N-degree polynomial with coefficients α, allowing leg length and its derivatives to be expressed linearly:

$$y(t) = \mathbf{\Phi}(t)\,\alpha \tag{26}$$

where $y(t) = (L(t), \dot{L}(t), \ddot{L}(t))$ contains the planned trajectory of leg length $L(t)$, velocity $\dot{L}(t)$, and acceleration $\ddot{L}(t)$ at time t. The matrix $\mathbf{\Phi}(t)$ is the basis matrix defined as:

$$\mathbf{\Phi}(t) = \begin{bmatrix} 1 & t & t^2 & \cdots & T^N \\ 0 & 1 & 2t & \cdots & N\,T^{N-1} \\ 0 & 0 & 2 & \cdots & N(N-1)\,T^{N-2} \end{bmatrix}. \tag{27}$$

The polynomial trajectory serves distinct roles in take-off and landing. During take-off, the leg starts from rest and accelerates to achieve the desired take-off velocity, generating upward momentum. In contrast, the landing trajectory begins with negative leg velocity due to ground impact and decelerates smoothly to zero, reducing jerk. In both phases, the polynomial formulation ensures smooth transitions while satisfying boundary conditions over trajectory duration T

Algorithm 1 Minimum Leg Length Constrained Planning

Input: $(L_0, \dot{L}_0, L_f, \dot{L}_f, \ddot{L}_f)$, L_{min}
Output: α, T_{opt}
Initialize $T = T_{\text{init}}$, range $[T_{\text{min}}, T_{\text{max}}]$
repeat
  Solve α with current T
  Find critical points from $\dot{L}(t)$ in $[0, T]$
  $L_{\text{min, actual}} \leftarrow \min L(t)$ at critical points
  if $|L_{\text{min, actual}} - L_{\text{min}}| < \epsilon$ **then**
    $T_{\text{opt}} = T$; **break**
  else if $L_{\text{min, actual}} < L_{\text{min}}$ **then**
    $T_{\text{max}} = T$; $T = (T_{\text{min}} + T)/2$
  else
    $T_{\text{min}} = T$; $T = (T + T_{\text{max}})/2$
  end if
until max iterations
return α, T_{opt}

To ensure the leg trajectory respects mechanical constraints, we propose an optimization algorithm that enforces a minimum leg length L_{min}. Given the initial condition L_0 and final conditions L_f for leg length, velocity, and acceleration, Algorithm 1 iteratively searches for an optimal trajectory duration T such that the planned trajectory satisfies the leg length constraint. At each iteration, the polynomial coefficients α are computed for the current estimate of T. The minimum leg length along the trajectory is evaluated by inspecting critical points from $\dot{L}(t)$, and the algorithm adjusts T using a bisection strategy until the minimum leg length matches the desired constraint. This approach avoids manually prescribing the duration T, which can be inconvenient or unintuitive, and instead allows L_{min} to implicitly define an appropriate T. The procedure terminates upon convergence or reaching a maximum number of iterations, yielding polynomial coefficients and optimal trajectory duration.

B. Aerial Phases Trajectory Planner

To efficiently overcome obstacles, the robot must squat during flight to create an optimal ballistic trajectory for the center of mass (CoM). During flight, the robot behaves as a projectile under gravity with parabolic motion described by the CoM height $z_{\text{CoM}}(t)$ as a function of time t.

$$z_{CoM}(t) = z_0 + \dot{z}_0 t - \frac{1}{2} g t^2 \tag{28}$$

where z_0 is the initial CoM height at takeoff, $\dot{z}_0$ is the initial vertical velocity, and g is the gravitational acceleration.

To model the CoM during flight, we consider the robot's mass distribution where m_t and m_w are separated by distances d_t and d_w from the CoM respectively, and determine that l is very low, near zero, and not considered part of the leg. The geometric constraint requires that these distances sum to the total leg length L, while moment equilibrium gives the CoM position as:

$$x_{CoM} = \frac{m_t L \sin\theta}{m_t + m_w}, \quad z_{CoM} = \frac{m_t L \cos\theta}{m_t + m_w}. \tag{29}$$

The flight time T_f is determined by:

$$T_f = \frac{1}{g}\left(\dot{z}_0 + \sqrt{\dot{z}_0^2 - 2g\left(\frac{m_t L_{TD}\cos(\theta_{TD})}{m_t + m_w} + z_f - z_0\right)}\right) \tag{30}$$

To maximize jump efficiency, we minimize leg length at the apex while ensuring obstacle clearance and respecting physical constraints. The apex time $T_{\text{top}} = \dot{z}_0/g$ occurs when the CoM reaches maximum height. With boundary conditions at touchdown (L_{TD}, θ_{TD}) and terrain height z_f, the optimization can be formulated as:

$$\begin{aligned} \min_{L(\cdot)} \quad & \|L(T_{\text{top}}) - L_{\text{des}}\|_2^2 \\ \text{s.t.} \quad & L(0) = z_0, \quad \dot{L}(0) = \dot{z}_0, \\ & L(T_f) = L_{\text{TD}}, \quad \dot{L}(T_f) = \dot{L}_{\text{TD}}, \end{aligned} \tag{31}$$

To solve the optimization problem (31), we parameterize the leg length profile $L(t)$ using a polynomial representation:

$$L(t) = \sum_{i=0}^{N} a_i t^i \tag{32}$$

The leg length trajectory is parameterized as a polynomial, and the coefficients a_i are computed by solving a quadratic programming (QP) problem. This formulation enforces boundary conditions at takeoff and touchdown while minimizing deviation from a desired apex length. The QP-based approach ensures a smooth and physically feasible trajectory $L(t)$, which is then tracked in flight by substituting $L^* = L(t)$ into the height controller (23).

V. Flight Posture Controller (FPC)

Previous studies [12], [20], [21] identify a critical issue in robotic jumping: neglecting angular momentum conservation leads to uncontrolled increases in the robot's pendulum angular rate during leg retraction in the flight phase, jeopardizing landing stability. This problem stems from the inverse relationship between the robot's moment of inertia and its angular velocity: as the leg retracts, its effective length $L(t)$ as planned in Section IV-B decreases, reducing the moment of inertia $\mathcal{I}(t)$. This relationship is governed by $\dot{\theta}(t) = \mathcal{I}_0\dot{\theta}_0/\mathcal{I}(t)$, where

$$\mathcal{I}_0 = \frac{m_t\, m_w}{m_t + m_w} L_0^2, \quad \mathcal{I}(t) = \frac{m_t\, m_w}{m_t + m_w} L(t)^2 \tag{33}$$

Here, $\dot{\theta}(t)$ is the pendulum angular rate during flight, and $\mathcal{I}_0$, $\dot{\theta}_0$ are initial values. As $L(t)$ decreases, $\dot{\theta}(t)$ increases quadratically, causing rapid angular acceleration. A large pendulum angle at landing may exceed control limits.

To regulate posture during flight, we propose using wheel torques as internal reaction forces. Acting as reaction wheels, they generate corrective torques based on action-reaction principles. The resulting dynamics of the free-fall pendulum with reaction wheels are:

$$\ddot{\theta} = -\frac{2}{\mathcal{I}}\left(\tau_l + \tau_r - \mu(\dot{\phi} - \dot{\theta})\right) \tag{34}$$

$$\ddot{\phi} = \frac{2}{I_w^y}\left(\tau_l + \tau_r - \mu(\dot{\phi} - \dot{\theta})\right) \tag{35}$$

where I_w^y is the wheel's moment of inertia about its rotation axis. However, this system is partially controllable [22]. The wheel state is uncontrollable and does not need to be controlled, so we use Kalman canonical decomposition [23] to decompose the system into controllable and uncontrollable subsystems. Expressing the system in state-space form yields (15), with the controllable state vector $\hat{\mathbf{x}} = [\theta,\ \dot{\theta}]^\top$ and controllable subsystem matrices $\mathbf{A}_c$ and $\mathbf{B}_c$ defined as:

$$\mathbf{A}_c = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{2\mu}{\mathcal{I}} \end{bmatrix}, \quad \mathbf{B}_c = \begin{bmatrix} 0 \\ \frac{2}{\mathcal{I}} \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} 0 \\ \tau_l + \tau_r \end{bmatrix} \tag{36}$$

The system in (36) represents the controllable subsystem, particularly the pendulum states. A reduced-order LQR is applied, allowing full-state feedback on the controllable states. Substituting (36) into (20) and (21) yields the wheel torque controller for flight.

VI. Results and Dicussion

This paper focuses on the proof-of-concept of a modeling and control framework presented in Fig.3. The system was validated using *Gazebo Classic* [24] in ROS2 Humble with *ODE* as backend physics engine. The dynamic properties are estimated using *SOLIDWORKS*, and the actuation torque limits for hip and knee joints are set to match the concept prototype specifications using *Xiaomi Cybergear Micromotor* and *ODrive BotWheels* for the wheels. For the leg mechanism, we simplified the parallelogram mechanism to a serial link configuration since closed kinematic structures are not supported in URDF. To obtain the *OTTO* state information, we utilize an IMU equipped on the torso and encoders for the leg and wheel joints. To solve the QP and optimization problems (Sections IV-A, IV-B), we use CVXPY [25], [26]. Complete implementation details, controller parameters, and experiment configurations are available at here. The experiments we conducted to validate this paper are as follows:

To validate the modeling presented in Section II and the stance posture controller described in Section III, we conducted experiments beyond flat terrain traversal. The robot was tested across various terrain conditions as shown in Figure 4, with each terrain possessing distinct properties that challenge different aspects of the locomotion system.

Ramp Terrain: The ramp features a slope of 15° with a maximum height of $15\,\text{cm}$. This configuration allows observation of the robot's response when traversing inclined surfaces and evaluating its ability to maintain stability during continuous slope navigation.

Stair Terrain: The experimental setup consists of a continuous ramp inclined at 15° reaching a peak elevation of $15\,\text{cm}$, followed immediately by a three-step staircase where each step has a height of $5\,\text{cm}$. This terrain enables observation of the robot's response to abrupt height changes when vertical drops, testing its capability to handle discrete elevation transitions.

Rough Terrain: This terrain incorporates a 15° uphill slope, a 15° downhill slope, and an uneven surface modeled as $2.5\sin(fx)$, where f represents the spatial frequency. This configuration results in a maximum elevation variation of $5\,\text{cm}$. The purpose is to observe the robot's response when navigating combined sloped and irregular surfaces, testing its adaptability to unpredictable terrain variations.

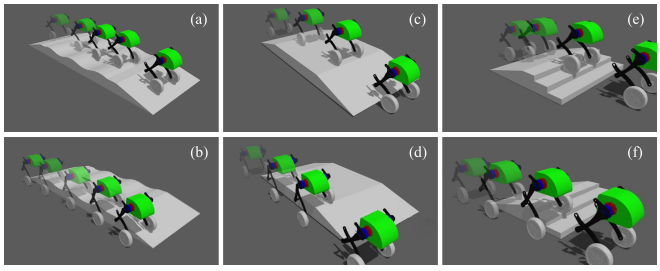

Fig. 4. Demonstration of *OTTO* ground locomotion at 1 m/s under different terrain conditions. The first row shows *OTTO* locomotion with both wheels in contact with the terrain. The second row demonstrates *OTTO* locomotion with one leg on the ground and other leg navigating over terrain obstacles.

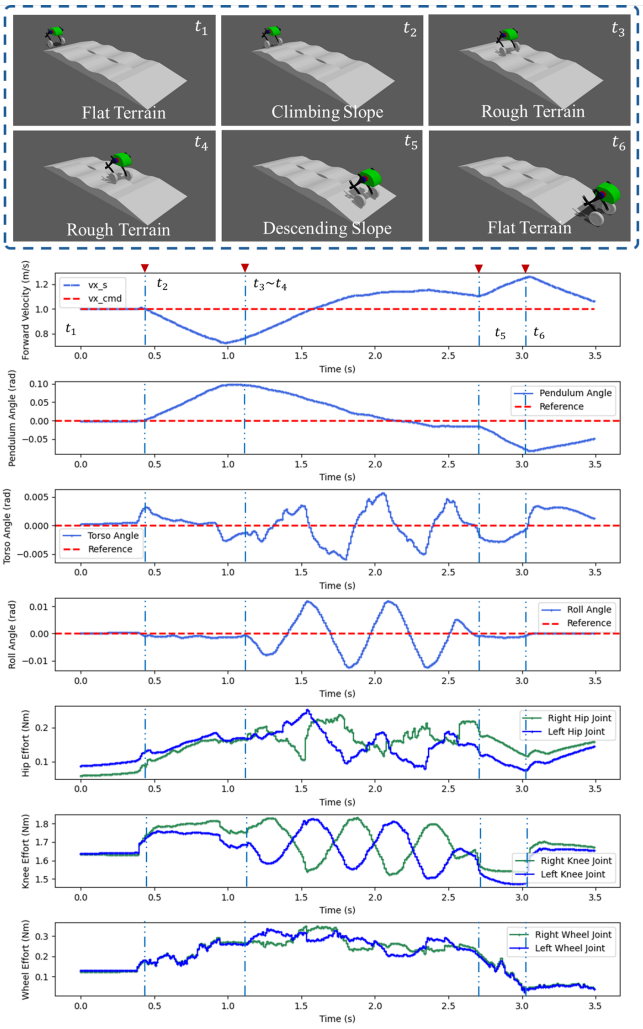

Fig. 5. Experimental validation of the control framework for *OTTO* navigating varied terrain at 1 m/s. Top: Sequential phases showing (t_1) flat terrain, (t_2) climbing, (t_3–t_4) rough terrain, (t_5) descending, and (t_6) return to flat. Bottom: Time-series data demonstrating successful terrain adaptation and state convergence.

A. Terrestrial Locomotion Experiments on Uneven Terrain

To describe the behavior of the control framework, we first picked up rough terrain when two legs were in contact with the terrain. In experiment shown in Fig. 5, we commanded the *OTTO* to move forward at a speed of $1,\text{m/s}$ through the rough terrain. The results showed that at t_1, the robot successfully tracked the commanded forward velocity of $1,\text{m/s}$ on flat terrain. At t_2, while climbing the 15° incline, the speed decreased due to gravitational resistance. The pendulum angle increased to approximately 6°, and the torso pitch angle peaked at around 0.5° before converging to 0 completely. To counteract the added load with a converging trend according to the control law, all control inputs increased, including motor torques and balance control efforts. Upon reaching the rough surface on top of the ramp at t_3–t_4, the knee joint on the lower side exerted more effort to compensate for the tilt and maintain body stability. The performance of the response can be described by the roll angle: the robot experienced a roll angle within $0.01,\text{rad}$ (approximately 0.4°). The robot's forward velocity and pendulum angle did not fluctuate and converged to the desired control law compensation as on flat terrain, even while navigating rough terrain as it traversed the uneven surface. The left and right wheel torques were unequal during this period, adjusting dynamically to regulate yaw and preserve directional control, and the torso angle remained quite low with a peak of only 0.28°. At t_5, during the descent, the robot's speed increased as expected, and the control effort decreased, returning to levels close to those seen before the climb. The robot responded by attempting to decelerate and regain balance. Finally, at t_6, once back on flat terrain, all states reconverged to the desired equilibrium.

In the single-leg experiment, we tested the robot on stair terrain (Figure 6) to validate that the controller can handle discrete terrain changes. The results showed that most states

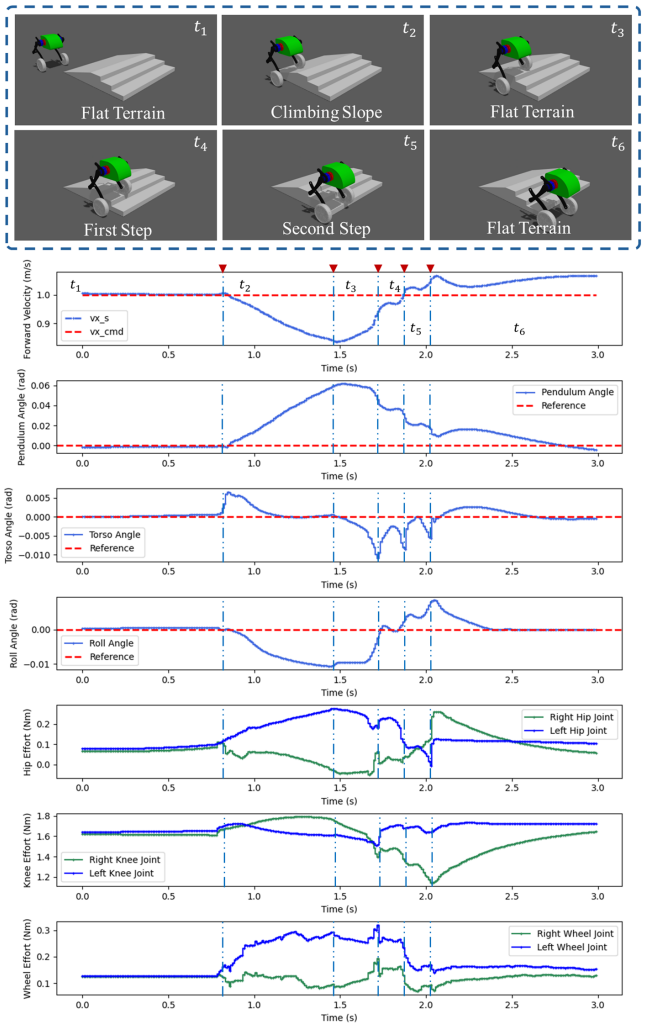

Fig. 6. Single-leg stair descent experiment showing control framework behavior across discrete terrain transitions. The phases include: (t_1) flat approach, (t_2) slope climbing, (t_3) upper flat terrain, (t_4–t_5) stair descent, and (t_6) flat return. Key observations: knee effort increases during uphill motion and decreases during descent; wheel effort shows terrain-dependent actuation; roll angle exhibits greater deviation than two-leg locomotion.

behaved similarly to the slope climbing with two legs in 5, except for the roll angle, which experienced larger deviations. When descending stairs, the convergence trends of all states were similar to those observed when moving on flat terrain, but exhibited some spikes during discrete transitions when stepping down each stair.

B. Jumping Strategies Experiments

The experiment in Fig. 7 observes the aerial phase planner, controller, and take-off planner performance. The take-off planner successfully enables the robot to jump at desired velocity while maintaining $1, \text{m/s}$ forward speed. During flight, the aerial phase planner retracts the leg to maximum extent at apex. At take-off, the pendulum angle is 3° with rate $11.5^\circ/\text{s}$. Due to conservation of angular momentum, the rate increases significantly at apex, which would normally exceed control boundaries.

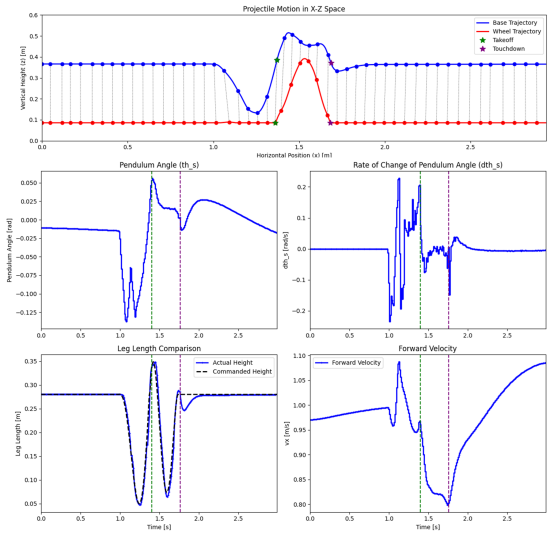

Fig. 7. Projectile motion experiment of OTTO jumping. Top: X-Z trajectory. Bottom: pendulum angle, rate, leg length, and velocity time series.

However, with our FPC, the pendulum angle converges to 0° during landing with rate $-8^\circ/\text{s}$ due to touchdown impulse. The touchdown speed is $0.8, \text{m/s}$, and the system converges back to desired forward velocity.

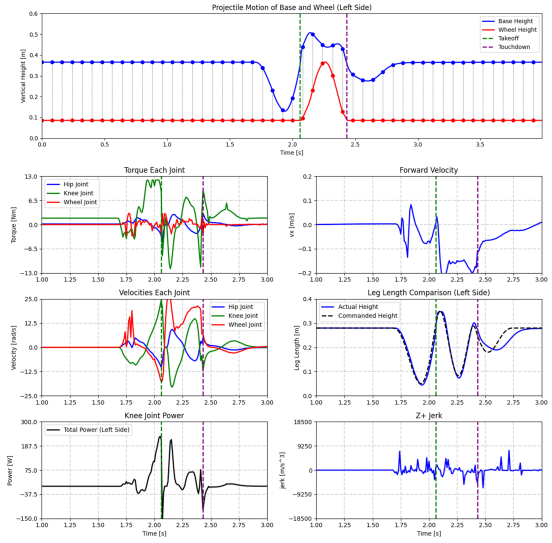

Fig. 8. Integrated terminal phases trajectory planner during vertical jumping showing trajectories, joint dynamics, velocity, leg length, power, and jerk.

We evaluated the performance of the integrated terminal phases trajectory planner that incorporates the takeoff planner, flight phase planner, and touchdown planner working in sequence. We conducted experiments shown in Fig. 8 with vertical jumping without forward velocity command. The key distinguishing characteristic of the integrated system is the dramatic reduction in touchdown jerk. The results demonstrate that the jerk magnitude remains at only -6000 units during touchdown, representing a 3.5-fold reduction compared to the system without the touchdown planner. This substantial improvement in jerk mitigation highlights the

effectiveness of the touchdown planner in providing smooth landing transitions and reducing impact forces. The power usage increases significantly for the knee during takeoff and for the wheel while in aerial phase due to the FPC compensating for lean angle and its rate.

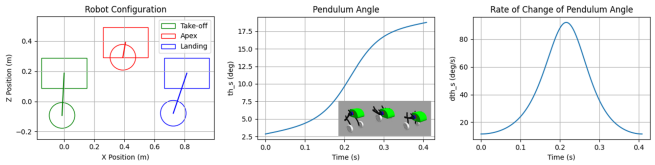

Fig. 9. Demonstration of angular momentum conservation issue during jumping without flight posture control.

Finally, Figure 9 demonstrates the angular momentum conservation issue by using the same jump configuration as Figure 7 but without flight posture control. The results validate equation (33): the simulation accurately reproduces the theoretical prediction, showing failure to maintain control upon touchdown with the pendulum angle increasing from 3° to 18° and angular rate peaking at 92°/s at apex. However, with FPC active (Figure 7), the system ensures safe touchdown within the controller's operational range.

VII. Outlook

This paper presents a modeling and control framework for wheeled bipedal robots, validated through simulation studies of terrain locomotion and jumping. Hardware implementation is the critical next step to validate real-world performance and address practical challenges including actuator dynamics, sensor noise, and computational constraints. Future work includes prototype testing and quantitative comparisons with state-of-the-art WBRs.

VIII. Acknowledgement

This work was supported in part by the King Mongkut's University of Technology Thonburi (KMUTT) and Hospital Automation Research Center (HAC)

References

[1] Boston dynamics. Introducing handle. Youtube, [Online]. Available: https://www.youtube.com/watch?v=-7xvqQeoA8c

[2] V. Klemm et al., "Ascento: A Two-Wheeled Jumping Robot," 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 2019, pp. 7515-7521, doi: 10.1109/ICRA.2019.8793792.

[3] V. Klemm et al., "LQR-Assisted Whole-Body Control of a Wheeled Bipedal Robot With Kinematic Loops," in IEEE Robotics and Automation Letters, vol. 5, no. 2, pp. 3745-3752, April 2020, doi: 10.1109/LRA.2020.2979625

[4] J. Zhang et al., "An Adaptive Approach to Whole-Body Balance Control of Wheel-Bipedal Robot Ollie," 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Kyoto, Japan, 2022, pp. 12835-12842, doi: 10.1109/IROS47612.2022.9981985.

[5] Mao, Nan & Chen, Junpeng & Spyrakos-Papastavridis, Emmanouil & Dai, Jian. (2024). Dynamic modeling of wheeled biped robot and controller design for reducing chassis tilt angle. Robotica. 42. 1-29. 10.1017/S0263574724001061.

[6] Y. Wang, T. Chen, X. Rong, G. Zhang, Y. Li and Y. Xin, "Design and Control of SKATER: A Wheeled- Bipedal Robot With High-Speed Turning Robustness and Terrain Adaptability," in IEEE/ASME Transactions on Mechatronics, vol. 30, no. 2, pp. 1310-1321, April 2025, doi: 10.1109/TMECH.2024.3420390.

[7] D. Liu, F. Yang, X. Liao and X. Lyu, "DIABLO: A 6-DoF Wheeled Bipedal Robot Composed Entirely of Direct-Drive Joints," 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Abu Dhabi, United Arab Emirates, 2024, pp. 3605-3612, doi: 10.1109/IROS58592.2024.10801943.

[8] Nguyen TP, Nguyen H, Ngo T. Design and implementation of a field robot using a parallelogram mechanism. Science Progress. 2024;107(4). doi:10.1177/00368504241291124

[9] W. Zhu, F. Raza and M. Hayashibe, "Reinforcement Learning based Hierarchical Control for Path Tracking of a Wheeled Bipedal Robot with Sim-to-Real Framework," 2022 IEEE/SICE International Symposium on System Integration (SII), Narvik, Norway, 2022, pp. 40-46, doi: 10.1109/SII52469.2022.9708882.

[10] J. Zhang, B. Lu, Z. Liu, Y. Xie, W. Chen and W. Jiang, "A Balance Control Method for Wheeled Bipedal Robot Based on Reinforcement Learning," 2023 42nd Chinese Control Conference (CCC), Tianjin, China, 2023, pp. 2288-2293, doi: 10.23919/CCC58697.2023.10240160.

[11] H. Zhao, L. Yu, S. Qin, G. Jin and Y. Chen, "Design and Control of a Bio-Inspired Wheeled Bipedal Robot," in IEEE/ASME Transactions on Mechatronics, doi: 10.1109/TMECH.2024.3449397.

[12] A. Kollarčík, "Modeling and Control of Two-Legged Wheeled Robot," M.S. thesis, Dept. Control Eng., Faculty Elect. Eng., Czech Tech. Univ. Prague, Prague, Czech Republic

[13] H. Cao, B. Lu, H. Liu, R. Liu and X. Guo, "Modeling and MPC-based balance control for a wheeled bipedal robot," 2022 41st Chinese Control Conference (CCC), Hefei, China, 2022, pp. 420-425, doi: 10.23919/CCC55666.2022.9901979.

[14] H. Chen, B. Wang, Z. Hong, C. Shen, P. M. Wensing and W. Zhang, "Underactuated Motion Planning and Control for Jumping With Wheeled-Bipedal Robots," in IEEE Robotics and Automation Letters, vol. 6, no. 2, pp. 747-754, April 2021, doi: 10.1109/LRA.2020.3047787.

[15] S. Tang, B. Lu, H. Cao and Y. Fang, "Somersaulting Jump of Wheeled Bipedal Robot: A Comprehensive Planning and Control Strategy," in IEEE Robotics and Automation Letters, vol. 10, no. 5, pp. 4460-4467, May 2025, doi: 10.1109/LRA.2025.3551587.

[16] Y. Zhang, L. Zhang, W. Wang, Y. Li and Q. Zhang, "Design and Implementation of a Two-Wheel and Hopping Robot With a Linkage Mechanism," in IEEE Access, vol. 6, pp. 42422-42430, 2018, doi: 10.1109/ACCESS.2018.2859840.

[17] MathWorks, "dlqr," MATLAB Control System Toolbox Documentation. [Online]. Available: https://www.mathworks.com/help/control/ref/lti.dlqr.html.

[18] "ME389-MEM04-PendulumGantry-Guideline," Scribd, 2023. [Online].Available: https://www.scribd.com/document/621034490/ME389-MEM04-PendulumGantry-Guideline.

[19] S. Aranovskiy, I. Ryadchikov, E. Nikulchev, J. Wang and D. Sokolov, "Experimental Comparison of Velocity Observers: A Scissored Pair Control Moment Gyroscope Case Study," in IEEE Access, vol. 8, pp. 21694-21702, 2020, doi: 10.1109/ACCESS.2020.2968221.

[20] Z. Xu, J. Chen, B. Chen, P. Sun, G. Xiao and X. Luo, "A Wheeled Bipedal Robot Capable of Standing Jumps," 2024 IEEE International Conference on Robotics and Biomimetics (ROBIO), Bangkok, Thailand, 2024, pp. 1514-1519, doi: 10.1109/ROBIO64047.2024.10907282.

[21] Y. Zhuang et al., "Height Control and Optimal Torque Planning for Jumping With Wheeled-Bipedal Robots," 2021 6th IEEE International Conference on Advanced Robotics and Mechatronics (ICARM), Chongqing, China, 2021, pp. 477-482, doi: 10.1109/ICARM52023.2021.9536196.

[22] P. J. Antsaklis and A. N. Michel, *Linear Systems*, 1st ed. New York, NY, USA: McGraw-Hill, 1997, ISBN: 978-0-07-041433-4.

[23] I. R. Petersen, "A Kalman Decomposition for Possibly Controllable Uncertain Linear Systems," in *IFAC Proceedings Volumes*, vol.41, no.2, pp.6389–6395, 2008, doi: 10.3182/20080706-5-KR-1001.01078.

[24] Open Source Robotics Foundation (OSRF), "Gazebo Classic" 2002. [Online]. Availble: https://classic.gazebosim.org/

[25] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016

[26] A. Agrawal, R. Verschueren, S. Diamond, and S. Boyd, "A rewriting system for convex optimization problems," *Journal of Control and Decision*, vol. 5, no. 1, pp. 42–60, 2018.