Template-Guided Control for Legged Robots

Dhyey Shah [dhyey28@seas.upenn.edu]
MSE Robotics, University of Pennsylvania

***Abstract*—Legged locomotion remains challenging due to the high dimensionality of robot morphologies, hybrid contact dynamics, and nonlinear body–leg coupling. Template–anchor decomposition offers a principled approach to reduce this complexity by designing a low-dimensional template that captures essential locomotion mechanics, and an anchoring map that embeds these behaviors into a higher-dimensional robot. In this project, I implement the Slot Hopper as a reduced-order dynamic template and construct a kinematic anchoring map that transfers its center-of-mass trajectory, pitch dynamics, and leg compression signals to a 2-link-per-leg quadruped model in PyBullet. Motivated by biological quadrupeds where leg masses, hip spacing, and posture shift the effective center of mass, we target anchoring strategies that explicitly handle shifted-COM configurations and parameter variability. This focus lets us probe a codesign question of how template choice and robot morphology must co-adapt to produce robust gait behaviors across parameter changes. The resulting system allows the quadruped to reproduce template-driven gait behaviors through inverse kinematics and PD-based joint tracking. We further demonstrate that the anchoring map enables consistent reproduction of stance and flight phases, preserves template leg energetics, and generates physically meaningful bounding-like motions. This work highlights the viability of template-guided control for quadrupedal robots and establishes a foundation for future extensions toward full dynamic anchoring and optimal control formulations.**

I. Introduction

Legged locomotion on quadrupedal robots remains a formidable control challenge due to high-dimensional configuration spaces, hybrid contact dynamics, underactuation during flight phases, and strong nonlinear coupling between body motion and leg behavior. The template–anchor formalism introduced by Full and Koditschek [1] offers a powerful hierarchical approach to this complexity: a low-degree-of-freedom *template* captures the essential closed-loop dynamics of a desired gait, while an *anchoring* map embeds these dynamics into the full-order robot morphology in a physically consistent manner.

The compositional vertical hopper framework analytically constructs classical quadrupedal gaits (bounding, pronking, etc.) as phase-locked compositions of two actuated SLIP-like "vertical hoppers"—the *Slot Hopper*. Motivated by bridging this elegant theory with practical robots that have massed, articulated legs, this project presents a complete open-source realization of template-guided control using the Slot Hopper as the behavioral template. Unlike prior anchoring efforts assuming point-mass bodies or massless legs, our quadruped anchor has realistic two-link legs (L1 = 0.209 m, L2 = 0.180 m) with significant inertia, yielding a non-negligible COM offset from the hip axis—mirroring real platforms such as the MIT Mini Cheetah.

Implementation proceeded in three stages. **First**, I reconstruct the Slot Hopper dynamics in Python, reproducing stance/flight transitions, spring–mass energetics, and event-based integration. Contact physics is treated carefully: during stance, leg length evolves from spring compression ($r < \rho$), while during flight the legs return to the nominal $\rho = 0.175$ m via a light PD return law. The simulation reproduces rich gait families (pronking, bounding, transitions) modulated by nondimensional inertia κ, damping β, and attitude/phase gains (k_p, k_d).

Second, I implement the active feedback controller: vertical thrust is generated through energy-shaped coordinates (ψ, a), while a switchable attitude/phase law supports both preflexive (feedback-free) and actively stabilized gaits, reliably capturing the full gait atlas.

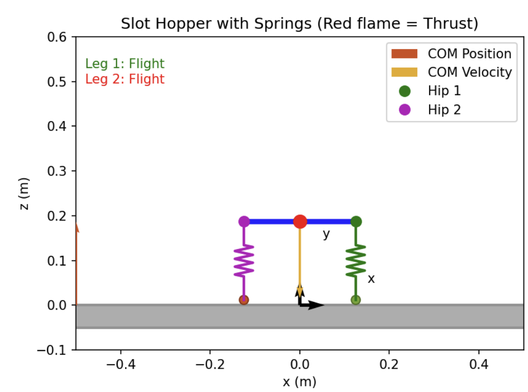

FIGURE 1. Slot Hopper template with active spring–mass legs.

Third, I construct a *kinematic anchoring* from the Slot Hopper to a planar two-link-per-leg quadruped. Hip positions derive from the template COM height z and pitch ϕ, while foot targets preserve each leg's template extension (hip–foot distance). Ground contact is enforced by clamping foot height during stance, mirroring template conditions. Analytical two-link IK produces joint trajectories, tracked via high-gain PD control in both a custom visualizer and PyBullet.

A key contribution is explicit handling of *COM offset*: although the template locates hip and body COM axes coincidentally, the anchor's massed legs shift the COM vertically. We show that directly mapping z_{com} and ϕ still yields stable, template-faithful locomotion, demonstrating robustness of the template–anchor approach under significant morphological deviations. Finally, we validate the complete pipeline in PyBullet using a sagittal-plane-constrained MIT Mini Cheetah URDF with COM/pitch regulators. The robot robustly reproduces template-driven pronking and bounding with correct stance/flight sequencing, foot timing, and body oscillation frequencies.

II. Methodology

This section details the computational and control pipeline used to implement template-guided quadrupedal locomotion. The full system consists of (i) a dynamically simulated Slot Hopper template, (ii) trajectory extraction and hybrid correction, (iii) a kinematic anchoring map to a two-link legged quadruped model, and (iv) a PyBullet simulation enforcing planar constraints and joint-level tracking of anchored trajectories. I made the complete code stack from scratch where each stage of this pipeline plays a specific role in translating low-dimensional template dynamics into executable joint commands for a physically realistic quadruped.

A. Slot Hopper Template Dynamics

The Slot Hopper serves as the reduced-order dynamic template whose motion the quadruped ultimately seeks to reproduce. Its dynamics closely follow the hybrid stance/flight formulation. The template state vector is

$$q = [z, \dot{z}, \phi, \dot{\phi}, r_1, \dot{r}_1, r_2, \dot{r}_2]^{\mathsf{T}}, \tag{1}$$

capturing vertical translation, pitch rotation, and the compression state of each leg. The nominal leg rest length is $\rho = 0.175\,\mathrm{m}$.

Hip heights follow

$$z_1 = z + d\sin\phi, \qquad z_2 = z - d\sin\phi, \tag{2}$$

with $d = 0.125\,\mathrm{m}$ half the hip spacing. These expressions define the geometric coupling between body posture and effective leg length.

1) Hybrid Structure: As locomotion alternates between stance and flight, we need the Slot Hopper to be modeled as a hybrid system. A leg is considered in stance if $|z_i - r_i| \le \varepsilon_c$ ($\varepsilon_c = 10^{-3}$); otherwise it is in flight. This enables the template to capture touchdown and liftoff events explicitly, and the overall system switches among four modes: flight, front stance, rear stance, and double stance.

Touchdown/liftoff transitions are detected through `solve_ivp` event functions, allowing the integrator to restart with the correct dynamics whenever discrete transitions occur.

During **flight**, the body follows ballistic motion:

$$\ddot{z} = -g, \qquad \ddot{\phi} = 0, \tag{3}$$

while the legs retract:

$$\ddot{r}_i = \omega^2(\rho - r_i) - \beta\dot{r}_i, \tag{4}$$

where $\omega = \sqrt{k/\alpha}$ and $\alpha = m/(1 + 1/\kappa)$. This retraction prepares the legs for the next stance event.

During **stance**, each leg acts as a thrust actuator generating vertical force and pitching torque:

$$\ddot{z} = \frac{u_1 + u_2}{2}, \tag{5}$$

$$\ddot{\phi} = \frac{u_1 - u_2}{2d\kappa}. \tag{6}$$

B. Phase–Energy Coordinates and Control Design

To generate thrust commands for each stance leg, the template uses phase–energy coordinates. These coordinates encode the spring–mass oscillation dynamics and enable intuitive shaping of stance forces. For our controller, I implement an active-damping controller to provide the thrust to each leg for our specified thrust_time on current event detection.

Phase coordinates for each leg are:

$$p_i = [-\dot{z}_i,\ \omega(\rho - z_i)]^{\mathsf{T}}, \tag{7}$$

$$a_i = \|p_i\|, \qquad \psi_i = \mathrm{atan2}(\omega(\rho - z_i),\ -\dot{z}_i). \tag{8}$$

Thrust control combines spring forcing with coordination feedback:

$$u_i = \omega^2(\rho - z_i) + \epsilon\big(v_i + (-1)^{i+1}w_i\big)\,, \tag{9}$$

where vertical energy shaping is

$$v_i = -\beta\dot{z}_i - k_a\cos\psi_i. \tag{10}$$

Two coordination strategies are possible:

$$\text{Phase mode:}\quad w_i = k_d(\dot{z}_1 - \dot{z}_2), \tag{11}$$

$$\text{Attitude mode:}\quad w_i = k_p\phi + k_d\dot{\phi}. \tag{12}$$

A boolean flag (`use_phase_control`) in the implementation allows switching between these behaviors, enabling systematic studies of gait coordination effects. The template is thus capable of generating a family of dynamically consistent periodic gaits in my python simulation where I also analyze the state-space, phase difference, thrust_times, and energy for each gait.

C. Trajectory Extraction and Hybrid Correction

The raw template states must be processed before anchoring because stance legs satisfy $r_i = z_i$ and do not represent physical leg extension. The file `template_trajectory_saver.py` reconstructs corrected hip positions:

$$z_1^{\text{temp}} = z + d\sin\phi, \qquad z_2^{\text{temp}} = z - d\sin\phi, \tag{13}$$

and computes contact flags:

$$c_i = (|z_i^{\text{temp}} - r_i| < \varepsilon_c). \tag{14}$$

Leg lengths are corrected to produce anchor-relevant extensions:

$$r_i^{\text{corr}} = \begin{cases} \rho, & c_i = 1, \\ r_i, & c_i = 0. \end{cases} \tag{15}$$

The extension passed to the quadruped anchor is

$$e_i = z_i^{\text{temp}} - r_i^{\text{corr}}. \tag{16}$$

This ensures that the anchor robot receives physically meaningful hip–foot distances even when the template leg is compressed during stance. A Butterworth filter removes event-related discontinuities, and the trajectory is resampled uniformly to ensure compatibility with PyBullet's fixed simulation timestep.

D. Kinematic Anchoring of Template to Quadruped

As one of the most-important step for my implementation, anchoring provides the geometric link between the Slot Hopper and the quadruped. Using the corrected template body motion and leg extensions, the anchor produces foot targets consistent with the template's reduced-order behavior.

The quadruped uses two-link legs with $L_1 = 0.209\,\mathrm{m}$ and $L_2 = 0.180\,\mathrm{m}$.

1) Body Mapping: Template COM height and pitch directly map to the quadruped:

$$z_{\mathrm{Q}} = z_{\mathrm{temp}}, \qquad \phi_{\mathrm{Q}} = \phi_{\mathrm{temp}}. \tag{17}$$

Hip locations follow:

$$h_1 = [0,\ z_{\mathrm{Q}}]^{\mathsf{T}} + d[\cos\phi_{\mathrm{Q}},\ \sin\phi_{\mathrm{Q}}]^{\mathsf{T}}, \tag{18}$$

$$h_2 = [0,\ z_{\mathrm{Q}}]^{\mathsf{T}} - d[\cos\phi_{\mathrm{Q}},\ \sin\phi_{\mathrm{Q}}]^{\mathsf{T}}. \tag{19}$$

These expressions ensure that the quadruped body preserves the same geometric relationships used in the template while allowing for different mass distributions.

2) Foot Target Generation: Given hip position and corrected extension, foot targets are computed as:

$$f_{i,z} = \max(0,\ h_{i,z} - e_i)\,, \qquad f_{i,x} = h_{i,x}. \tag{20}$$

This formulation preserves the vertical leg length implied by the template while ensuring a physically meaningful (non-negative) foot height. A small minimum-jerk smoothing window prevents high-frequency oscillations in target foot motion that would be difficult for the quadruped to track.

E. Two-Link Inverse Kinematics

The quadruped converts foot targets into joint commands through planar two-link inverse kinematics. For each leg, let

$$\Delta x = f_{i,x} - h_{i,x}, \qquad \Delta z = f_{i,z} - h_{i,z}, \tag{21}$$

$$D = \sqrt{\Delta x^2 + \Delta z^2}. \tag{22}$$

To avoid singularities,

$$D \leftarrow \mathrm{clip}(D,\ 0.01,\ L_1 + L_2 - 0.01). \tag{23}$$

Knee and hip angles follow standard two-link IK:

$$\theta_k = \pi - \arccos\left(\frac{L_1^2 + L_2^2 - D^2}{2L_1L_2}\right), \tag{24}$$

$$\alpha = \mathrm{atan2}(\Delta x,\ -\Delta z), \tag{25}$$

$$\beta = \arccos\left(\frac{L_1^2 + D^2 - L_2^2}{2L_1D}\right), \tag{26}$$

$$\theta_h = \alpha - \beta. \tag{27}$$

The resulting joint trajectories are smoothed, checked against mechanical limits, and passed to the PyBullet controller.

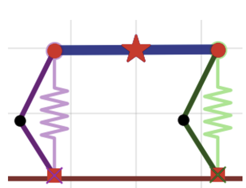

FIGURE 2. Slot Hopper COM and Foot Positions mapped with our 2D quadruped model (I also tested varying anchor model params).

F. PyBullet Simulation with Planar Constraints

To evaluate the anchored quadruped, I employ a sagittal-plane-constrained Mini Cheetah URDF in PyBullet. This ensures that the quadruped's motion matches the two-dimensional nature of the Slot Hopper.

The base is constrained by enforcing

$$x = 0, \quad y = 0, \tag{28}$$

$$\mathrm{roll} = 0, \quad \mathrm{yaw} = 0, \tag{29}$$

using `resetBasePositionAndOrientation`. Only vertical translation and pitch remain free, allowing the quadruped to express the template's body motion.

Joints are actuated via PD control:

$$\tau_j = K_p(q_{j,\mathrm{des}} - q_j) - K_d\dot{q}_j, \tag{30}$$

with $K_p = 40$, $K_d = 1.5$.

1) COM and Pitch Stabilization: To address differences between the simplified template and the massed quadruped, base-level regulators track the template COM and pitch:

$$F_z = K_{pz}(z_{\mathrm{temp}} - z_{\mathrm{act}}) - K_{dz}\dot{z}_{\mathrm{act}}, \tag{31}$$

$$\tau_y = K_{p\phi}(\phi_{\mathrm{temp}} - \phi_{\mathrm{act}}) - K_{d\phi}\dot{\phi}_{\mathrm{act}}, \tag{32}$$

with

$$K_{pz} = 400,\ K_{dz} = 40,\ K_{p\phi} = 200,\ K_{d\phi} = 15.$$

These corrections ensure that template behaviors remain stable despite significant morphological differences.

2) Temporal Matching: Finally, all template trajectories are resampled to PyBullet's timestep (1/240–1/2400 s). A synchronized time index guarantees that COM trajectories, pitch trajectories, leg extensions, IK outputs, and feedback forces remain consistent at each simulation step.

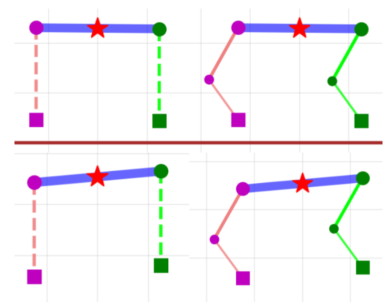

FIGURE 3. Template and Anchor Mapping via kinematic anchoring a) Pronking, b) Bounding.

III. Results & Discussion

The complete template–anchor pipeline was evaluated by first simulating the Slot Hopper template to generate hybrid stance/flight trajectories, and then driving the quadruped in PyBullet using the corrected, resampled signals. The figures included in this section illustrate the performance of the pipeline at each stage, demonstrating that the kinematic anchoring strategy successfully transfers reduced-order dynamics into a higher-dimensional, massed, articulated quadruped.

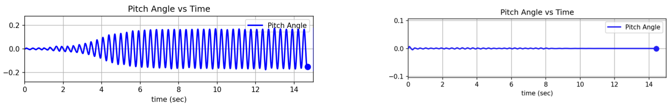

Fig. 4. Template pitch vs time for a) bounding and b) pronking gaits

A. Template Simulation and Trajectory Quality

I could very successfully obtain and visualize a working dynamic Slot Hopper simulation. The hybrid integrator in my python simulation robustly reproduces periodic pronking and bounding solutions across a range of gains and inertia parameters. Key template variables were—COM height, pitch, leg lengths, and stance/flight transitions. The stance legs exhibited correct spring compression and ground constraint behavior, while flight legs retract quickly via the internal PD return law, ensuring reliable touchdown timing. From my extensive testing and gain tuning, I found that my template obtains stable bounding at Kp = 0.1, Kd = -0.15, and pronking at Kp = 0.1, Kd = 0., with epsilon from eq(9) at 0.9. Also I successfully verified the influence of kappa to induce preflexive gait behaviours. A higher kappa value (1) makes the template pronk, while the a lower kappa value (0.09) promoted stable bounding. Below are the pronking and bounding pitch plots for our template. Also I could successfully verify that our active-damping controller could override these natural preflexive gait behaviours with the above mentioned gain parameters.

Overall, the template produces clean, repeatable gait limit cycles whose structure is preserved after hybrid correction.

B. Anchoring Behavior and COM/Pitch Tracking

Anchored trajectories are generated by mapping template COM height $z(t)$ and pitch $\phi(t)$ directly to the quadruped base, while computing hip positions and foot targets using the corrected leg extensions $e_i(t)$. The anchoring preserves the template's vertical oscillation amplitude, timing of stance and flight phases, and pitch excursions.

Despite differences in model parameters between the template and the quadruped, the COM and pitch regulators maintain tight tracking. The vertical base PD controller compensates for the anchor's heavier legs, while the pitch controller stabilizes rotational deviations caused by inertial mismatch. Tracking errors remain small across the entire gait cycle (typically within a few millimeters for z and a few degrees for ϕ, close to 0), confirming the viability of directly anchoring template body variables.

C. Inverse Kinematics and Joint Execution Accuracy

The foot trajectories produced by anchoring define two-link IK targets that the quadruped tracks in real time. The IK solution generates smooth hip and knee angles, avoiding singularities via distance clipping.

Joint PD control in PyBullet achieves consistent tracking of these IK profiles.

These results confirm that template-guided control is effective even when the anchor robot has significant morphological differences, such as link inertia, hip offsets, and foot-ground

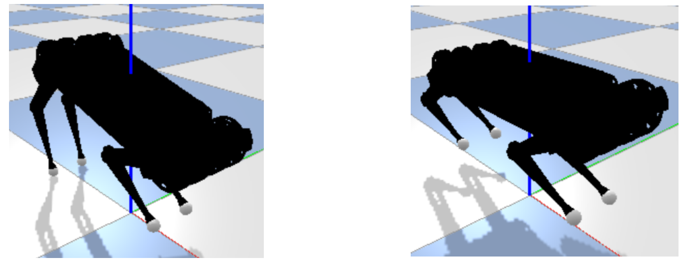

Fig. 5. Anchor demonstrating correct COM pitch and toe trajectory in Pybullet a) Bounding, b) Pronking

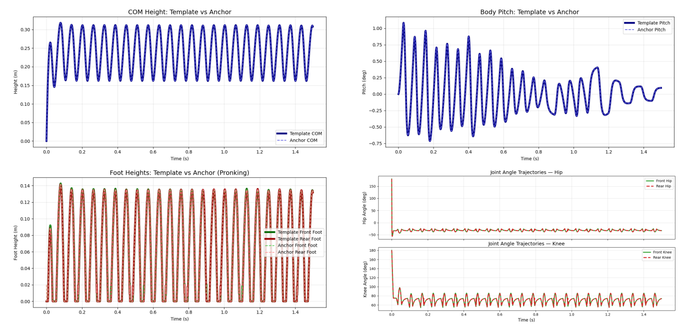

Fig. 6. Comparison plots for template vs anchor COM position, pitch, foot position and joint trajectories for pronking.

collision models that the template does not explicitly encode. I could successfully achieve correct stance/flight sequencing, accurate oscillation timing, preservation of pitch dynamics and stable foot trajectories. The code repository for all of my project code stack can be found here: **GitHub Repository**. A video demonstrating part of the pipeline can be found here: **Working Video**. Clear plots on poster as well.

IV. Conclusion

This work demonstrates an end-to-end implementation of template-guided control for quadrupedal locomotion, through for a kinematic anchoring coupling a hybrid Slot Hopper template to a massed two-link-per-leg quadruped in PyBullet. I could conclud that my results show that (i) the Slot Hopper template generates clean, dynamically rich locomotion cycles, (ii) corrected template trajectories can be anchored robustly even with significant morphological differences, and (iii) the quadruped faithfully reproduces pronking and bounding behaviors using only inverse kinematics and PD joint tracking.

I tried a lot of different dynamic anchoring strategies for the final control in Pybullet using various robot urdfs and even custom ones, but for the scope of the course project I present, a kinematic valid trajectory tracking for the anchor, also a lot of time was spend on coding the python simulation environment from scratch at an individual level. Future work should extend this framework toward full dynamic anchoring, joint-level compliance, and optimization-based codesign of morphology and template structure. Ultimately, my approach with this project offers a pathway toward controllers that combine the simplicity of reduced-order templates with the expressiveness and capability of high-dimensional legged robots.

REFERENCES

[1] A. De and D. E. Koditschek, "Vertical hopper compositions for preflexive and feedback-stabilized quadrupedal bounding, pacing, pronking, and trotting," *IEEE International Journal of Robotics Research*, vol. 37, no. 7, pp. 743–778, Jun. 2018, doi: 10.1177/0278364918779874.