Trajectory Optimization Methods for Energy Efficient Gait Transitions on Multi-Modal Robots

Minh Nguyen
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Dr.
Pasadena, CA 91109
minh02@berkeley.edu

Matthew Suntup
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Dr.
Pasadena, CA 91109
msun0594@uni.sydney.edu.au

Arthur Bouton
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Dr.
Pasadena, CA 91109
arthur.bouton@jpl.nasa.gov

Travis Brown
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Dr.
Pasadena, CA 91109
travis.l.brown@jpl.nasa.gov

William Reid
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Dr.
Pasadena, CA 91109
william.reid@jpl.nasa.gov

Hari Nayar
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Dr.
Pasadena, CA 91109
hdnayar@jpl.nasa.gov

***Abstract*—For robots that achieve locomotion through the movement of articulation joints, having a set of gaits can allow them to navigate variable terrain. Effectively transitioning between these gaits on multi-modal robots is important for maintaining locomotive constraints and reducing unnecessary movements during gait switches, which decreases energy usage. While previous works have examined transitions between specific gaits on certain robots, this work aims to propose and experimentally examine general frameworks for transitioning between gaits with a nonlinear, local optimized trajectory over the articulation joints. The proposed frameworks are targeted towards compute-constrained platforms that require efficient and online trajectory generation. We compare the maximum joint acceleration, transition time, cost of transport, and heading change between four transition frameworks: an optimized Bezier function, optimized Linear waypoints, an optimized Spline, and resetting to a neutral position before each gait. These transition strategies were then implemented and experimentally evaluated on a four-wheeled multi-modal rover.**

TABLE OF CONTENTS

1. INTRODUCTION 1
2. RELATED WORK 2
3. GAITS AND TRANSITIONS 2
4. APPLICATIONS 4
5. IMPLEMENTATION AND EXPERIMENTS 5
6. RESULTS 6
7. CONCLUSION 6
8. FUTURE WORK 6

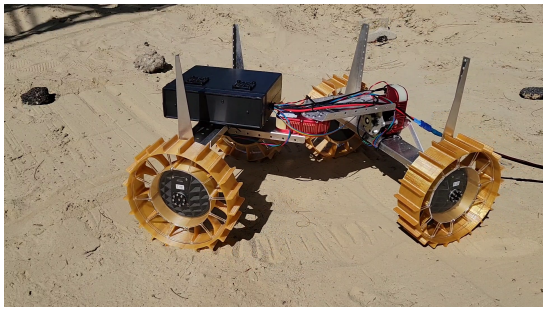
ACKNOWLEDGMENTS 6
REFERENCES 6
BIOGRAPHY 8

1. INTRODUCTION

With increasing demands on robotic mobility, multi-modal robots are useful for their ability to traverse varying terrain. The ability to choose a locomotion mode, or gait, depending on the terrain enables a greater degree of exploration. While there is significant activity in developing multi-modal robotic mechanisms and their accompanying gaits [1] , optimally switching between different gaits remains an open problem.

979-8-3503-0462-6/24/$31.00 ©2024 IEEE

When operating in uncertain environments, it is vital that the system can efficiently and safely vary its gait. This means taking into account joint position and velocity constraints. A gait transition should also avoid consuming excess energy caused by taking unnecessary movements and time. To achieve these goals, it's important that a transition should maintain continuity in both joint space and the workspace.



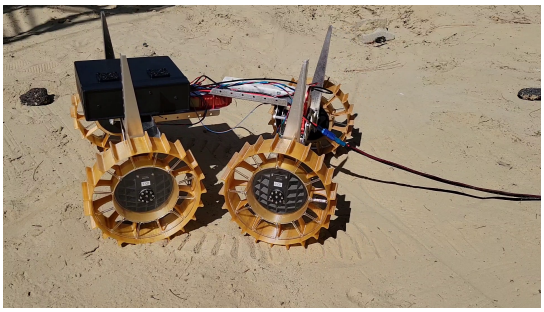


Figure 1. The testing platform, Asterix, executing its squirming gait in the JPL Mini-Mars Yard [1]

The difficulty of implementing optimal gait transitions lies in gaits being extremely platform specific and maintaining trajectory continuity in the workspace is computationally intensive. Multi-modal robots are generally custom built for an environment which leads to widely varying hardware mechanisms and gaits. Trying to optimize a trajectory in the workspace heavily relies on computing the robot's forward dynamics which is expensive to run online. Previous works [2] [3] closely examine specific gaits to generate an optimal transition policy, but this work seeks to present and examine a set of general transition frameworks.

We approach this problem using online nonlinear local trajectory optimization in the joint space, where each gait is represented by its motion through joint space. To maintain trajectory continuity in the workspace, the forward dynamics of the robot are calculated in the cost function, which attempts to minimize trajectory deviation.

These algorithms were all tested on the the Jet Propulsion Labratory (JPL) Steep Terrain Mobility Asterix platform [1], a four-wheeled rover with two articulation joints, and evaluated based on maximum joint acceleration, trajectory deviation, transition time, and cost of transport. We targeted the gait transition solvers for Asterix's Raspberry Pi 4, which is representative of the computational constraints seen on smaller platforms or space rovers.

Our main contributions are:

- Four general, online gait transition frameworks for compute-constrained and non-hybrid multi-modal robots.
- The implementation and evaluation of these transition frameworks on the Asterix platform

2. Related Work

With the development of more complex robots that are capable of different gaits it has been shown that switching to the correct gait depending on the terrain increases a robot's locomotion capabilities. For example, implementing optimal gait switching on a hexapod robot leads to decreased power consumption on varying surfaces and increased traversal capabilities, as certain gaits are required to navigate on difficult terrain types [4].

In addition to selecting the correct gait, transitioning from one gait to another is of great importance. Owaki and Ishiguro [5] show how to parameterize quadraped gaits with a central pattern generator (CPG). The CPG parameters can then be interpolated between to smoothly transition between gaits, assuming the robot remains stable throughout the transition. Haynes and Rizzi [3] exploit the cyclic joint movement and stability of a hexaped robot to transition between gaits through synchronizing the phase of each leg over time to match the desired gait. For non-legged robots, Bing et al. [2] demonstrates how a snake robot's gaits can be parameterized using a CPG which allows for smooth gait transitions. Work with CPG-driven gait transitions shows they work well in systems with high stability and similar gaits.

The gait transition problem has been approached with reinforcement learning as seen in Shao et al. [6]. In this work, a quadrapedal robot successfully learned efficient gait transitions while maintaining stability. Approaching transitions from learning removes any error from model inaccuracies and allows for potentially higher control frequency, although the associated memory or connectivity requirements are unsuitable for compute-constrained embedded systems.

As opposed to prior work, this paper focuses on examining general model-based frameworks for gait transitions on robots that don't necessarily have gaits parameterized by a CPG. Instead, we consider each gait of a robot as a separate joint trajectory and generate transition trajectories to switch between them. We also focus on computationally constrained platforms that require online, efficient trajectory generation, such as robots powered by embedded systems. This systems are limited in their ability to implement the computationally-intensive state-of-the-art methods such as the Mixed-Integer Convex Optimization motion planning done by Aceituno-Cabezas et al [7].

Trajectory optimization is a common framework for such trajectory generation. As shown by Sundaralingam and Hermans [8], it can be implemented as a general framework for hand joint trajectories used for grasping. Similar to their work, we will be proposing general frameworks for generating gait transition trajectories.

3. Gaits and Transitions

Haynes and Rizzi [3] go into great detail about a mathematical representation of gaits and gait transitions. Similar to that work, in this paper we will define a gait as a periodic motion pattern that produces locomotion. A gait is a function that maps from a phase $\mathbb{P} \in \mathbb{R}$ to a robot configuration in joint space $\mathbb{Q} \in \mathbb{R}^n$ where n is the number of joints.

$$g : \mathbb{P} \to \mathbb{Q} \tag{1}$$

Transitions

Transitions between gaits g_1 and g_2 can be seen as a trajectory through joint space that have boundary conditions in g_1 and g_2. Specifically, we can define a gait transition $q(t) \in \mathbb{R}^n$ over the interval $[t_1, t_1 + T] \in \mathbb{R}$ with the constraints:

$$q(t_1) = g_1(t_1) \tag{2}$$
$$q(t_1 + T) = g_2(t_1 + T - \phi) \tag{3}$$

where ϕ is the phase shift of g_2. The simplest transition is when g_1 and g_2 overlap at time t_1, then one would simply set ϕ to satisfy $g_2(t_1 + T - \phi) = g_1(t_1)$. However, this doesn't take into account differences in velocity between those two endpoints, which could result in a jerky transition. Therefore, it's beneficial to enforce continuity in the first derivative too.

$$\dot{q}(t_1) = \dot{g}_1(t_1) \tag{4}$$
$$\dot{q}(t_1 + T) = \dot{g}_2(t_1 + T - \phi) \tag{5}$$

Continuity in the workspace is also desired so the robot doesn't deviate from its intended trajectory during a gait transition. Let $x(t) \in \mathbb{R}^6$ be the robot's resultant trajectory through the workspace by executing $Q(t)$ in the workspace and $G_1(t)$ and $G_2(t)$ be the gait's trajectory through the workspace. Then we desire

$$x(t_1) = G_1(t_1) \tag{6}$$
$$x(t_1 + T) = G_2(t_1 + T - \phi) - G_2(t_1 - \phi) + G_1(t_1) \tag{7}$$

The pose of the robot in the workspace after executing the gait transition should be the same as executing g_2 with ϕ phase shift over $[t_1, t_1 + T]$ with the initial pose of $G_1(t_1)$. We define the workspace trajectory deviation for a gait transition, Δx, to be the difference between the sides of Equation 7.

Three trajectory optimization strategies are implemented and evaluated in this work, as well as a Naive transition strategy. The optimized transition types are a Bezier Curve, Linear Waypoints, and a Spline.

For the three optimization methods, we formulate the optimization problem from a trajectory q in joint space.

$$\min_q w_1 T + w_2 \sum \|\ddot{q}\| + w_3 \|\Delta x\| \tag{8}$$

The cost function is split into three main components: transition time, the norm of joint accelerations, and the norm of the workspace trajectory deviation. w_1, w_2, and w_3 represent the weight associated with each cost. Limiting the gait transition's trajectory time and acceleration prevents the trajectory from consuming excess power. Preventing workspace trajectory deviation is important for uncertain terrain where closely following the planned trajectory is essential.

Naive Transition

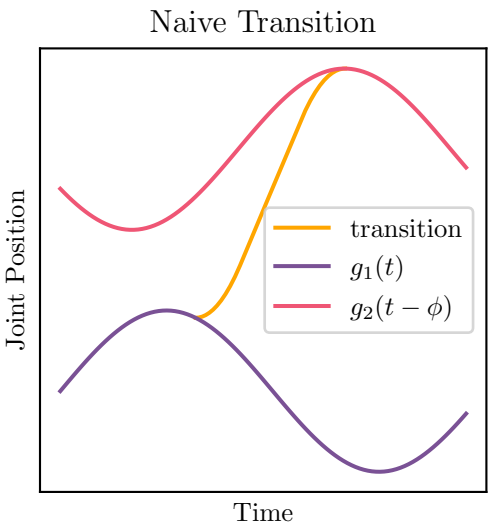


Figure 2. Transition generated by Naive strategy between two sinusoids

The Naive transition strategy has the following policy: at time transition time t_1, generate a trapezoidal trajectory from $g_1(t_1)$ to $g_2(0)$. Execute this trajectory, then continue executing $g_2(t-\phi)$ from $t = 0$, $\phi = 0$. This method has no optimization components. A one dimensional example is shown in Figure 2.

Bezier Trajectory

To switch between the two gaits, the trajectory optimization algorithm outputs parameters T and ϕ, that define a cubic Bezier Curve trajectory through the joint space over $[t_1, t_1 + T]$ and ϕ is the phase shift of g_2. The equation of a cubic Bezier curve is:

$$q(t) = \left(1 - \frac{t-t_1}{T}\right)^3 p_0 + 3\left(1 - \frac{t-t_1}{T}\right)^2 \left(\frac{t-t_1}{T}\right) p_1 + 3\left(1 - \frac{t-t_1}{T}\right)^3 \left(\frac{t-t_1}{T}\right)^2 p_2 + \left(\frac{t-t_1}{T}\right)^3 p_3 \quad (9)$$

The trajectory is constrained to be continuous with the first derivative, velocity, so the function is fully determined by the beginning and end gaits:

$$p_0 = g_1(t_1) \quad (10)$$

$$p_1 = p_0 + \frac{1}{3T}\dot{g_1}(t_1) \quad (11)$$

$$p_2 = p_3 - \frac{1}{3T}\dot{g_2}(t_1 + T - \phi) \quad (12)$$

$$p_3 = g_2(t_1 + T - \phi) \quad (13)$$

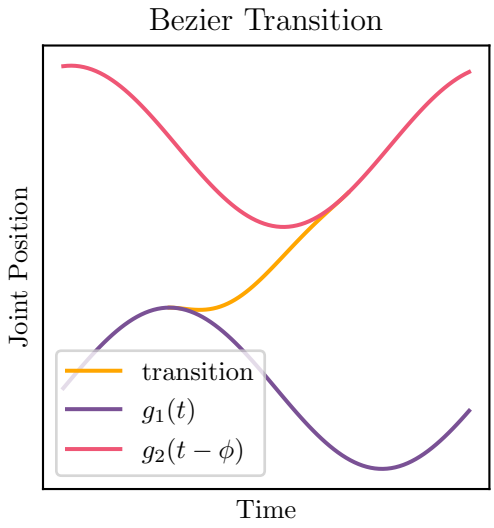


Figure 3. Transition generated by the Bezier trajectory strategy between two sinusoids

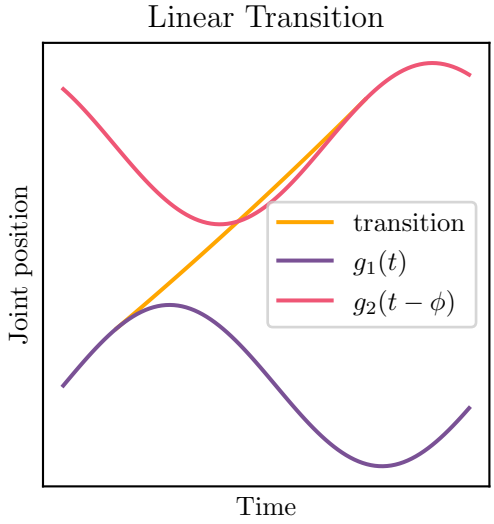


Figure 4. Transition generated by Linear trajectory strategy between two sinusoids

where $p_i \in \mathbb{R}^n$ is a vector of the joint positions.

The acceleration cost is calculated by summing up the second derivative of the function uniformly across the time period. The time cost is T. The trajectory deviation cost is calculated by the method described in the Asterix Platform section.

Linear Trajectory

In this strategy, the optimization algorithm outputs a transition duration T and a series of joint angle waypoints $\Theta \in [\theta_1, \theta_T]$ where

$$\theta_1 = g_1(t_1) \quad (14)$$

$$\theta_T = g_2(t_1 + T) \quad (15)$$

Every waypoint is equally spaced in time, so $\Delta t = \frac{T}{\|\Theta\|}$. We used 100 waypoints in our implementation. Each waypoint is subject to

$$\theta_{min} \leq \theta_t \leq \theta_{max}, \forall t \tag{16}$$

$$-\dot{\theta}_{max} \leq \frac{\theta_{t+1} - \theta_t}{\Delta t} \leq \dot{\theta}_{max}, \forall t \tag{17}$$

The first constraint keeps the trajectory from violating the joint position limits and the second limits the velocity between each set of waypoints.

To prefer linear trajectories, we introduce another cost term:

Define the linear interpolation between $g_1(t_1)$ and $g_2(t_1 + T - \phi)$ as $l(t)$.

$$l(t) = \frac{g_2(t_1 + T - \phi) - g_1(t_1)}{T} \cdot (t - t_1) + g_1(t_1) \tag{18}$$

then the linear cost term is:

$$cost_{linear} = \sum_t \|l(t) - \theta_t\|_2 \tag{19}$$

After the waypoints are generated, they are interpolated through by cubic Bezier curves to create a continuous function.

Spline Trajectory

This strategy is identical to the Linear Trajectory one, but without the linear cost term. In result, the generated transition has higher flexibility in its shape.

4. APPLICATIONS

The gait transition framework described above is suitable for multi-modal robots that maintain the same dynamics across gaits. For hybrid robots, the interpolation based transition trajectories would be unsuitable due to the switching system dynamics. That said, Li et al. [9] use a similar interpolation scheme between gait parameters, as opposed to gait trajectories, to achieve gait transitions on a hybrid robot.

Asterix Platform

In this work, we examine these algorithms on the Asterix platform, a 7.8 kg four-wheeled platform. The four wheels are accompanied by an actively controlled center steering joint and an active rear roll joint.

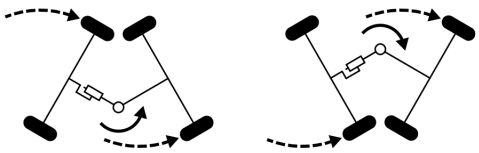


Figure 5. Asterix executing a squirming gait: the center steering joint moves back and forth while the wheels synchronize with its movement. [1]

Asterix is capable of multiple locomotion modes [1], the two we consider in this work are squirming and wheel-walking. Squirming consists of oscillating the center steering joint between its two limits on either side. The wheels are synchronized with the movement of the steering joint to move forward with theoretically no tangential or lateral skidding on a planar surface. Wheel walking also oscillates the center joint, but moves the rear roll joint as well, resulting in the load being shifted to mainly two wheels while one of the wheels is lifted. Which wheel is lifted depends on the overall static equilibrium. As the system dynamics remain the same irrespective of the current gait, Asterix could also transition to and from any other gait, such as standard Ackermann driving with the same framework. However, we focus on squirming and wheel-walking for the following experiments.

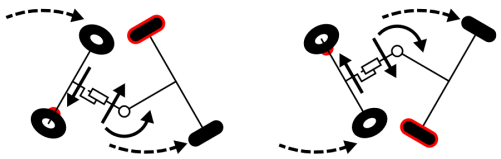


Figure 6. Asterix executing a wheel-walking gait: the center steering joint and the back roll joint moves back and forth while the wheels synchronize with their movement. [1]

To calculate the displacement of the rover given a gait transition, we can integrate the velocity of the wheels over time. Define V_r as the forward velocity of the rover, l_x as the longitudinal distance between the wheels when $\theta = 0$, R to be the wheel radius, ω_i for $i \in [1, 4]$ to be the velocity of wheel i, x_i, y_i to be the Cartesian coordinates of wheel i, and β_i to be the yaw of wheel i.

Then for each wheel:

$$\frac{d}{dt}x_i = R\omega_i \cos(\beta_i) \tag{20}$$

$$\frac{d}{dt}y_i = R\omega_i \sin(\beta_i) \tag{21}$$

$$\frac{d}{dt}\beta_i = \frac{V_r}{l_x}\tan(-\frac{\theta}{2}) + (-1)^{\frac{i(i+1)}{2}}\frac{\dot{\theta}}{2} \tag{22}$$

Then the Cartesian coordinates x, y at time T of the whole rover can be found by summing the wheel displacements, taking care to note that β changes each timestep.

$$\begin{bmatrix} x \\ y \\ \beta \end{bmatrix} = \sum_i \int_0^T \begin{bmatrix} dx_i \\ dy_i \\ d\beta_i \end{bmatrix} dt \tag{23}$$

Evaluation Metrics

Heading deviation is calculated by taking the difference between the rover's heading before and after a transition. The heading being the yaw of the rover in respect to the world frame.

Cost of transport (COT) is a dimensionless number that quantifies the energy efficiency of a gait transition. We expect cost of transport to be related to maximum joint acceleration as lower joint accelerations leads to less energy usage and higher cost of transport.

$$COT = \frac{E}{mgd} \tag{24}$$

where E is the mechanical energy produced by the motors during the transition, m is the mass of the rover, and d is the distance travelled during the transition.

Transition time is the duration of the transition, which is a direct output of the optimization.

The maximum joint acceleration is calculated for both the steering and bogie joints. The maximum rotational acceleration between consecutive data points is found for each transition.

5. Implementation and Experiments

Robot Deployment

The trajectory optimization problems were solved using the NLOPT library [10] using Sequential Least Squares Programming (SLSQP) [11] in C++. The solver was set with a maximum time of 0.5 seconds with the previously described cost functions subject to the joint velocity and position constraints. The algorithms were deployed on a Raspberry Pi 4B onboard Asterix which invoked the C++ solver through a Python front end. The forward dynamic calculation was run using the Boost ODE library. The optimization time limit was chosen to ensure a smooth gait transition, as longer times resulted in the joints straying too far from the initial position specified in the optimization problem while the solver ran.

The program outputted joint positions which were followed by the feedback controllers of the articulation. Another controller synchronized the wheel velocities with the steering joint.

Experimental Protocol

Experiments were executed in the JPL Mini Mars Yard, an outdoor sandpit. A Vicon motion capture system was setup to record motion capture data. The Raspberry Pi recorded onboard telemetry to keep track of power draw and transition requests.

Figure 7. Vicon motion capture setup to evaluate gait transitions on the Asterix platform

For each transition type, three different weight distributions for the cost function were empirically selected: minimize transition time, minimize workspace trajectory deviation, or a balance of both. For each weight distribution, three identical experiments were run. In each experiment the rover switched gaits from squirming to wheel-walking at 5 uniform points over the period of the squirming gait. In the same run, the rover repeated the uniformly distributed switching, but from wheel-walking to squirming instead.

Transition Type	Minimize	w_1	w_2	w_3
Bezier	Duration	30	1	1
Bezier	Deviation	1	1	20
Bezier	Balanced	5	1	1
Linear	Duration	20	5	1
Linear	Deviation	4	5	20
Linear	Balanced	4	5	1
Spline	Duration	20	5	1
Spline	Deviation	4	5	20
Spline	Balanced	4	5	1

Table 1. Weight distributions for each trajectory optimization strategy. w_1 corresponds to duration cost, w_2 corresponds to average acceleration, and w_3 corresponds to workspace deviation

A combination of Vicon motion capture data and onboard telemetry was time synced to create the data for each experimental run.

Performance Evaluation

The heading deviation was taken directly from the motion capture data, looking at the robot's pose before and after a transition.

To calculate the Cost of Transport, the mechanical energy expended during the transition was calculated from the articulation and wheel motors.

$$E = E_{wheel} + E_{articulation} \tag{25}$$
$$E_{wheel} = k_t I \omega_w \tag{26}$$
$$E_{articulation} = \tau \omega_a \tag{27}$$

The articulation motors directly reported torque, while the torque constant on the wheel motors, k_t, was experimentally determined to be $1.45\ \mathrm{N\,m\,A^{-1}}$.

The distance traveled is the euclidean distance between the initial position and the final position of the gait transition trajectory

$$d = \|G_2(t_1 + T - \phi) - G_1(t_1)\|_2 \tag{28}$$

which is taken directly from the motion capture data. The mass of the rover is 7.8 kg.

It is important to note that the cost function for Bezier is not the same as the Linear and Spline cost functions as the weights are different. This is due to differences in the calculation for each cost function. The weights were subjectively selected so metrics that were not directly optimized for, such as the Cost of Transport metric, are not directly comparable between the groups of optimizations. Despite this, it is fruitful to look at the cost of transport for each strategy as a way to qualify their individual performance.

6. Results

We now discuss our results with the performance metrics represented in Figures 8-11 as column graphs for each configuration of transition strategy.

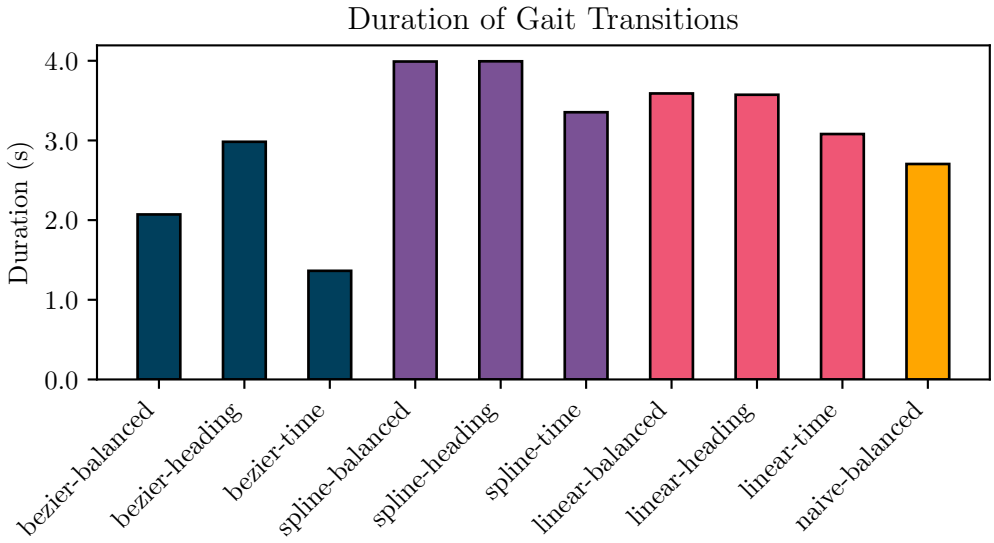


Figure 8. Comparison of average transition duration for each gait-transition type. During each run, the average of the 10 uniformly distributed transitions are taken. Colors indicate the four transition frameworks, "Bezier", "Spline", "Linear", and "Naive", (each of which has three variations of cost functions except the Naive type).

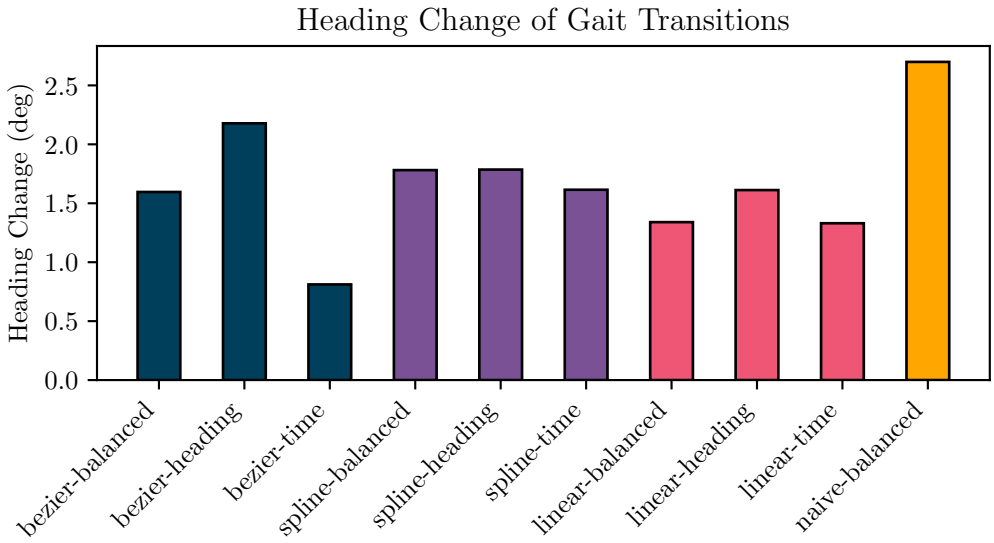


Figure 9. Comparison of average heading deviation caused by transition for each gait-transition type. During each run, the average of the 10 uniformly distributed transitions are taken. Colors indicate the four transition frameworks, "Bezier", "Spline", "Linear", and "Naive", (each of which has three variations of cost functions except the Naive type).

Naive Transition

The Naive transition has an extremely low cost of transport at the expense of a long duration and the highest average heading change. This is to be expected since the Naive strategy always resets to the initial position of g_2: $g_2(0)$, meaning it biases towards that side. Due to the long transition time, the rover travels a large distance so the cost of transport is low, but there is no control over workspace deviation.

Bezier Transition

The Bezier Transition strategy is characterized by low duration, average heading change, and low joint acceleration. This could be due to the constrained nature of the trajectory, there are only two free parameters, so it doesn't have much freedom to correct heading deviation. On the other hand, its simple trajectory leads to short transition times which is less time to cause a trajectory deviation. Since the curve is a continious function, the joint acceleration is expected to be lower as there's no sudden changes between waypoints that are present in the Linear and Spline strategies.

Linear Waypoints Transition

The Linear transitions have high joint acceleration and duration, while benefiting in the heading change. It's important to note that its improvement in heading change is relatively small, about two degrees compared to the Naive transition, so this metric can be weighted less heavily. The Linear transition trajectory has enough freedom with its waypoints to correct the trajectory deviation but it would come at the heavy expense of duration and acceleration.

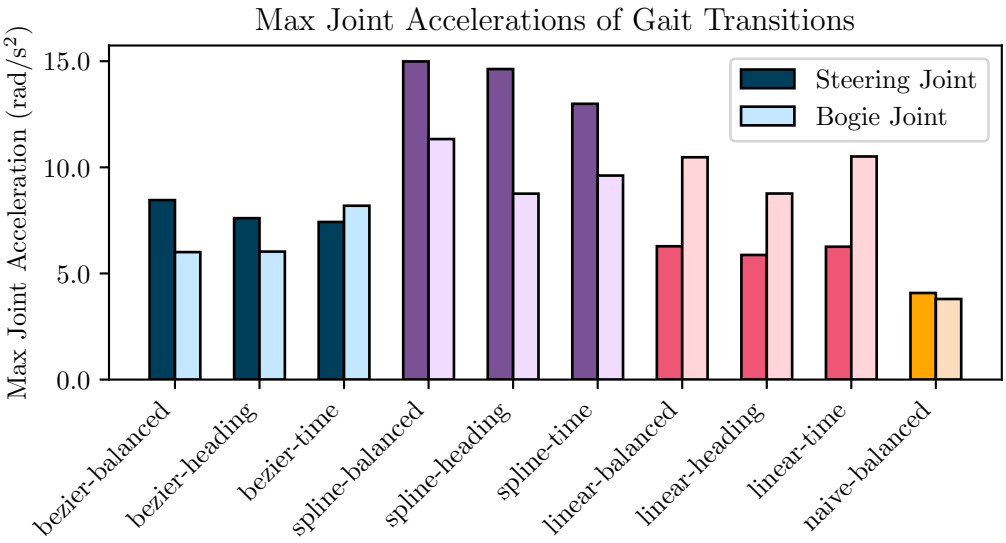


Figure 10. Comparison of maximum rotational acceleration experienced at the steering and bogie joints over the transition for each gait-transition type. Colors indicate the four transition frameworks, "Bezier", "Spline", "Linear", and "Naive", (each of which has three variations of cost functions except the Naive type). For each pair of columns, the left column (darker color) represents the steering joint and the right column (lighter color) represents the bogie joint.

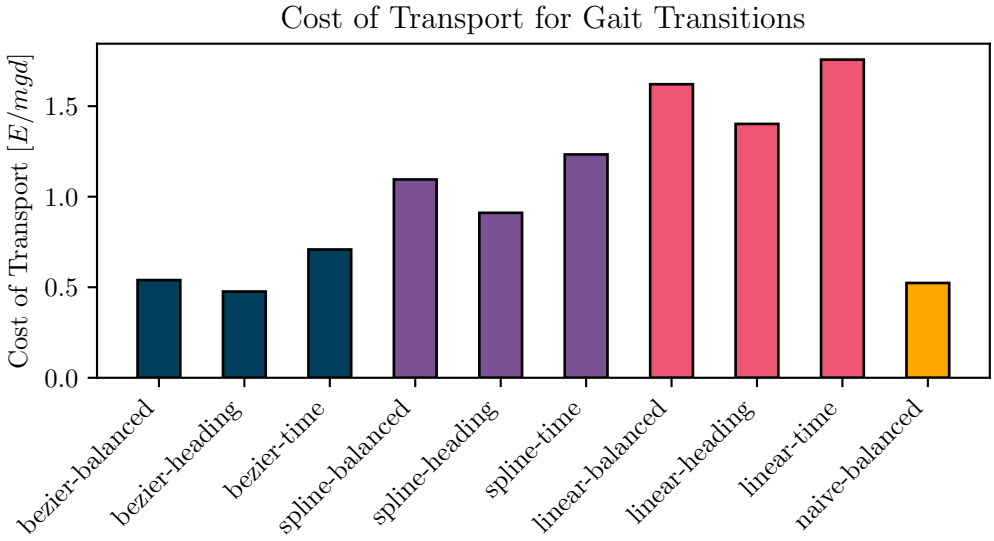


Figure 11. Comparison of average cost of transport over the 3 runs for each gait-transition type. During each run, the average of the 10 uniformly distributed transitions are taken. Colors indicate the four transition frameworks, "Bezier", "Spline", "Linear", and "Naive", (each of which has three variations of cost functions except the Naive type).

Spline

The Spline strategy performs poorly in the joint acceleration method, possibly due to the lack of constraints between waypoints. Its freedom to place waypoints may have run into computational issues where the optimization could not converge in the allotted time.

Overall, the Bezier method performed the best, possibly due to its short transition durations which led to less movement of the platform and, in turn, better cost of transport and heading deviation. The Bezier method also had the lowest cost of transport out of all strategies which speaks to its energy efficiency. The waypoint methods, Spline and Linear, had more freedom in their trajectories but it may be that in wheeled platforms, maintaining continuity in the workspace trajectory is too expensive in terms of duration and cost of transport.

Another explanation comes from the optimization time which was limited to 0.5 seconds. This favors optimization problems with less degrees of freedom as they can reach their optima faster. It could be that the Bezier method with its two degrees of freedom outperformed the Spline and Linear methods because the Bezier method was less computationally intensive to optimize.

7. Conclusion

We presented four general transition frameworks for gait transitions. Each was evaluated based on cost of transport, duration, maximum joint accelerations, and workspace trajectory deviation. While the strategies were framed to be for general frameworks for fixed-dynamic multi-modal robots, they were evaluated on the wheeled Asterix platform. We outline how to implement the described strategies on real hardware and how it performed. The results showed that the best performing transition strategy was the Bezier strategy which may be primarily explained through its short transition times and simpler problem formulation. In other platforms, the optimization weight distributions should be tuned to the task at hand. For example, the frequency of transitions affects which strategy is optimal as one could trade off the higher per-transition duration cost for better trajectory deviation.

8. Future Work

To draw more conclusions about the effectiveness of the different frameworks, it would be beneficial to subject them to a unified cost function. In this way we can more accurately compare their effectiveness in minimizing the cost and metrics that are not directly optimized, such as cost of transport.

For robots with a higher number of kinematic constraints, adding those into the frameworks as well as collision avoidance would be increase the safety of the generated transitions.

Implementing these frameworks on other four-wheeled robots with articulation joints would be a good comparison to our testing platform. The implementation of other gaits to transition between would also shed more insight into the performance of each strategy. As stated previously, these frameworks are unsuitable for robots with hybrid system dynamics but restructuring the frameworks to interpolate between gait parameters, such as speed, instead of trajectories could bridge this gap.

Rather than using optimization, a reinforcement-learning based trajectory generator could be useful as the optimization has to be time constrained when run online. Although learning methods might be inhibited by the limitations of an embedded system.

Acknowledgments

The research described in this publication was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

References

[1] A. Bouton, "Experimental Study of Alternative Rover Configurations and Mobility Modes for Planetary Exploration," 2023.

[2] Z. Bing, L. Cheng, G. Chen, F. Röhrbein, K. Huang, and A. Knoll, "Towards autonomous locomotion: CPG-based control of smooth 3D slithering gait transition of a snake-like robot," *Bioinspiration & Biomimetics*, vol. 12, no. 3, p. 035001, Apr. 2017.

[3] G. Haynes and A. Rizzi, "Gaits and gait transitions for legged robots," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, May 2006, pp. 1117–1122.

[4] N. Kottege, C. Parkinson, P. Moghadam, A. Elfes, and S. P. N. Singh, "Energetics-informed hexapod gait transitions across terrains," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 5140–5147.

[5] D. Owaki and A. Ishiguro, "A Quadruped Robot Exhibiting Spontaneous Gait Transitions from Walking to Trotting to Galloping," *Scientific Reports*, vol. 7, no. 1, p. 277, Mar. 2017, number: 1 Publisher: Nature Publishing Group.

[6] Y. Shao, Y. Jin, X. Liu, W. He, H. Wang, and W. Yang, "Learning free gait transition for quadruped robots via phase-guided controller," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1230–1237, 2022.

[7] B. Aceituno-Cabezas, C. Mastalli, H. Dai, M. Focchi, A. Radulescu, D. G. Caldwell, J. Cappelletto, J. C. Grieco, G. Fernández-López, and C. Semini, "Simultaneous contact, gait, and motion planning for robust multilegged locomotion via mixed-integer convex optimization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2531–2538, 2018.

[8] B. Sundaralingam and T. Hermans, "Relaxed-Rigidity Constraints: Kinematic Trajectory Optimization and Collision Avoidance for In-Grasp Manipulation," Jun. 2018.

[9] Q. Li, L. Qian, S. Wang, P. Sun, and X. Luo, "Towards generation and transition of diverse gaits for quadrupedal robots based on trajectory optimization and whole-body impedance control," pp. 2389–2396, 2023.

[10] S. G. Johnson, "The NLopt nonlinear-optimization package," https://github.com/stevengj/nlopt, 2007.

[11] D. Kraft, "Algorithm 733: TOMP–fortran modules for optimal control calculations," *ACM Transactions on Mathematical Software*, vol. 20, pp. 262–281, 1994.

BIOGRAPHY

Minh Nguyen *is an intern at JPL and a student at the University of California, Berkeley. His research interests are in control and hardware acceleration. He is currently working towards a B.A. in Computer Science.*

Matthew Suntup *is a Visiting Student Researcher at JPL and a student at the University of Sydney, Australia. His research focuses on steep terrain mobility for planetary rovers. He received both his B.Eng. Honours (Mechatronic Engineering) majoring in Space Engineering and B.Sci. (Advanced) majoring in Physics and Computer Science from the University of Sydney in 2023.*

Arthur Bouton *is a Robotics Technologist at JPL. His primary areas of research include wheel-on-limb hybrid locomotion, mechanical design, machine learning and control. He received his M.Sc.Eng. degree in Advanced Systems and Robotics from E´cole Nationale Sup´erieure d'Arts et M´etiers ParisTech in France and his Ph.D degree from Sorbonne Universit´es, Universit´e Pierre et Marie Curie, Paris, in 2017. After a postdoc at Surrey Space Centre, Guildford, UK, he joined JPL in 2021.*

William Reid *is a JPL Robotics Technologist investigating novel mobility systems for planetary surface exploration. William received a PhD in Robotics from the Australian Centre for Field Robotics at the University of Sydney in 2018. He has Bachelor's degrees in Mechatronics Engineering and Computer Science from the University of Melbourne.*

Travis Brown *is a robotic systems engineer in the Robotic Mobility Group at JPL. He holds a Ph.D. in robotics from the University of Notre Dame, where he studied planning and control of underactuated bipedal robots. He has been with JPL since 2016, working on a variety of projects such as the Axel Rappelling Rover, Mars Helicopter (Ingenuity), Moon Diver Mission Concept, Perseverance Rover, ColdArm, and many others. He is interested in the design and control of advanced robotic mobility systems. He is currently serving as the deputy chief engineer for Ingenuity.*

Hari Nayar *Hari Nayar is a principal technologist in the Mobility and Robotics Systems Section at JPL. He is the Principal Investigator of the Steep Terrain Mobility task at JPL where this work was done. He was a Visiting Lecturer in the Mechanical and Aerospace Engineering Department at UCLA. His research interests include analysis & modeling, manipulation, mobility, teleoperation and autonomy for space and medical robotics applications. He received his ScD in Mechanical Engineering at MIT.*