Simulation-in-the-loop for Planning and Model-Predictive Control

Christoffer Heckman, Nima Keivan, and Gabe Sibley
Department of Computer Science
University of Colorado
Boulder, CO 80309
Email: christoffer.heckman@colorado.edu

I. Introduction

The primary goal of optimal planning and control algorithms is to lay out behaviors which conserve time and/or fuel and apply these findings to real-world systems. There are several popular lines of development; traditional, continuous-time control theory often has as a prerequisite that some technical assertions about the plant dynamics be made, either in the form of explicit equations or propositions. For nonlinear and high-order systems, this can be a significant challenge. Another approach involves using motion primitives combined with stochastic search algorithms such as in RRT-based methods [9, 8, 7]. These searches invariably sample infeasible trajectories and largely provide an open-loop approach that is not robust to model or state uncertainty. In light of these solution-specific requirements and drawbacks, we wish to develop an intuitive, computationally efficient method that is robust to uncertainty.

In this work, we present an approach to both the planning and control of lightweight ground vehicles that makes use of physical simulation and efficient computation to perform these tasks concurrently and in real-time. As in our previous work by Keivan and Sibley [5], we apply this physical simulator both to test the feasibility of planned paths with respect to the vehicle dynamics and the terrain, as well as in determining the control signals required in order to follow the specified path. The use of high-fidelity physical simulation within the planning and control loops provides an extremely dense feed-forward model for vehicle and terrain dynamics. Furthermore, the approach allows for considering arbitrarily sophisticated models of these features. This method relies crucially on the notion that accurate and fast simulation can be used to generate feasible plans and control signals that generate desired states in the future.

To solve for both feasible plans and control signals that tend toward the planned path, we employ an optimization-based two-point boundary value problem (BVP) solver. The solver takes as input a starting configuration, a terrain model, and a goal configuration; combined with the dynamics of the vehicle as it drives on the terrain, this is a well-posed two-point BVP. In finding the solution to this problem, a feasible trajectory is generated for the vehicle. Using the same approach, a controller is then calculated to drive the vehicle.

II. Problem Statement

We consider the deployment of a ground vehicle on a given three-dimensional terrain; the vehicle's challenge is to drive through a series of waypoints along the terrain. Let $\boldsymbol{x}_\alpha$ be the vector of position and velocity on the surface of the terrain at waypoint α, and $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\theta})$ be the dynamics of the vehicle driving on the terrain which has parameters $\boldsymbol{\theta}(\boldsymbol{x})$ describing it. The resulting two-point BVP is straightforward; we are solving for $\boldsymbol{x}(t)$ such that:

$$\boldsymbol{x}'(t) = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\theta}) \quad \boldsymbol{x}(t_\alpha) = \boldsymbol{x}_\alpha \tag{1}$$

The waypoints have coordinates $\boldsymbol{x} = [\boldsymbol{T}_{lv}\ \boldsymbol{v}]$, where $\boldsymbol{T}_{lv} \in \mathbb{R}^{4\times 4}$ is the transformation matrix from vehicle to the local coordinates, and $\boldsymbol{v} \in \mathbb{R}^3$ is the velocity at the given point. Note that the t_α may be chosen within an allowable range according to limits on the dynamics of the vehicle controls; generally to make the problem well-posed there is an additional stipulation that the control terms be bounded and that the BVP is solved for minimal time. This is a classical optimal control problem; we use it in both our "planning" problem (finding a feasible trajectory given the dynamics of the system) and the control problem. The controller will minimize the difference between the forward-simulated vehicle and the desired trajectory given by the planned path. In order to accomplish both planning and control, we rely on an accurate model of the environment and the vehicle.

III. Methodology

Our approach involves simulating the effect of control inputs to the system as they affect the system dynamics interacting with the physical features of the terrain. We use the Bullet physics simulator [3] to model the system dynamics in our computations. We solve for feasible trajectories given the current & goal poses and the intermediate terrain using an optimization framework. When calculating the control inputs, we still use the simulation-in-the-loop controller which solves a BVP via optimization that relies on finite-difference estimations of the gradient of the vehicle model.

We follow the motivation of model-predictive control in order to ascertain implied underlying parameters in the system. However, in this approach those underlying parameters are in fact intrinsic and extrinsic physical parameters, e.g. the

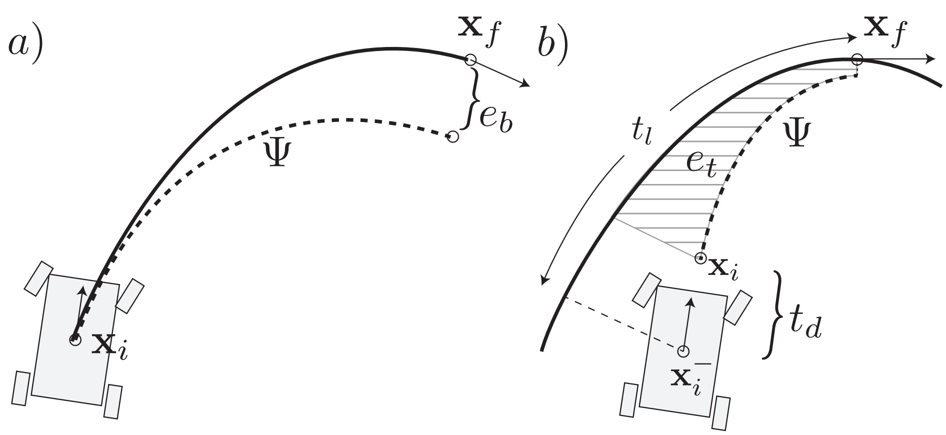

Fig. 1: a) Boundary cost used for planning trajectories to waypoints. b) Trajectory cost weighted for integral of difference over the planned trajectory; used for optimizing the control input functions.

wheel base of the vehicle and its spring coefficients. These parameters are used in the physical simulator to determine the overall behavior of the system with different control inputs.

A. Planning

As previously stated, planning involves determining feasible trajectories along the terrain that are capable of optimizing for some quantity, be it minimizing engine use or reaching a goal in minimum time. We treat this problem by solving a BVP between waypoints that have been specified *a priori*. This requires that waypoints are placed to ignore certain impassable obstacles since each waypoint is sacrosanct in the resulting global path.

To plan the path between waypoints, we run a series of simulations forward in time with various control inputs $\boldsymbol{c}(t)$ and minimize the error in satisfying the system dynamics in order to determine a feasible path. The cost function we wish to minimize between each pair of waypoints is:

$$e_b = \left\| \boldsymbol{x}_{lf} \boxminus \boldsymbol{\Psi}_l \left(\boldsymbol{\theta}, \boldsymbol{c}, \boldsymbol{x}_i, t_f \right) \right\|_2 \tag{2}$$

where we rename $\boldsymbol{x}_f$ to be the coordinate of the next waypoint, t_f is the time at which that waypoint is reached, and $\boldsymbol{\Psi}_l$ is the simulated estimate of the path for a given control input $\boldsymbol{c}$. We assert that whatever small perturbations from reality the physical simulator presents are not significant in producing deleterious dynamical behaviors, or formally $\boldsymbol{\Psi} \asymp \boldsymbol{f}$. This is conceivably achievable by using appropriately detailed physical simulators. The operator $\boxminus$ calculates the velocity and pose error between two vehicle states; see Figure 1a for a schematic of this arrangement. For each of the proposed inputs the error corresponding to the simulated trajectory is used as a cost function over which we minimize via an optimization framework.

B. Control

Once a planned path has been calculated, we employ a model-predictive controller such as that employed by [1] to follow the path; this involves running a series of simulations forward with various control inputs $\boldsymbol{c}(t)$ and optimizing for the inputs that minimize a specified cost function. This cost function when tracking a reference trajectory for timestep t_j is:

$$e_t = \left\| \left\| \sum_{j=0}^{n} w_j \boldsymbol{x}_{lj} \boxminus \boldsymbol{\Psi}_l \left(\boldsymbol{\theta}, \boldsymbol{c}, \boldsymbol{x}_i, t_j \right) + t_f \right\| \right\|_2 \tag{3}$$

where $w_j \in \mathcal{W} = (w_0, \ldots, w_n)$ is a weight imposed such that current near-future simulation offset from the planned path is of greater importance to the optimization. The weights may also be used to specify certain degrees of freedom must be more strictly enforced than others, e.g. that the vehicle should not be allowed to roll significantly.

C. Optimization

The goal in solving the control problem is to minimize in a least-squares sense the weighted cost. We establish a weight matrix $\boldsymbol{W} = \text{diag}(w_j)$, a basis for the control inputs as a function of time $\boldsymbol{p}$, an error vector for the proposed control input $\boldsymbol{r}$ and a Jacobian $\boldsymbol{J} = \frac{\partial \boldsymbol{e}}{\partial \boldsymbol{p}}$. The optimization proceeds by minimizing the error vector in the equation $\left(\boldsymbol{J}^T \boldsymbol{W} \boldsymbol{J}\right) \boldsymbol{p} = \boldsymbol{J}^T \boldsymbol{W} \boldsymbol{r}$. Each column of the Jacobian corresponding to a simulated trajectory is calculated via separate threads in order to vastly improve performance. In the control optimization, we find that if the time horizon over which we simulate t_l is too short then the optimization problem has a null solution space; this is often resolved using heuristics to determine the time horizon as pursued by Grieder et al. [4]. However, once an acceptable t_l is found, our optimization reduces the time horizon by construction.

IV. Results

The model learning via optimization over motion samples as described in Section III was found to be effective in learning the relevant car parameters. This demonstrates that the search space for these parameters is convex and that through normal maneuvers the relevant physical parameters

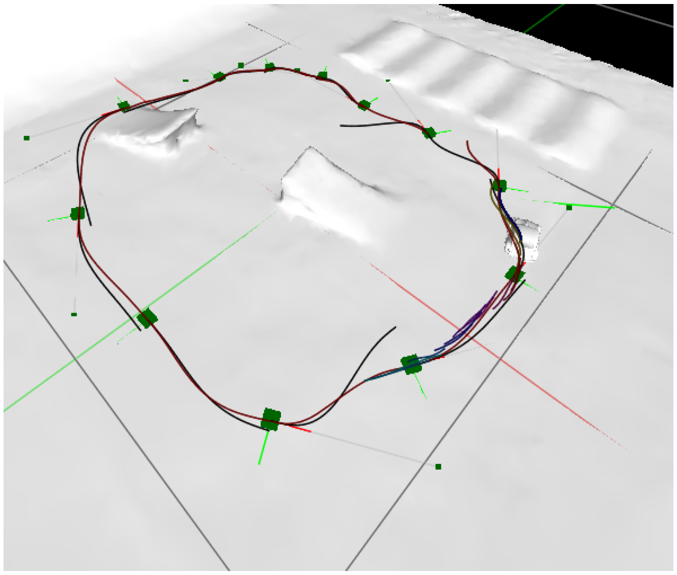

Fig. 2: Front-end of planning and control application to drive a car around a model of a lab space with ramps and varying terrain. Planned trajectories (continuous red line) connect the waypoints (green cubes) and also visualize control signals (discontinuous black lines); control candidates over which optimization occurs are simulated with their results as the multicolored discontinuous lines.

are observable. The machine learning framework effectively provided approximations for otherwise difficult-to-measure quantities such as tire and friction coefficients. In order to make the problem more tractable, we chose to search for control inputs over a low-dimensional space, rather than the infinite-dimensional space of continuous functions in which the boundary value problem is originally posed. We therefore parametrized the control inputs by searching over the space of Bezier curve control points as demonstrated by Choi et al. [2]; this guaranteed immediately that certain physical constraints be considered, e.g. that the steering on a four-wheel vehicle is continuous in time, as is the voltage to the engine. We also found that Bezier curves provide computational efficiency in the required search. Figure 2 shows the resulting planner and control candidate paths as a simulated vehicle drive along a model terrain.

The simulation-in-the-loop framework was tested experimentally on a number of challenging terrain features, including jumps, loop-the-loops and quarter pipes. The experiments were carried out using a small remote-controlled car with all computations made on-line but on a networked system. Pose was provided by a Vicon motion capture system and fused with IMU data in order to interpolate between motion capture-provided data, and operated when such data was not available (such as when upside-down).

V. Conclusions

Our effort applies modeled physics using fast simulation in order to ascertain the effect of control inputs and tune them accordingly. Feedback to disturbances is provided via the minimization of trajectory tracking error as in Eq. (3). Therefore even if desired physical models are not analytically tractable, e.g. they rely on lookup tables or special functions without tabulated derivatives, they may be fused into the models over which simulation is executed. This enables the use of new and exotic models for such challenging features as granular terrain or all-wheel drive vehicles. The approach has been successful in local planning, continuous re-planning and controlling a car by guiding it toward a planned trajectory. Finally, the potential for accomplishing change detection, such as detecting a flat tire, by comparing observed behavior with simulated behavior is an exciting possibility that could be introduced in an intuitive fashion.

VI. Future work

In this work, we treated this problem by solving a BVP between waypoints that had been specified *a priori*. This required that waypoints were wisely placed, since each waypoint would be fixed in the resulting global path. Furthermore, the solving of this series of BVPs was a computationally expensive process. We therefore wish to use of machine learning approach to determine the feasible paths via a convolutional neural network (CNN). CNNs are widely used in the image recognition community [6], and are uniquely well-suited in addressing the problem of planning over terrain.

Acknowledgments

We gratefully acknowledge the generous support of Toyota and Google.

References

[1] Patrick Bouffard, Anil Aswani, and Claire Tomlin. Learning-based model predictive control on a quadrotor: Onboard implementation and experimental results. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 279–284. IEEE, 2012.

[2] Ji-wung Choi, Renwick Curry, and Gabriel Elkaim. Path planning based on bézier curve for autonomous ground vehicles. In *World Congress on Engineering and Computer Science 2008, WCECS'08. Advances in Electrical and Electronics Engineering-IAENG Special Edition of the*, pages 158–166. IEEE, 2008.

[3] Erwin Coumans. Bullet physics library. URL http://bulletphysics.org/.

[4] Pascal Grieder, Francesco Borrelli, Fabio Torrisi, and Manfred Morari. Computation of the constrained infinite time linear quadratic regulator. *Automatica*, 40(4):701–708, 2004.

[5] Nima Keivan and Gabe Sibley. Realtime simulation-in-the-loop control for agile ground vehicles. In Ashutosh Natraj, Stephen Cameron, Chris Melhuish, and Mark Witkowski, editors, *Towards Autonomous Robotic Systems*, volume 8069 of *Lecture Notes in Computer Science*, pages 276–287. Springer Berlin Heidelberg, 2014.

ISBN 978-3-662-43644-8. URL http://dx.doi.org/10.1007/978-3-662-43645-5_29.

[6] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[7] Michael Otte and Emilio Frazzoli. Rrt-x: Real-time motion planning/replanning for environments with unpredictable obstacles. In *Algorithmic Foundations of Robotics XI*, pages 461–478. Springer, 2015.

[8] Alejandro Perez, Robert Platt Jr, George Konidaris, Leslie Kaelbling, and Tomas Lozano-Perez. Lqr-rrt*: Optimal sampling-based motion planning with automatically derived extension heuristics. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 2537–2542. IEEE, 2012.

[9] D. J. Webb and J. van den Berg. Kinodynamic RRT*: Optimal Motion Planning for Systems with Linear Differential Constraints. *ArXiv e-prints*, May 2012.