Robotics and Autonomous Systems 62 (2014) 214–228

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

A hierarchical approach for primitive-based motion planning and control of autonomous vehicles

David J. Grymin, Charles B. Neas, Mazen Farhood *

Department of Aerospace and Ocean Engineering, Virginia Tech, Blacksburg, VA 24061, United States

HIGHLIGHTS

- We examine graph-based search methods for motion planning using motion primitives.
- We develop a locally greedy algorithm and compare it to a version of Weighted A*.
- Greedy algorithm is advantageous when utilizing large motion primitive libraries.
- We develop a hybrid control technique for tracking concatenated motion primitives.
- The approach is applied to a hovercraft subject to disturbances and uncertainties.

ARTICLE INFO

Article history:
Received 4 August 2012
Received in revised form
29 June 2013
Accepted 3 October 2013
Available online 18 October 2013

Keywords:
Motion planning
Heuristic search
Hybrid control
Time-varying systems
Linear matrix inequalities

ABSTRACT

A hierarchical approach for motion planning and control of nonlinear systems operating in obstacle environments is presented. To reduce the computation time during the motion planning process, dynamically feasible trajectories are generated in real-time through concatenation of pre-specified motion primitives. The motion planning task is posed as a search over a directed graph, and the applicability of informed graph search techniques is investigated. Specifically, we develop a locally greedy algorithm with effective backtracking ability and compare this algorithm to weighted A* search. The greedy algorithm shows an advantage with respect to solution cost and computation time when larger motion primitive libraries that do not operate on a regular state lattice are utilized. Linearization of the nonlinear system equations about the motion primitive library results in a hybrid linear time-varying model, and an optimal control algorithm using the ℓ_2-induced norm as the performance measure is provided to ensure that the system tracks the desired trajectory. The ability of the resulting controller to closely track the trajectory obtained from the motion planner, despite various disturbances and uncertainties, is demonstrated through simulation.

1. Introduction

Motion planning for unmanned vehicles involves developing feasible trajectories through an obstacle field from a given initial state to a desired goal state; see, for instance, [1,2]. By using a discretized set of feasible motion primitives, the problem of finding a trajectory from the start to the goal becomes a graph search, a topic that has received a wealth of attention in the literature. This paper takes the approach of utilizing a set of pre-specified motion primitives, i.e. state and control histories defined over finite (or semi-infinite) time intervals, to generate, in real-time, collision-free trajectories from start to goal via graph search methods. As for the execution of the motion plan, the series of motion primitives generated by the planner will correspond to a sequence of pre-designed subcontrollers to be applied consecutively.

The notion of constructing a solution from available trajectories is a common approach for vehicle motion planning. Prior methods have used online optimization to determine the trajectory [3], concatenating trim and maneuver trajectories to form a dynamically feasible path from the start state to the goal. In certain scenarios, the solution of such an optimization problem may require more computational effort than can be allotted to the planning task. Deterministic and sampling-based searches over graphs are two broad categories that have received considerable attention related to robot and vehicle trajectory planning in obstacle environments; a comprehensive review of motion planning with respect to unmanned aerial vehicles is given in [4].

Deterministic graph search algorithms use knowledge obtained during the search as well as prior knowledge of the environment to work towards an optimal solution. A heuristic, or rule of thumb, assists in determining the order of expansion during the search. For vehicle motion planning problems, the cost-to-goal is a commonly chosen heuristic. The A* algorithm, a complete and optimal algorithm, uses the path cost to reach each node as well as the future path cost estimate from the heuristic, and traverses the graph by

* Corresponding author. Tel.: +1 540 231 2983; fax: +1 540 231 9632.
E-mail address: farhood@vt.edu (M. Farhood).

0921-8890/$ – see front matter
http://dx.doi.org/10.1016/j.robot.2013.10.003

expanding nodes with the lowest total path cost. For vehicle motion planning, the length of the path to reach a node can be used for path cost. In some applications, finding the optimal path may become burdensome, and a suboptimal solution is accepted to reduce the computational load. Weighted A* (WA*) relaxes optimality by weighting the heuristic in relation to the cost-to-go, effectively increasing the greediness of the algorithm, and is able to return solutions much faster with a bound on the suboptimal path cost [5,1]. Recent work related to A* based search methods has focused on iteratively improving suboptimal trajectories towards the minimum-cost path; see, for instance, the anytime search heuristic developed in [6,7]. Anytime search attempts to quickly return a feasible yet suboptimal path, and then improve upon this path successively in the time allotted for planning. In [7], for example, successive WA* searches are run with decreasing weight to achieve the best possible path in the given time for computation.

In sampling-based planners, such as the probabilistic roadmap (PRM) and rapidly-exploring random trees (RRTs), completeness is probabilistic; a solution will be returned, should one exist, with a probability converging to one as the number of samples tends towards infinity [8,9]. In practice, the RRT algorithm in particular is capable of returning a path to the goal fairly quickly, even in high-dimensional search spaces and subject to differential vehicle constraints. RRTs quickly examine unexplored regions of the state space, and are able to find paths through complicated obstacle fields with relative ease. The trade-off, however, is in the solution quality, as the path is often erratic due to the random sampling which drives the expansion. Additionally, Karaman and Frazzoli showed that the probability of the RRT algorithm converging to the optimal solution was zero. Their development of the RRT* algorithm, however, provides conditions for asymptotic optimality in addition to probabilistic completeness [10]. This algorithm has since been extended to an anytime framework, in which an initial solution is obtained quickly and then improved upon in the remaining time allotted [11].

The representation of the input and search spaces is also a factor in selecting the method to use. In [12], a discretized set of control inputs was used to compute a path for nonholonomic vehicles, with numerical integration performed during the planning process. Graph search was then utilized over a partitioning of the configuration space to determine a sequence of control inputs that brought the vehicle from its initial position to a goal region. A similar approach was taken by [13], with integration of control actions performed offline and stored for use with an online planner; solutions were obtained by performing a search over a tree. Precomputed vehicle motions can also be developed that result in a grid-based representation of the configuration space, referred to as a state lattice. In this framework, the state lattice is represented as a directed graph, with vertices corresponding to specific reachable states of the vehicle and edges indicating the dynamically feasible motions which connect the states exactly. Such a representation is resolution complete, i.e. it is complete with respect to the resolution at which the lattice is generated [14,15].

It is important to note that when using pre-computed control input and state histories, the ability of the vehicle to track the resulting motion plan is subject to model accuracy. Unmodeled dynamics, parametric uncertainty, and exogenous disturbances may result in deviations from the original motion plan during execution. In the work of Burridge et al., a sequence of pre-computed feedback controllers is used to bring the system to a desired goal state in the presence of disturbances and obstacles in the robot workspace [16]. This framework has also been used for motion planning using controllers valid over regions of the free space; the vehicle is guided to the goal region by the sequence of controllers, with no path explicitly determined [17,18].

The approach in this paper utilizes a distinct set of motion primitives and entails performing a graph search to find an appropriate dynamically feasible trajectory through an obstacle environment. The set of motion primitives, hereafter referred to as a library, is developed offline. State and control histories for each motion primitive can be obtained through a variety of methods, for instance, by solving an optimization problem involving the nonlinear system equations or by recording human operator control inputs. The task of the motion planner is to then concatenate available motion primitives to find a trajectory from the initial state to the goal. This approach eliminates the need to solve for dynamically feasible state and control histories online. As far as the motion planner is concerned, any dynamically feasible motion primitive can be incorporated into the library. But, since these primitives can be generated experimentally, it is important that the primitive be within the state-space envelope where the derived mathematical model constitutes a reasonably accurate description of the vehicle dynamics. This requirement is imposed because the proposed control approach is model-based, as discussed later. We examine in this paper graph-based search techniques for motion planning, where the graph does not represent a state lattice but rather exhibits a tree structure, and the edges of the graph correspond to pre-specified motion primitives. Specifically, we develop a locally greedy algorithm with effective backtracking ability and compare it to a version of weighted A* based on a tree search. Both algorithms are applied in simulation to a hovercraft system and evaluated in environments composed of known, randomly constructed static obstacle fields. The greedy algorithm shows an advantage with respect to solution cost and computation time when relatively large motion primitive libraries with multiple velocities are utilized.

In addition, the paper provides a hybrid control approach, with the ℓ_2-induced norm as the performance measure, to ensure that the system tracks the desired trajectory generated by the motion planning algorithm despite various disturbances and uncertainties. The hybrid systems of interest in this paper are composed of linear time-varying (LTV) subsystems obtained from linearizing the nonlinear system equations describing the vehicle dynamics about the library of pre-specified primitives. The switching between these subsystems and ultimately their corresponding subcontrollers is dictated by the motion planning algorithm. The synthesis solution is provided in terms of a semidefinite program, and is based on the results of [19,20]. Related to this work is the paper [21] which provides a hybrid dynamics framework for the design of guaranteed safe switching regions using reachable sets. The paper [22] also gives a control algorithm for maneuver-based motion planning, which is robust to a certain class of perturbations.

The paper is organized as follows. Section 2 presents the deterministic search methods used in this paper. Section 3 provides a control result for hybrid LTV systems. Section 4 gives a detailed implementation of the motion planning and control methods on a hovercraft system. This paper serves as an extension of the results presented in the conference paper [23], and provides additional details regarding the implementation of the methodology. The intent of this paper, borrowing terminology from [4], is to provide a framework for *sound* motion planning, where the devised plan guarantees a collision-free trajectory despite possible disturbances, measurement errors, and other uncertainties.

The notation is mostly standard. We denote the set of real $n \times m$ matrices by $\mathbb{R}^{n\times m}$. The adjoint of an operator X is written X^*, and we use $X \prec 0$ to mean it is negative definite. The normed space of square summable vector-valued sequences is denoted by ℓ_2. It consists of elements $x = (x_0, x_1, x_2, \ldots)$, with each $x_k \in \mathbb{R}^{n_k}$ for some n_k, having a finite 2-norm $\|x\|_{\ell_2}$ defined by $\|x\|_{\ell_2}^2 = \sum_{k=0}^{\infty} |x_k|^2 < \infty$, where $|x_k|^2 = x_k^* x_k$.