2010

A Pseudospectral Optimal Motion Planner for Autonomous Unmanned Vehicles

Hurni, Michael A.

2010 American Control Conference
Marriott Waterfront, Baltimore, MD, USA
June 30-July 02, 2010

A Pseudospectral Optimal Motion Planner for Autonomous Unmanned Vehicles

Michael A. Hurni, Pooya Sekhavat, Mark Karpenko, and I. Michael Ross

Abstract— **This paper presents a pseudospectral (PS) optimal control algorithm for the autonomous motion planning of a fleet of unmanned ground vehicles (UGVs). The UGVs must traverse an obstacle-cluttered environment while maintaining robustness against possible collisions. The generality of the algorithm comes from a binary logic that modifies the cost function for various motion planning modes. Typical scenarios including path following and multi-vehicle pursuit are demonstrated. The proposed framework enables the availability of real-time information to be exploited by real-time reformulation of the optimal control problem combined with real-time computation. This allows the each vehicle to accommodate potential changes in the mission/environment and uncertain conditions. Experimental results are presented to substantiate the utility of the approach on a typical planning scenario.**

I. INTRODUCTION

Various methods have been proposed and examined for autonomous guidance and control of unmanned vehicle systems [1], [2]. These methods include road maps, cell decompositions, artificial potential fields, and optimal control schemes. A variety of obstacle avoidance algorithms are also proposed in the literature including bug algorithms, vector field histograms, the bubble band technique, the curvature velocity techniques, the Schlegel approach, and the ASL method. Bug algorithms [3], [4] represent an example of obstacle avoidance algorithms. They do not provide the control necessary to traverse the planned trajectory; in most cases they do not provide an optimal solution and they often fail in dynamic obstacle avoidance. Bug algorithms do not account for vehicle dynamics or constraints and operate based on local information only. Artificial potential field methods [5], [6] do not provide the control necessary to traverse the planned trajectory with a measure of optimality. Vehicle constraints are also not accounted for without the addition of other techniques. Roadmaps [7], [8] also do not directly compute the control for the planned trajectory. They are computationally expensive, and thus are not suitable for real-time applications. Probabilistic roadmaps can be implemented in real-time, but sacrifice optimality for the faster run times. Cell decomposition [9], [10] methods are computationally complex when using a sufficient resolution to obtain optimality. This makes dynamic obstacle avoidance difficult, and limits their ability to be updated frequently enough to be able to be used in real-time and receive and incorporate local information updates.

Multi-vehicle trajectory planning has been associated with many descriptive terms and categories such as centralized, decentralized, decoupled, coordinated, cooperative and/or prioritized motion. Centralized planning involves utilizing one state space vector that includes all the vehicle states. Decentralized (also referred to as decoupled) planning involves path planning of each vehicle individually, thus reducing the complexity and size of the state space. Coordinated and cooperative planning are interchangeable and refer to the amount of information each vehicle has about every other vehicle in decoupled motion. Of course, centralized planning could also be called fully cooperative, since the vehicles are in one state space and operate synergistically. Prioritization can be inserted in decoupled planning and refers to the priority of certain vehicles over others.

Methods based on artificial potential fields have been extended to decoupled multi-vehicle planning, control and prioritization [11]–[13]. Roadmap theory has also been used to tackle multi-vehicle scenarios. In [14], optimal motion planning is presented for multiple robots by defining a state space that simultaneously represents the configurations of all of the robots. The SBL planner [15] has been proven to be more reliable at centralized planning than decoupled planning. Approximate cell decomposition has also been used for multi-vehicle path planning. The complexity of the problem is reduced by means of a hierarchical multilevel discretization using a simple navigation function [16].

Trajectory planning using numerical optimal control techniques is a direct approach that solves the complete motion-planning problem. The method determines the vehicle trajectory to the target by searching within the vehicle's state space. The planner requires the kinodynamic equations of the vehicle, the obstacles formulated in the form of path constraint functions, and an appropriate cost function. The cost function can be any function of state variables, control variables and time, as long as it is sufficiently smooth (i.e. continuous and differentiable). The kinodynamic equations can also be viewed as constraints (like the obstacles), defining the relationship between the vehicle states and the control inputs. Even though the exact obstacle models may not be smooth functions, the constraints used to formulate them within the optimal control framework can be chosen as

M. A. Hurni is with the Department of Weapons and Systems Engineering, United States Naval Academy, Annapolis, MD, 21402, USA hurni@usna.edu

P. Sekhavat is with the Department of Aerospace Engineering, Texas A&M University, College Station, TX, 77843, USA psekhava@nps.edu

M. Karpenko is with the Department of Mechanical and Astronautical Engineering, Naval Postgraduate School, Monterey, CA 93943 USA mkarpenk@nps.edu

I. M. Ross is with the Department of Mechanical and Astronautical Engineering, Naval Postgraduate School, Monterey, CA 93943 USA imross@nps.edu

978-1-4244-7427-1/10/$26.00 ©2010 AACC

smooth approximate functions. By defining the problem in optimal control framework, we can find a solution from initial to final conditions for each vehicle, while avoiding obstacles and obeying vehicle state/control limits [17].

In this paper, we present a general Pseudospectral optimal control-based algorithm for autonomous motion planning and control of unmanned ground vehicles. Pseudospectral methods [17]–[21] are a family of computational methods that can be characterized according to the underlying orthogonal polynomials (e.g., Legendre polynomials, Chebyshev polynomials, etc.) and types of quadrature nodes (e.g., Gauss-Lobatto, Gauss-Radau or pure Gauss) [17], [22]. The basic idea in PS methods is to discretize the continuous-time optimal control problem into a sequence of discrete-time optimization problem using PS methods. The resulting sequence of optimization problems can then be solved by fast spectral algorithms [17], [23]. As will be apparent shortly, the motion planning framework requires the use of arbitrary boundary conditions; hence, we choose Gauss-Lobatto nodes as other grid points may fail to converge [24].

The main objective of this paper is to illustrate how a wide variety of single and multi-vehicle trajectory planning problems with varying levels of available information can be solved through adjustment of the optimal control problem formulation. The ability of the proposed framework to exploit the availability of real-time information to allow control solutions to be repeatedly recomputed and updated throughout the mission is described. This enables the trajectory planning algorithm to accommodate changing and uncertain environmental conditions. Experimental results are presented to demonstrate the utility of the approach on a laboratory scale UGV in a typical planning scenario.

II. PROBLEM STATEMENT

A. Optimal Control Formulation

The objective of the trajectory planning problem is to find the state-control function pair, $t \mapsto (x,u) \in \mathbb{R}^{N_x} \times \mathbb{R}^{N_u}$ that enables the vehicle to maneuver from initial state, x_o, at time t_o to target state, x_f, at final time, t_f. Feasible state-control function pairs are those that satisfy the boundary conditions, the limits on the vehicle states and controls, and the proximity constraints for collision avoidance. The optimal state-control function pair is the feasible trajectory that minimizes the mission specific cost function, $J[\cdot]$. The general formulation of the optimal control problem for autonomous trajectory planning of unmanned vehicles is given by [25]:

$$\left\{\begin{array}{lll} \text{Minimize} & J[\cdot] = & E(x_o,x_f,t_o,t_f)+ \int_{t_o}^{t_f} F(x(t),u(t),t)\ dt \\ \text{Subject to} & \dot{x}(t) = & f(x(t),u(t),t) \\ & x^L \le & x(t) \le x^U \\ & u^L \le & u(t) \le u^U \\ & h^L \le & h(x(t),u(t),t) \le h^U \\ & e^L \le & e(x_o,x_f,t_o,t_f) \le e^U \end{array}\right. \tag{1}$$

In (1), $\dot{x}(t) = f(x,u,t)$ represents the vehicle dynamics, subject to state and control constraints. The path constraints, $h(x,u,t)$, ensure collision-free trajectories and $e(x_o,x_f,t_o,t_f)$ enforce the desired initial and final conditions. Using the above framework, multi-vehicle missions can be handled in either a centralized or decentralized fashion. For centralized planning, the vehicle dynamic and constraint equations are expanded to include all vehicles. In the decentralized approach, an optimal control problem is solved for each vehicle, taking into account the available information about the others. In the following sections we illustrate how a wide variety of single and multi-vehicle trajectory planning scenarios can be solved using the above formulation.

B. Vehicle Model

A schematic of the UGV for mathematical modeling is shown in Fig. 1. The UGV has a front wheel steering arrangement, which introduces nonholonomic constraints to the system. The state equations describing the motion of the UGV are as follows:

$$\dot{x} = \frac{d}{dt}\begin{bmatrix} x \\ y \\ \theta \\ v \\ \gamma \end{bmatrix} = \begin{bmatrix} v\cos(\theta) \\ v\sin(\theta) \\ \frac{v}{L}\tan(\gamma) \\ a \\ \omega \end{bmatrix} \tag{2}$$

where variables, x, y, and θ give the position and orientation of the vehicle, measured with respect to the center of the rear axle. The vehicle speed is given by v, L is the vehicle wheelbase and γ is the steering angle. The control variables are $u = [a,\omega]^T$, where a is the vehicle acceleration and ω is the steering rate.

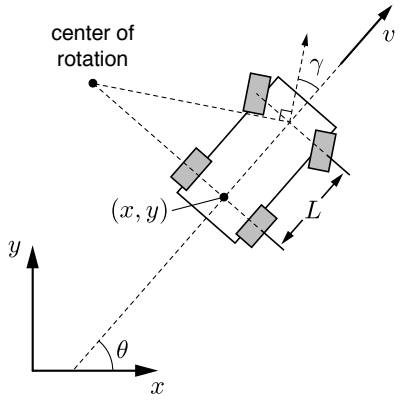


Fig. 1. Schematic of nonholonomic autonomous vehicle.

C. Obstacle Model

Obstacles are modeled using continuous algebraic functions of the form

$$d = \left(\frac{x-x_c}{a+v_s}\right)^p + \left(\frac{y-y_c}{b+v_s}\right)^p - 1 \tag{3}$$

where x_c and y_c indicate the location of the geometric center of the obstacle, and d is the distance between a point (x, y) and the boundary of the obstacle. By varying the value of parameter, p, equation (3) can be used to create a number of generic shapes that represent different types of obstacles. The values of a and b are chosen to define the obstacle dimensions and v_s represents the radius of a circle circumscribing the vehicle. Equation (3) can be used directly to check for collisions between the vehicle location, (x, y), and obstacles in the environment. Potential collisions between two vehicles are modeled by setting $a = b = v_s$ and $p = 2$.

An issue that must be considered when using (3) within a computational optimal control framework is the fact that the equations describing the problem formulation should be well-scaled for numerical solution. Since the value of d in (3) can become large when the vehicle is far from the obstacle, the natural logarithm of (3) is used to formulate the path constraints:

$$h_{ij} = \ln\left[\left(\frac{x_j(t) - x_{c_i}(t)}{a_i(t) + v_{s_j}}\right)^{p_i} + \left(\frac{y_j(t) - y_{c_i}(t)}{b_i(t) + v_{s_j}}\right)^{p_i}\right] \quad (4)$$

In (4), $i = 1, \ldots, m + n$ and $j = 1, \ldots, n$ are indices denoting the possible collisions between the m obstacles and n vehicles in the environment. By allowing the values of variables, x_{c_i} , y_{c_i}, a_i, and b_i in the right hand side of (4) to be time dependent, it is possible to model moving obstacles, constraints imposed by the motion of other vehicles, as well as obstacles with time-varying dimensions.

D. Pseudospectral Method

A pseudospectral (PS) method solves optimal control problems by approximating the vehicle state trajectories by N-th order Lagrange interpolating polynomials. The interpolating polynomials are evaluated at the Legendre-Gauss-Lobatto (LGL) nodes, $\tau \in [-1, 1]$. The non-uniformily distributed nodes are dense near the end points, which effectively inhibits the Runge phenomena, and is a necessary condition for convergence of the discretized solution to the continuous-time solution. It is worth noting that pure Gauss points, although dense near the end points, do not converge to the continuous-time problem [22], [24]; hence, we use Gauss-Lobatto points.

In the PS discretization, the time history of the approximate state trajectory is given by

$$x(t) \approx x^N(t) = \sum_{l=0}^{N} x_l \phi_l(\tau) \quad (5)$$

where x_l is the value of the approximant at node t_l and $\phi_l(\tau)$ are the Lagrange interpolating polynomials of order N. The time-derivatives of the state trajectories are obtained straightforwardly from the following equation

$$\dot{x}(t) \approx \dot{x}^N(t_k) = \sum_{l=0}^{N} \mathbf{D}_{kl} x_l \quad (6)$$

where $\mathbf{D}$ is a $N + 1 \times N + 1$ differentiation matrix with constant elements:

$$\mathbf{D}_{kl} = \frac{2}{t_f - t_0} \begin{cases} \frac{L_N(\tau_k)}{L_N(\tau_l)} \frac{1}{\tau_k - \tau_l} & \text{if } k \neq l; \\ -\frac{N(N+1)}{4} & \text{if } k = l = 0; \\ \frac{N(N+1)}{4} & \text{if } k = l = N; \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

In (7), L_N is the N-th order Legendre polynomial. The discretized controls satisfying the vehicle dynamics can be computed by ensuring that $\sum_{l=0}^{N} \mathbf{D}_{kl} x_l - \mathbf{f}(x_k, u_k, t) = 0$ for $k = 0, 1, \ldots, N$.

The cost function is evaluated by applying the Gauss-Lobatto integration rule

$$J[\cdot] = E(x_0, x_N, t_o, t_f) + \frac{t_f - t_o}{2} \sum_{k=0}^{N} w_k F(x_k, u_k) \quad (8)$$

where w_k are the LGL weights given by

$$w_k = \frac{2}{N(N+1)[L_N(\tau_k)]^2}, \quad k = 0, 1, \ldots, N \quad (9)$$

Using the above relations, the PS method transforms the continuous-time optimal control problem to sequence of discrete-time problems, which can be solved by fast spectral algorithms. Additional details on the PS method and the spectral algorithm can be found in [21]–[23].

III. The Pseudospectral Trajectory Planning Algorithm

A block diagram of the trajectory planning algorithm is given in Fig. 2. The figure illustrates how the initial open-loop optimal control problem is formulated, and how updates can be incorporated via feedback in order to enable the vehicles to operate autonomously in real-time. The formulation of the optimal control problem is first initialized based on the mission requirements and known information about the environment map. The *Initialize Problem Formulation* block also configures the general cost function:

$$J[\cdot] = k_T t_f + \int_{t_o}^{t_f} \left[k_R \mathcal{R}(t) + k_P \mathcal{P}(t) + k_S \mathcal{S}(t) + k_G \mathcal{G}(t)\right] dt \quad (10)$$

By changing the state of the logic switches, k, it is possible to configure the cost function for different kinds of trajectory planning missions. For example, if it is desired to minimize the maneuver time, then $k_T = 1$. If maneuver time is not important then k_T is set to zero. By initializing the remaining switches, path following, formation keeping or sector keeping modes can be selected. It is also possible to choose whether to treat multi-vehicle systems in a centralized or decentralized manner.

Unlike centralized planning, a decentralized planning scenario relies heavily on the type of information that is available for processing by each of the vehicles as they execute the planning algorithm. Three levels of information about the static/dynamic environment have been studied within the context of the proposed PS trajectory planning algorithm: instantaneous (snapshots), prediction (course and

TABLE I

SUMMARY OF COST FUNCTION LOGIC.

Motion Planning Mode	k_R	k_G	k_P	k_S	k_T
Centralized Multi-Vehicle	1	1	0	0	1
Decentralized Multi-Vehicle	1	0	0	0	1
Path Following	0	0	1	0	1
Formation Keeping	1	0	1	0	0
Sector Keeping	1	0	0	1	0

speed), and complete *a priori* knowledge of the environment dynamics [26].

If only instantaneous information about the surroundings and other vehicles is available, the planner assumes a static environment while computing each vehicle trajectory update. However, since the trajectory is continuously recomputed as the vehicle executes the motion, it is still possible to successfully navigate amongst other vehicles and moving obstacles without collision. A vehicle with prediction capability uses estimates of moving obstacle and/or other vehicles' course and speed to extrapolate their positions over the trajectory planning horizon. A vehicle with *a priori* knowledge of the environment has the complete knowledge of future changes in the course and speed of all other vehicles as well as all static/dynamic obstacles. Each of these three information levels can be exploited by the motion planner for multi-vehicle missions. A summary of the possible motion planning modes that can be configured by setting the cost function switches is given in Table I.

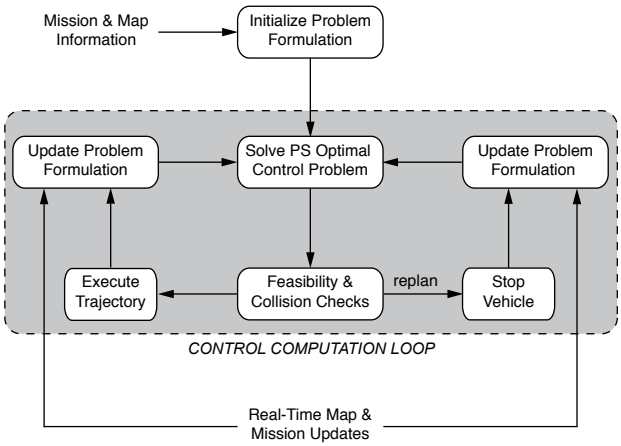


Fig. 2. Block diagram of trajectory planning algorithm.

Next, we explain the various elements comprising the running cost that enable the different trajectory planning modes to be realized. The first element of the running cost, $\mathcal{R}(t)$, ensures robustness of the vehicle collision avoidance against model uncertainty and control interpolation errors that could otherwise cause the vehicle to collide with an obstacle, or another vehicle. The robustness cost is defined as

$$\mathcal{R}(t) = \sum_{j=1}^{n} \sum_{i=1}^{n+m} w_{r_{ij}} (e^{e^{-h_{ij}}} - 1) \tag{11}$$

where $w_{r_{ij}}$ are weights that are incremented to increase the maneuver robustness when the trajectory of a vehicle passes in close proximity to an obstacle or another vehicle. The precise manner in which the weights are chosen depends upon the type of trajectory planning problem being solved and is explained later in this section. The value of p_i used to compute h_{ij} (see equation 4) is limited to be less than 4 in order to ensure that the value of $\mathcal{R}(t)$ is large enough to enable the vehicle to navigate around obstacle corners without collision. Fig. 3 shows how adding the robustness cost influences the vehicle motion when navigating around an obstacle.

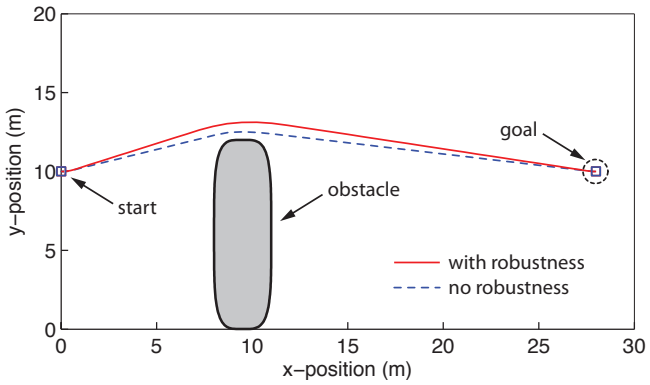


Fig. 3. The influence of robustness cost on obstacle avoidance ($w_r = 1/7$).

The next component in running cost, $\mathcal{P}(t)$, is used for path following and leader-follower formation keeping in multi-vehicle teams. Here, each vehicle must track a predefined path without colliding with other vehicles or other obstacles. Path following is accomplished by introducing the running cost:

$$\mathcal{P}(t) = \sum_{j=1}^{n} \tan^{-1} \left[(x_j(t) - x_j^d(t))^2 + (y_j(t) - y_j^d(t))^2 \right] \tag{12}$$

The use of the transcendental function ensures that the cost of deviating from the specified path increases smoothly as the path following error increases. The interplay between the robustness cost, $\mathcal{R}(t)$, and the path following cost, $\mathcal{P}(t)$, allows the vehicles to successfully avoid obstacles while tracking predefined trajectories as closely as possible. This is achieved by monitoring the value of the robustness function and toggling between the robustness and path following costs. For example, if $\mathcal{R}(t)$ exceeds a threshold (e.g. in the event of an unforeseen pop-up obstacle along the prescribed path), the algorithm can automatically switch from path following mode to obstacle avoidance mode by setting k_P to zero and k_R to one for robust collision avoidance. A path-following trajectory for a single vehicle is shown in Fig. 4. Without $\mathcal{P}(t)$ active in the cost function, the vehicle traverses the path denoted by the dashed line. However,

including $\mathcal{P}(t)$ as part of the cost cost will drive the vehicle along the predefined path until the vehicle encounters the unanticipated obstacle in its path. As the vehicle approaches the obstacle the value of $\mathcal{R}(t)$ continues to increase until the motion planner switches to obstacle avoidance mode ($k_R = 1, k_P = 0$). After successfully avoiding the obstacle, the value of $\mathcal{R}(t)$ decreases below the switching threshold and the path planning cost is reactivated ($k_R = 0, k_P = 1$) for the remainder of the maneuver.

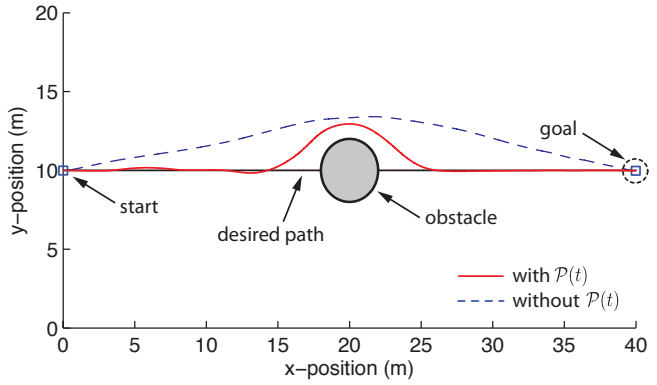


Fig. 4. Path-following with obstacle avoidance illustrating the effect of toggling between $\mathcal{R}(t)$ and $\mathcal{P}(t)$.

The objective of sector keeping is to maintain the position of one or more vehicles within a predefined region relative to a reference point. A typical sector keeping example is a pursuit maneuver where each pursuing vehicle should track a moving target while maintaining a certain buffer distance from the target. In the proposed framework, this type of sector keeping mission is achieved by invoking the cost function, $\mathcal{S}(t)$:

$$\mathcal{S}(t) = \sum_{j=1}^{n} s_j(t)$$

$$s_j(t) = \begin{cases} -\ln\left[\left(\dfrac{x_j - x_t}{d_{\min}}\right)^2 + \left(\dfrac{y_j - y_t}{d_{\min}}\right)^2\right] & ; d \le d_{\min} \\ 0 & ; d_{\min} < d < d_{\max} \\ \ln\left[\left(\dfrac{x_j - x_t)}{d_{\max}}\right)^2 + \left(\dfrac{y_j - y_t}{d_{\max}}\right)^2\right] & ; d \ge d_{\max} \end{cases} \tag{13}$$

In (13), x_t and y_t denote the location of the moving target and the values of $d_{\min}$ and $d_{\max}$, which define the desired sector, are dictated by the mission requirements. Fig. 5 shows an example of using the cost function (13) in a pursuit scenario. The target vehicle travels along the random path (red line) amongst eight obstacles, while three pursuit vehicles maintain their position within the desired sector (dotted rings) throughout the maneuver.

The final component making up the running cost is the distance-to-goal function, $\mathcal{G}(t)$. This component of the cost is used exclusively for centralized multi-vehicle motion planning. The objective here is to ensure that vehicles that are able to move to their final positions more quickly than others

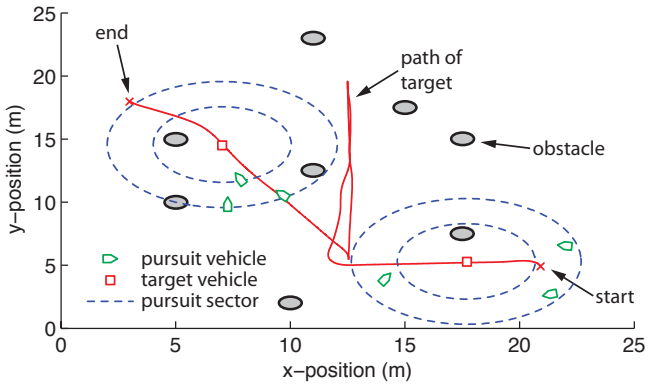


Fig. 5. An example multi-vehicle pursuit scenario.

do not 'wander' while waiting for slower vehicles to reach their goals (see Fig. 6a). The cost $\mathcal{G}(t)$ is simply a function of each vehicles current distance from the desired goal:

$$\mathcal{G}(t) = \sum_{j=1}^{n} \ln\left[\left(\frac{x_j(t) - x_j^d(t)}{\epsilon}\right)^2 + \left(\frac{y_j(t) - y_j^d(t)}{\epsilon}\right)^2\right] \tag{14}$$

where ϵ is a user-defined small parameter. The distance-to-goal cost adds to the overall cost when a vehicle has not reached its goal position. Therefore, the cost is minimized when all vehicles reach their goals in minimum time. The same multi-vehicle maneuver shown in Fig. 6a is repeated in Fig. 6b with $\mathcal{G}(t)$ added to the cost function. In Fig. 6a each vehicle took 32-sec to complete its maneuver. With the addition of running cost, $\mathcal{G}(t)$, in Fig. 6b, the individual maneuver times varied from 22 sec (fastest vehicle) to 32 sec (slowest vehicle) [26].

Once the motion planning problem is formulated according to (1) and initialized by invoking the appropriate elements of the cost function, it is repeatedly solved using PS tools. This occurs in the *Solve Problem* block of Fig. 2. The number of LGL nodes used to discretize the problem at each iteration is determined based on the overall distance the vehicle must travel, as well as the scale of the obstacles in the environment. In this paper, the PS-based software package DIDO [27] is used to rapidly generate the extremals. The advantage of using PS optimal control algorithms lies in the fact that solutions can be obtained in a robust and rapid fashion. Typical computation rates for the motion planning problems shown in Figs. 3 to 6 range between 0.5 and 2 Hz. Therefore, by including a mechanism that allows the problem formulation to be continually updated, the benefits of feedback can be exploited to adapt to changes in environmental conditions such as dynamic and pop-up obstacles.

After solving the optimal control problem, the solution trajectory is validated in the *Feasibility and Collision Checks* block of Fig. 2. The first check is a feasibility check to ensure that the numerically computed control profile for each vehicle generates the desired optimal state trajectory (DIDO output), within a tolerance, when interpolated and propagated

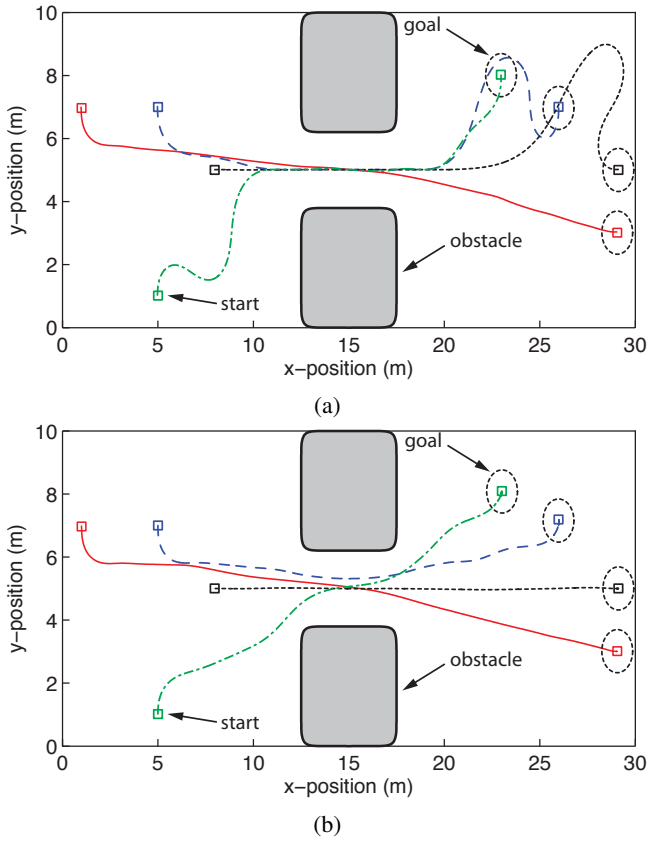


Fig. 6. Centralized multi-vehicle planning: (a) without distance-to-goal cost; (b) with distance-to-goal cost.

through the vehicle kinematics. If the feasibility check fails, the solution to the PS discretized problem has not converged to the solution of the corresponding continuos problem. In this case, the problem is reformulated by increasing the number of LGL discretization nodes and then resolved. The number of additional nodes required is generally small owing to the rapid convergence properties of the PS discretization [28].

The second set of verification checks are proximity checks. These checks are carried out by first constructing a proximity matrix, $\mathbf{H}_j$, for each vehicle ($j = 1, \ldots, n$). The elements of each matrix, h_{jk}, are the distances, computed using (3), between the vehicle and other obstacles/vehicles, at each point on a refined mesh. Therefore, each proximity matrix has dimensions $(n+m) \times N^r$, where N^r is the number of points on the refined mesh. Two types of proximity checks are performed, (*i*) collisions with a single object, and (*ii*) simultaneous collisions with multiple objects. The former check establishes the degree to which the vehicles can maneuver without colliding with objects distributed in the environment. The latter check is used to determine whether the solution needs to be refined to reduce the potential of collision when maneuvering amongst closely spaced obstacles. The result of each proximity check determines which *Update Problem Formulation* block will be followed in Fig. 2. A replan is required when any element of the proximity matrix indicates a direct collision between a vehicle and an obstacle. The distance between the vehicle and other vehicles/obstacles at any time instant on each vehicle's trajectory is also calculated to check if the vehicle fails to navigate any narrow passage. In either case, a new bias is selected to reinitialize the optimization solver and the problem is resolved.

A second set of proximity checks is performed next in order to avoid collisions arising from model uncertainty and control interpolation errors. However, in these checks, the proximity is evaluated only over a finite time horizon determined based on the mission. Thus, the algorithm can react to proximity events only when the vehicle is operating near obstacles.

If the vehicle is operating near a single obstacle, the robustness cost is modified by incrementing the weight, $w_{r_{ij}}$, according to the following relation

$$w_{r_{ij}} = w_{r_{ij}} + K(1 - e^{p_i/8}) \tag{15}$$

where $j = 1, \ldots, m+n$ and $j = 1, \ldots, n$ and K is selected based on how aggressive the vehicle trajectory should be refined by the robustness cost, $\mathcal{R}(t)$. If the vehicle is navigating near an obstacle or through a passage between two closely-spaced obstacles, model uncertainty and control interpolation errors are addressed simply by increasing the number of nodes, N, used in the PS discretization.

IV. Experimental Implementation

To demonstrate the practical utility of the algorithm, a trajectory planning test was conducted on a laboratory scale unmanned vehicle testbed. The experimental test environment and the autonomous UGV are shown in Fig. 7. The UGV platform (see Fig. 7a) has a 4-wheel drive chassis with a front wheel steering arrangement. Thus, the vehicle kinematics can be modeled directly using equation (2). The vehicle is retrofitted with Parallax HB-25 motor controllers driven by a Robostix micro-controller board. An XBee-PRO wireless RF module is used to facilitate communications between the vehicle and a ground-based command and control center. The Robostix board is also used to interface a number of infrared proximity sensors that can be used to detect collisions between the vehicle and obstacles in the environment. The position of the vehicle is measured using an optical motion capture (MoCap) measurement system. The MoCap system (see Fig. 7b) determines the three-dimensional positions of retroreflective markers affixed to the vehicle from a series of two-dimensional images acquired from CCD cameras distributed throughout the experimental test cell. The MoCap system provides position data at a rate of 120 Hz with sub-millimeter accuracy making it ideal for recording the motion of the experimental UGV.

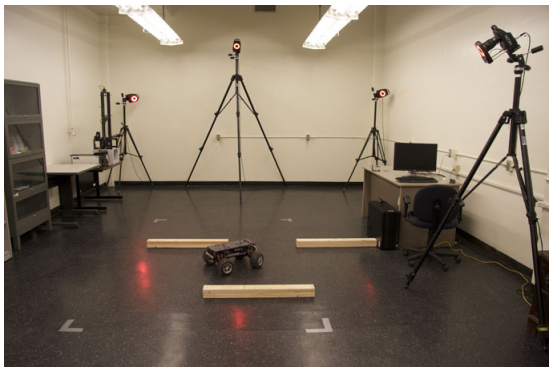
The results of the experimental test are shown in Fig. 8. The objective was to compute and execute a minimum time maneuver between the start and goal positions shown, while avoiding collisions with several obstacles present in the workspace. In this scenario, it is assumed that a map of the environment is known *a priori* and that the obstacles remain stationary throughout the maneuver. Therefore, the proposed

trajectory planning algorithm could be implemented in open-loop mode and problem updates are not incorporated in the maneuver. Constraints on the acceleration, maximum velocity and steering angle of the experimental vehicle were included in the problem formulation to ensure that the numerical solution is physically realizable.

The resulting collision-free maneuver was implemented by streaming the computed vehicle speed and steering angle commands from the ground terminal to the UGV over the XBee network. The overall maneuver time was about 12.5 seconds and the UGV reached speeds of nearly 1 m/sec during the test. Referring to Fig. 8, it is observed that the experimentally executed trajectory closely follows the computed optimal trajectory. The slight discrepancies between the two paths can be attributed to the error introduced due to imperfect tracking of the vehicle commands. Nonetheless, the experimental results confirm the feasibility of the numerical solution and demonstrate the functionality of the trajectory planning algorithm on a real vehicle.

A more comprehensive suite of experiments is currently being conducted to further illustrate the efficacy of the developed motion planning algorithm for other types of motion planning problems.

(a)



(b)

Fig. 7. Experimental setup: (a) autonomous unmanned ground vehicle; (b) indoor test environment showing optical motion capture system.

V. Conclusions

It is possible to solve a large variety of motion planning problems using a single optimal control framework. The key to this generality is the unification of different optimal control problem formulations solved in real-time using real-time information updates. The real-time computation is made possible through the use of convergent PS optimal control techniques. The Legendre PS method was chosen as it is the only proven PS method for convergence and consistency. The utility of the approach was demonstrated by way of a typical trajectory planning experiment in which a UGV must navigate amongst several stationary obstacles.

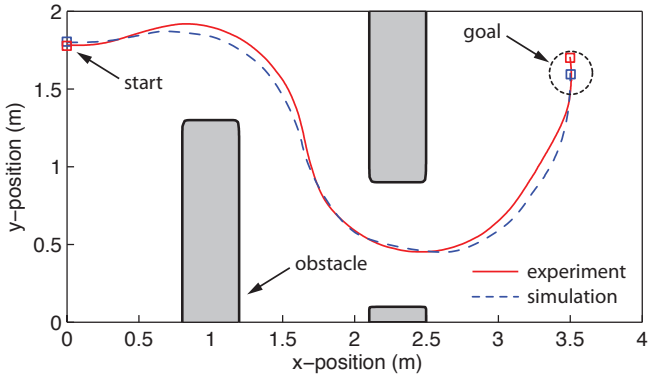


Fig. 8. Experimental obstacle avoidance maneuver.

References

[1] Siegwart, R. and Nourbakhsh, I.R., "Introduction to Autonomous Mobile Robots," *The MIT Press*, 2004.

[2] Choset, H., Lynch, K.M., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L.E., and Thrun, S., "Principles of Robot Motion; Theory, Algorithms, and Implementation," *The MIT Press*, 2005.

[3] Langar, R.A., Coelho, L.S., and Oliveira, G.H., "K-Bug, A New Bug Approach for Mobile Robot's Path Planning," *IEEE International Conference on Control Applications*, 2007.

[4] Magid, E. and Rivlin, E., "CautiousBug: A Competitive Algorithm for Sensory-Based Robot Navigation," *Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2004.

[5] Vadakkepat, P., Tan, K.C., and Ming-Liang, W., "Evolutionary Artificial Potential Fields and Their Application in Real Time Robot Path Planning," *Proceedings of the 2000 Congress on Evolutionary Computation*, Vol. 1, pp. 256-263, 2000.

[6] Urakubo, T., Okuma, K., and Tado, Y., "Feedback Control of a Two Wheeled Mobile Robot with Obstacle Avoidance Using Potential Functions," *Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2004.

[7] Li, T.Y. and Shie, Y.C., "An Incremental Learning Approach to Motion Planning with Roadmap Management," *IEEE International conference on Robotics and Automation*, 2002.

[8] Song, G., Miller, S., and Amato, N.M., "Customizing PRM Roadmaps at Query Time," *International Conference on Robotics and Automation*, 2001.

[9] Cowlagi, R.V. and Tsiotras, P., "Beyond Quadtrees: Cell Decompositions for Path Planning Using Wavelet Transforms," *Proceedings of the 46th IEEE Conference on Decision and Control*, 2007.

[10] Zhu, D. and Latombe, J.C., "New Heurisic Algorithms for Efficient Hierarchical Path Planning," *IEEE Transactions on Robotics and Automation*, Vol. 7, No. 1, 1991.

[11] Warren, C.W., "Multiple Robot Path Coordination Using Artificial Potential Fields," *Proceedings of the 1990 IEEE International Conference on Robotics and Automation*, vol. 1, pp. 500-505, 1990.

[12] Baras, J.S., Tan, X., and Hovareshti, P.,"Decentralized Control of Autonomous Vehicles," *Proceedings of the 42nd IEEE Conference on Decision and Control*, 2003.

[13] Zheng, T., and Zhao, X.,"A Novel Approach for Multiple Mobile Robot Path Planning in Dynamic Unknown Environment," *IEEE Conference on Robotics, Automation and Mechatronics*, 2006.

[14] Lavalle, S.M. and Hutchinson, S.A.,"Optimal Motion Planning for Multiple Robots Having Independent Goals," *IEEE Transactions on Robotics and Automation*, vol. 14, No. 6, 1998.

[15] Sanchez, G. and Latombe, J.C.,"Using a PRM Planner to Compare Centralized and Decoupled Planning for Multi-Robot Systems," *Proceedings of IEEE International conference on Robotics & Automation*, 2002.

[16] Conte, G. and Zulli, R. "Hierarchical Path Planning in a Multi-Robot Environment with a Simple Navigation Function," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 25, No. 4, 1995.

[17] Ross, I.M. and Fahroo, F., "Issues in the Real-Time Computation of Optimal Control," *Mathematical and Computer Modelling*, Volume 43, Issues 9-10, pp. 1172-1188, 2006.

[18] Bollino, K.P., Lewis, L.R., Sekhavat, P., and Ross, I.M.,"Pseudospectral Optimal Control: A Clear Road for Autonomous Intelligent Path Planning," *AIAA Infotech at Aerospace 2007 Conference and Exhibit*, 2007.

[19] Gong, Q., Kang, W., Bedrossian, N., Fahroo, F., Sekhavat, P., and Bollino, K.P., "Pseudospectral Optimal Control for Military and Industrial Applications," *Proceedings of the 46th IEEE Conference on Decision and Control*, pp. 41284142, 2007.

[20] Ross, I.M., Sekhavat, P., Gong, Q., and Fleming, A., "Optimal Feedback Control: Foundations, Examples, and Experimental Results for a New Approach," *Journal of Guidance, Control, and Dynamics*, 31 (2), 2008.

[21] Ross, I.M. and Fahroo, F., *New Trends in Nonlinear Dynamics and Control and their Applications*, ser. Lecture Notes in Control and Information Sciences. vol. 295, ch. "Legendre Pseudospectral Approximations of Optimal Control Problems", pp. 327–342, Springer-Verlag, 2003.

[22] Fahroo, F. and Ross, I. M.,"Advances in Pseudospectral Methods for Optimal Control," *AIAA Guidance, Navigation, and Control Conference*, AIAA Paper 2008-7309, 2008.

[23] Gong, Q. , Fahroo, F., and Ross, I. M.,"A Spectral Algorithm for Pseudospectral Methods in Optimal Control," *Journal of Guidance, Control and Dynamics*, Vol. 31, No. 3, pp. 460-471, 2008.

[24] Fahroo, F. and Ross, I. M.,"Convergence of the Costates Does Not Imply Convergence of the Control," *Journal of Guidance, Control and Dynamics*, Vol. 31, No. 5, pp. 1492-1497, 2008.

[25] Ross, I.M., "A Primer on Pontryagin's Principle in Optimal Control", *Collegiate Publishers*, San Francisco, CA, November 2009.

[26] Hurni, M.A. "An information-centric approach to autonomous trajectory planning utilizing optimal control techniques," Doctoral Dissertation, Naval Postgraduate School, Monterey, CA, September 2009.

[27] Ross, I.M., "A Beginner's Guide to DIDO: A MATLAB Application Package for Solving Optimal Control Problems", *Elissar Technical Report TR-711*, http://www.elissarllc.com, 2007.

[28] Gong, Q. , Kang, W. and Ross, I. M., "A Pseudospectral Method for the Optimal Control of Constrained Feedback Linearizable Systems", *IEEE Transactions on Automatic Control*, 51 (7), pp. 1115-1129, 2006.