2005 American Control Conference
June 8-10, 2005. Portland, OR, USA

Receding Horizon Implementation of MILP for Vehicle Guidance

Yoshiaki Kuwata and Jonathan How

I. MILP Trajectory Optimization

A. Problem Statement

The problem statement is to optimize the vehicle trajectory by minimizing time of arrival at the goal, given the initial position $\boldsymbol{x}_0$, the goal location $\boldsymbol{x}_{\rm goal}$, and the no-fly zones $\mathcal{X}_{\rm obst}$. The resultant trajectory must be consistent with the vehicle dynamics such as turn rate and speed constraints and avoid no-fly zones and obstacles. The vehicle is modeled as a simple point mass with position and velocity as state variables $\boldsymbol{x}$ and acceleration as control inputs $\boldsymbol{u}$ [1].

B. Fixed Horizon Minimum Time Controller

One approach to design a minimum arrival time controller is to make a plan over a fixed planning horizon [1]. At time step k, a series of control inputs $\{\boldsymbol{u}(k+i),\ i = 0, 1, \dots, n_p - 1\}$ are chosen that give the trajectory $\{\boldsymbol{x}(k+i),\ i = 1, 2, \dots, n_p\}$. Constraints are added to specify that one of the n_p trajectory points must visit the goal. The MILP optimization minimizes the time along this trajectory at which the goal is reached, using n_p binary decision variables $\boldsymbol{b}_{\rm arrival} \in \{0, 1\}$ as

$$\min_{\boldsymbol{u}(\cdot)} \ \phi_1(\boldsymbol{b}_{\rm arrival}) = \min_{\boldsymbol{u}(\cdot)} \ \sum_{i=1}^{n_p} b_{{\rm arrival},i} \Delta t$$

Experience has shown that the computational effort required to solve this optimization can grow rapidly with the length of the trajectory to be planned [1], [2], but a receding horizon approach can be used to design large-scale trajectories.

II. Receding Horizon Formulation

The receding horizon control strategy is comprised of a cost estimation phase and a trajectory design phase [3]. The cost estimation phase computes a compact "cost map" of the approximate minimum time-to-go from a limited set of points to the goal. The cost estimation phase is performed once for a given obstacle field and position of the goal, and would be repeated if the environment changes. The trajectory designer uses this cost map information in the terminal penalties of the receding horizon optimization to design a series of short trajectory segments that are followed until the goal is reached. An example of a result that would be expected from the trajectory design phase is shown schematically in Fig. 1. A trajectory consistent with discretized aircraft dynamics is designed from $\boldsymbol{x}(k)$ over a fine resolution planning horizon with length n_p steps. The trajectory is optimized using MILP to minimize

Y. Kuwata, Dept. of Aeronautics and Astronautics, MIT, Cambridge, MA 02139, USA kuwata@mit.edu

J. How Associate Professor, Dept. of Aeronautics and Astronautics, MIT, Cambridge, MA 02139, USA jhow@mit.edu

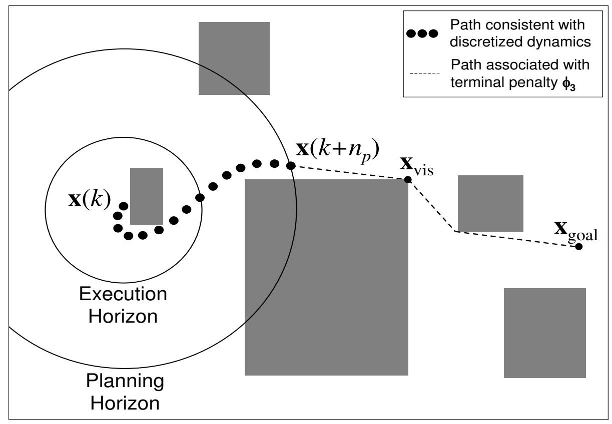

Fig. 1. Resolution Levels of the Planning Algorithm

the cost function assessed at the plan's terminal point $\boldsymbol{x}(k+n_p)$. The optimization chooses a visible point $\boldsymbol{x}_{\rm vis}$ from a list of candidate cost points $\boldsymbol{x}_{\rm cost}$, whose cost-to-go was previously estimated, plus the cost-to-go estimate $C_{\rm vis}$ for $\boldsymbol{x}_{\rm vis}$. This $C_{\rm vis}$ is estimated using a coarser model of the aircraft dynamics that can be evaluated very quickly.

$$\min_{\boldsymbol{u}(\cdot),\boldsymbol{b}_{\rm vis}} \ \phi_2 = \min_{\boldsymbol{u}(\cdot),\boldsymbol{b}_{\rm vis}} \ \frac{L_2(\boldsymbol{x}_{\rm vis} - \boldsymbol{x}(k+n_p))}{v_{\max}} + C_{\rm vis}$$

Only the first n_e steps are executed before a new plan is formed starting from $\boldsymbol{x}(k+n_e)$. Ref. [4] discusses the stability of this trajectory design approach.

III. Code Implementation Example

To show the ease of implementation, consider the AMPL and CPLEX code for the cost point selection. The values of the position $\boldsymbol{x}_{\rm vis}$ and cost $C_{\rm vis}$ are evaluated using binary variables $\boldsymbol{b}_{\rm vis} \in \{0, 1\}$ and the n points on the cost map as

$$\boldsymbol{x}_{\rm vis} = \sum_{j=1}^{n} b_{{\rm vis},j} \boldsymbol{x}_{{\rm cost},j} \tag{1}$$

$$C_{\rm vis} = \sum_{j=1}^{n} b_{{\rm vis},j} C_j \tag{2}$$

$$\sum_{j=1}^{n} b_{{\rm vis},j} = 1 \tag{3}$$

The associated AMPL source code is:

```
param costPosns {PTS, DIMS};
param costVals {PTS};
 var   bVisPt {PTS} binary;
 var   rVisPt {DIMS};
 var   visPtCost;
subject to visibilePtEqn {d in DIMS}:
 rVisPt[d] = sum{c in PTS} bVisPt[c]*costPos[c,d];
subject to visiblePointCostEqn:
 visPtCost = sum{c in PTS} bVisPt[c]*costVal[c];
subject to chooseOneVisPt:
 sum {c in PTS} bVisPt[c] = 1;
```

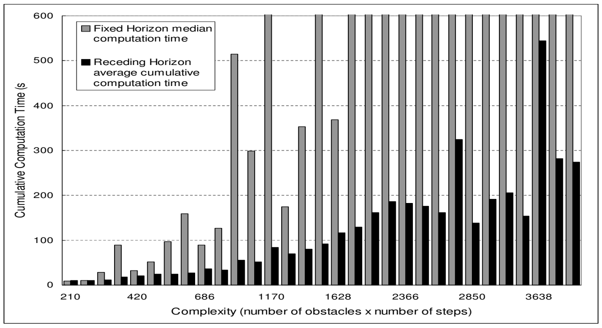

Fig. 2. Computation time reduction compared to full MILP [3]

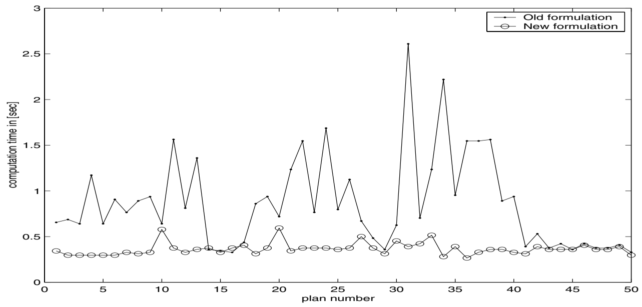

Fig. 3. Further computation time reduction due to pruning

The first block declares the parameters $\boldsymbol{x}_{\mathrm{cost},j}$ and C_j; the second block declares the binary decision variables $b_{\mathrm{vis},j}$ and continuous decision variables $\boldsymbol{x}_{\mathrm{vis}}$ and C_{vis}; and the third block specifies the constraints to select the cost point and the cost value. Note that the MILP implementation is as simple as the mathematical equations Eqs. 1 to 3. More complete sample codes are available for download at the ACL web site `http://hohmann.mit.edu/MILP`.

IV. EXTENSIONS

Simulation results confirmed that RH-MILP significantly reduces the computation time for complex trajectory planning problems, as shown in Figure 2. Hardware experiments also showed that the algorithm can generate trajectories on-line while accounting for the dynamic changes in the environment [5]. This section discuss several extensions to further enhance the RH-MILP capabilities.

A. Pruning Algorithm

Some of the nodes $\boldsymbol{x}_{\mathrm{cost}}$ used in the cost map will never be selected in MILP because they are behind obstacles or a tight maneuver is required to reach them. The pruning algorithm in [6] was developed to eliminate most of these nodes before performing the MILP optimization, which greatly reduces the search space for binary variable b_{vis} in Eqs. 1 to 3, resulting in a drastic reduction in the computation time (see Figure 3). This formulation only prunes the nodes that will never be selected, and still retains the freedom to choose the best path from the trees of nodes that remain.

B. Feasibility Guarantee

By including more detailed vehicle dynamics (*i.e.,* turning capability) in the cost estimation phase, it can be proven that RH-MILP always has a feasible solution and that the vehicle reaches the goal in finite time [6]. Modified Dijkstra's algorithm is developed that rejects infeasible node sequences to form a modified cost map. In order to maintain the feasibility of the trajectory optimization, the planning horizon must be extended beyond the execution horizon, and a lower bound of the planning margin required was derived analytically.

C. Three Dimensional Environment

In a three dimensional environment the paths are chosen to stay as close to the terrain as possible to avoid exposure to threats, but the vehicle can choose to pop-up over the obstacles if necessary. Candidate cost points are placed on the edges of the obstacles as opposed to just the obstacle corners. The extended RH-MILP algorithm is shown to be computationally tractable in complicated environment [7].

D. Initial Guess

In order to shorten the solution time of the MILP, an initial feasible solution can be provided with the solver [7], [8]. RH-MILP executes only the first step of the n_p step plan, and re-optimizes from the state that will be reached. The decisions (e.g. visible point selection, obstacle avoidance) made in the previous solution could be used when constructing an initial guess.

V. CONCLUSIONS

This paper presents MILP based techniques for trajectory optimization of unmanned aerial vehicles. To reduce the computation load associated with MILP optimization, a receding horizon controller has been developed. The combination of coarse cost map and detailed short trajectory significantly reduces the size of the MILP optimization. Several extensions are also presented that further reduce computation time and expand the capabilities of RH-MILP.

ACKNOWLEDGMENTS

Research funded by AFOSR Grant # F49620-01-1-0453.

REFERENCES

[1] A. Richards and J. How, "Aircraft Trajectory Planning With Collision Avoidance Using Mixed Integer Linear Programming," in *Proceedings of the IEEE American Control Conference*, May 2002.
[2] T. Schouwenaars, B. D. Moor, E. Feron, and J. How, "Mixed Integer Programming for Multi-Vehicle Path Planning," in *Proceedings of the European Control Conference*, September 2001.
[3] J. Bellingham, A. Richards, and J. How, "Receding Horizon Control of Autonomous Aerial Vehicles," in *Proceedings of the IEEE American Control Conference*, May 2002.
[4] J. Bellingham, Y. Kuwata, and J. How, "Stable Receding Horizon Trajectory Control for Complex Environments," in *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, Aug 2003.
[5] A. Richards, Y. Kuwata, and J. How, "Experimental Demonstrations of Real-time MILP Control," in *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, Aug 2003.
[6] Y. Kuwata and J. How, "Stable Trajectory Design for Highly Constrained Environments using Receding Horizon Control," in *Proceedings of the IEEE American Control Conference*. IEEE, June 2004.
[7] Y. Kuwata and J. How, "Three Dimensional Receding Horizon Control for UAVs," in *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, August 2004.
[8] ILOG, *ILOG CPLEX User's guide*, 1999.