Safe Receding Horizon Path Planning for Autonomous Vehicles [*]

Tom Schouwenaars[†] Eric Feron[†] Jonathan How[§]

[†] Laboratory for Information and Decision Systems
[§] Space Systems Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139

{toms, feron, jhow}@mit.edu

Abstract

This paper extends a recently developed approach to optimal path planning of autonomous vehicles to account for safety. Recent research in the field of trajectory optimization for Unmanned Aerial Vehicles (UAV's) is based on the use of mixed integer linear programming (MILP) to account for obstacle and collision avoidance constraints. To allow for real-time computation, a receding horizon approach was proposed, in which a path is computed online by solving a MILP over a limited horizon at each time step. However, satisfying anti-collision constraints over this limited planning horizon at a certain instant in time does not necessarily yield a collision free path during future time steps. In this paper, a receding horizon algorithm is presented that ensures *a priori* safety for a single vehicle with limited knowledge of the environment, if only stationary obstacles are present. More precisely, it guarantees that the vehicle can always transition to a safe mode, by maintaining a feasible path to a predefined safe state at each time step.

1 Introduction

In recent years, the space and defense community has gained considerable interest in fully autonomous space- and aircraft systems, or so called Unmanned Aerial Vehicles (UAV's). Such systems need no or minor human control from a ground station, thereby reducing operating costs and enabling missions in harsh or remote environments. A significant portion of the autonomy consists of path planning capabilities: the problem is to guide the vehicle through an obstacle field, taking into account its dynamic and kinematic properties.

It has been proven that the motion planning problem is intrinsically NP-hard [1]. Namely, the space of possible control actions is extremely large, requiring simplifications to reduce the dimensionality, when a solution needs to be found in practical time. In recent years, several methods based on randomized algorithms, such as Probabilistic

[*]Research funded under the DARPA Software Enabled Control Program, AFRL contract # F33615-01-C-1850. Tom Schouwenaars was partly supported by a Francqui Fellowship of the Belgian American Educational Foundation.

Road Maps (PRM's) [2] and Rapidly-exploring Random Trees (RRT's) [3], have been developed that tackle the dimensionality problem by sampling the possible control actions. However, although computationally tractable, these methods generally produce non-optimal trajectories.

An alternative approach based on Mixed Integer Linear Programming (MILP) was presented in [4, 5]. MILP is a powerful mathematical programming framework that extends continuous linear programming to include binary or integer decision variables [6]. These variables can be used to model logical constraints such as obstacle and collision avoidance rules, while the dynamic and kinematic properties of the vehicle are formulated as continuous constraints. The main advantage of using MILP for path planning is that time and/or fuel optimal trajectories can be computed with hard anti-collision constraints. Thanks to the implementation of powerful state-of-the-art algorithms in software packages such as CPLEX [7], and thanks to the increase in computer speed, MILP has become a feasible option for real-time path planning.

One approach that allows for real-time path planning is a receding horizon strategy, in which a new segment of the total path is computed at each time step by solving a MILP over a limited horizon. This strategy was proposed in [4, 5] and extended to account for local minima in [8]. However, as shown in this paper, despite the hard anti-collision constraints, the receding horizon strategy has no safety guarantees regarding avoidance of obstacles in the future: the algorithm may fail to provide a solution in future time steps due to obstacles that are located beyond the surveillance and planning radius of the vehicle. This translates into the MILP becoming infeasible at a certain time step. Therefore, in this paper an alternative receding horizon algorithm is presented that ensures *a priori* safety (*i.e.* no collision with unforeseen obstacles) by maintaining a feasible "rescue" path to a naturally collision-free basis state at each time step. Whenever the optimal planning problem then becomes infeasible, the vehicle can transition to the safe rescue path.

The paper is organized as follows. Section 2 briefly recapitulates the MILP formulation for path planning as presented in [4], restricting the exposition to the single vehicle case. In Section 3, the benefits of a receding horizon planning strategy are outlined, whereas in Section 4, it is shown that this approach introduces extra safety issues as described above. Section 5 then presents the alternative safe planning algorithm, which is applied to several examples in Section 6.

2 Path planning using MILP

The basic problem discussed in this paper is that of calculating an optimal path between two states of an autonomous vehicle. The discussion here is restricted to the case of fuel-optimal planning, as this leads to a straightforward linear cost function; a linear time-optimal formulation was proposed in [9]. For clarity of exposition, only planar motion of a vehicle with linear dynamics moving between rectangular obstacles is considered here. It is straightforward to extend the formulation to 3D motions, and to account for polygonal (polyhedral) obstacles. Moreover, ongoing research is aimed at including nonlinear dynamics.

To formulate the mathematical program, the vehicle dynamics are represented by a discretized linear state space model ($\mathbf{A}$,$\mathbf{B}$). Denote the initial state of the vehicle by $\mathbf{s}_{init}$, define the desired final state by $\mathbf{s}_f$, and assume for simplicity that the fuel consumption is proportional to the 1-norm of the input $\mathbf{u}$. The optimal planning problem over T time steps can then be formulated as follows:

$$\min_{\mathbf{s}_i,\mathbf{u}_i} J_T = \sum_{i=0}^{T-1} \left(\mathbf{q}'|\mathbf{s}_i - \mathbf{s}_f| + \mathbf{r}'|\mathbf{u}_i| \right) + f(\mathbf{s}_T, \mathbf{s}_f) \tag{1}$$

$$\text{subject to} \quad \begin{aligned} \mathbf{s}_{i+1} &= \mathbf{A}\mathbf{s}_i + \mathbf{B}\mathbf{u}_i,\ i = 0 \dots T-1 \\ \mathbf{s}_0 &= \mathbf{s}_{init} \end{aligned} \tag{2}$$

in which $\mathbf{q}$ and $\mathbf{r}$ are nonnegative weighting vectors, and $f(\mathbf{s}_T, \mathbf{s}_f)$ is a linear terminal cost term. By introducing auxiliary variables and constraints, the above problem can be reformulated as a linear program (LP) [4].

To avoid collision with a rectangular obstacle with lower left corner $(x_{\min}, y_{\min})$ and upper right corner $(x_{\max}, y_{\max})$, each trajectory point (x_i, y_i) must satisfy the following set of constraints:

$$\begin{aligned} x_i &\leq x_{\min} + Mb_{i1} \\ -x_i &\leq -x_{\max} + Mb_{i2} \\ y_i &\leq y_{\min} + Mb_{i3} \\ -y_i &\leq -y_{\max} + Mb_{i4} \\ \sum_{k=1}^{4} b_{ik} &\leq 3 \end{aligned} \tag{3}$$

Here, b_{ik} are binary variables and M is a sufficiently large positive number. The last constraint ensures that at least one of the position constraints is active, thereby guaranteeing that the trajectory point (x_i, y_i) lies outside the rectangle. This constraint set should be formulated for each time step $i = 1 \dots T$, and for each obstacle in the operating region of the vehicle.

The objective function (1) combined with the constraints (2) and (3) constitutes a mixed integer linear program (MILP). Several additional linear kinematic and dynamic specifications can be added to this problem, including limits on control amplitude and velocity [4], as well as limits on turn rate [5, 9].

3 Receding horizon planning

3.1 Fixed arrival time

The most direct implementation of the MILP-strategy is to use a *fixed arrival time*, which means that the arrival time is specified and the mathematical program is solved once over the entire time range. Hence, in this approach, instead of including it in the cost function, the desired final state $\mathbf{s}_f$ at time step T can be constrained:

$$\mathbf{s}_T = \mathbf{s}_f \tag{4}$$

and the cost function (1) can be reduced to

$$J_T = \sum_{i=0}^{T-1} \mathbf{r}'|\mathbf{u}_i|. \tag{5}$$

The calculated path is thus designed to minimize the fuel consumption for the specific time range T and weight vector $\mathbf{r}$.

A major drawback of solving a MILP, however, is that the computation time increases at least polynomially with the number of variables and constraints. Therefore, the fixed arrival time approach can only be applied to relatively small problems, *i.e.* to problems with a short planning horizon and few obstacles. Moreover, as the path is computed off-line, this strategy does not allow for changes in the dynamics of the vehicle or for modifications in the obstacle field. As such, the fixed arrival time approach is not robust to changes or uncertainties in the environment. All necessary information has to be available beforehand, *i.e.* before the vehicle starts its mission. During the execution phase, the navigation control system of the vehicle is restricted to tracking the precomputed path. A deviation from this path is likely to propagate in future time steps, leading to uncertainty about the actual final state and the safety with respect to collisions. The fixed arrival time *path planning* strategy therefore needs to be combined with an appropriate *path keeping* control architecture.

3.2 Receding horizon

When the computation time becomes impractically long, or when real-time replanning is required due to limited a priori information, a receding horizon (RH) planning strategy is appropriate. In this case, the path of the vehicle is composed of a sequence of locally optimal segments. At a certain time step, the MILP with cost function (1) is solved for T future time steps, where the length T of the planning horizon is chosen as a function of the available computational resources and the distance over which the environment is fully characterized. Solving this "local" MILP provides the input commands for the T future time steps. However, only a subset of these T input commands is actually implemented. The process is then repeated, and a new set of commands is developed for the next time window. Usually, the applied subset is restricted to the first control input, such that a new set of input commands is calculated at each time step. A general modeling framework for such receding horizon control schemes with logical constraints has been developed by Bemporad *et al* [10].

In case of a real-time implementation, the receding horizon strategy inherently introduces robustness to uncertainties in the dynamic model of the vehicle and to disturbances and changes in the environment. As typically a new MILP is solved at every time step, the new initial state can be precisely measured and used in the mathematical program. Moreover, new information about the environment can be incorporated at each time step, which is crucial when the environment is explored online.

3.3 Terminal cost

An important aspect of the RH-strategy is the construction of an appropriate terminal cost function. The terminal cost function provides an estimate of the cost-to-go from the last state in the planning horizon to the desired final state. A basic approach was presented in [4]. The terminal cost function was defined as the 1-norm of the difference between the last state of the planning horizon and the desired terminal state. However, with this formulation, the vehicle can get trapped in local minima behind obstacles. A better framework was proposed in [8]. There, the terminal cost is based on a graph representation of the environment, and a shortest path algorithm is executed offline to produce an approximate cost-to-go map from the nodes in the graph to the goal. A

MILP-formulation is used to connect the last state of the planning horizon with the cost-to-go map. A prerequisite of this approach is that the obstacle field is known a priori. In case of an uncertain environment, a randomized algorithm could be used, as proposed by Frazzoli in [11].

4 Safety failure

4.1 Infeasibility

Although the MILP-formulation allows for hard anti-collision constraints, in a receding horizon setting, this still does not guarantee that no collisions will occur. Namely, because of the limited look-ahead horizon, the vehicle can be led to a critical state for which the MILP has no solution in the next iteration: a feasible solution over T time steps at time step i does not guarantee a feasible MILP at time step $(i + 1)$.

For instance, the limited look-ahead horizon can cause the vehicle to move too close to an obstacle, without violating the anti-collision constraints. Consider a case in which in the last time step of the planning horizon, the vehicle is moving at maximum speed, while its position is just outside an obstacle that has not been spotted yet. Since the position of the vehicle satisfies the anti-collision constraints, this situation corresponds to a feasible solution of the MILP. However, if the time or space the vehicle needs to brake or turn exceeds the planning horizon or the available maneuver space, a collision with the obstacle will result.

This translates to the MILP not having a feasible solution in the next time step. The collision can thus be detected or predicted before the vehicle actually hits the obstacle, *i.e.* as soon as the MILP does not have a feasible solution at a certain time step, a collision will result in the future. More precisely, the time elapsed until the collision occurs, lies between the length of the planning horizon minus one time step and the total length of the planning horizon. These insights in the failure mechanism lead to a "safe" implementation of the receding horizon approach, *i.e.* one that is guaranteed to be collision-free, despite the use of the limited look-ahead horizon.

4.2 Examples

The following two examples illustrate the situation described above. As a first example, consider the simple case in which the receding planning horizon is shorter than the time the vehicle needs to brake from maximal velocity. The vehicle is modeled as a double integrator with unit mass. The maximal speed is $1m/s$; the maximal acceleration and decelaration are respectively $0.2m/s^2$ and $-0.2m/s^2$. As a result, the minimal time needed to brake from maximal velocity is $5s$, corresponding to a minimal braking distance of $7.5m$. However, a horizon of only $3s$ is used, with a sampling time of $0.5s$. Hence, the receding horizon (RH) MILP contains 6 time steps and only obstacles within a distance of $3m$ will play a role in the MILP-solution. Namely, the anti-collision constraints associated with obstacles that lie beyond the maximal distance the vehicle can travel within the planning horizon, are trivially satisfied.

The vehicle starts in point $(-12, 0)$ at zero speed, and is supposed to stop right in front of the obstacle at point $(-2.5, 0)$. Since the planning horizon is too short to take into account obstacles that are further than $3m$ away, and since the braking distance is at least $7.5m$, execution of the RH-algorithm will clearly lead to a collision. The situation is depicted in Figure 1. After 22 time steps, the MILP becomes infeasible: the vehicle is moving at maximal speed and has approached the obstacle too closely to avoid it. As a result, the MILP no longer has a feasible solution in the next time step, and a collision will occur after the last time step of the present horizon.

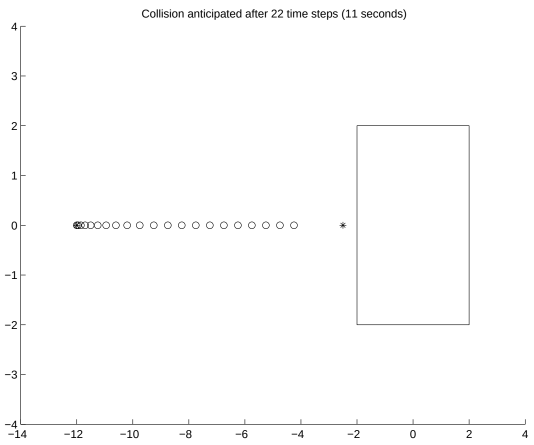

Figure 1: The receding horizon trajectory leads to an infeasible MILP after 22 time steps.

In a second – more realistic example– the vehicle also has a limited turn rate of $15deg/s$. This nonlinear constraint can be approximated by a set of linear inequalities on the thrust f and velocity v components in the x- and y-direction [5, 9]:

$$\forall i \in [0...T-1],\ \forall k \in [1...K]:$$

$$f_{x_i} \sin\left(\frac{2\pi k}{K}\right) + f_{y_i} \cos\left(\frac{2\pi k}{K}\right) \leq f_{max} \tag{6}$$

$$v_{x_i} \sin\left(\frac{2\pi k}{K}\right) + v_{y_i} \cos\left(\frac{2\pi k}{K}\right) \leq v_{max} \tag{7}$$

For this example, $v_{max} = 1m/s$ and $f_{max} = 0.262N$.

In this case, it is hard to tell which planning horizon length will ensure safe operation. Namely, because of the limited turn rate, the way the vehicle maneuvers around a certain obstacle affects the ability to avoid the next obstacle. This is shown in Figures 2 and 3, where a planning horizon of respectively $10s$ and $8s$ is used. The vehicle starts in $(4, -2.5)$ at zero speed and has to maneuver to stagnation in $(15, 8)$. With the $10s$ horizon, the vehicle safely reaches the goal in $29s$ (see Figure 2). With an $8s$ horizon, however, the MILP becomes infeasible after 17 time steps (see Figure 3). Because the vehicle makes the first turn at a slightly higher speed than in the $10s$ horizon case, it will not be able to maneuver through the opening further up.

5 Safety algorithm

5.1 Safe basis states

The key to a safe receding horizon implementation is to detect in time, whether a certain state of motion will lead to a collision, *i.e.* to detect in time whether the MILP will become infeasible in the future time steps. The planning algorithm must then prevent the vehicle from evolving to a state from which it cannot move further without colliding.

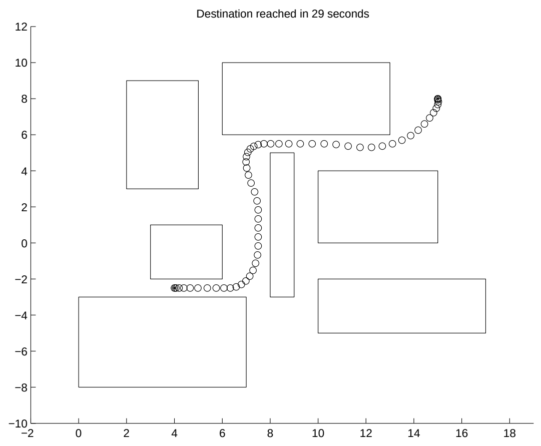

Figure 2: A planning horizon of 10s results in a safe trajectory to the goal.

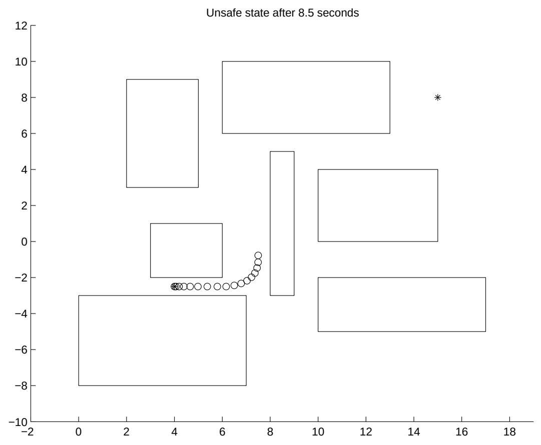

Figure 3: A planning horizon of 8s results in an infeasible MILP after 17 iterations.

A certain state of motion is defined to be *safe*, if from that state on, the vehicle can perform a maneuver that brings it to a *basis* state, *i.e.* to a certain predefined motion or position that is naturally collision-free. The basis state typically depends on the nature and the dynamics of the vehicle. A simple example of such a basis state is zero velocity on the ground, *i.e.* stagnation in any obstacle-free position in the field. For a helicopter, the basis state could be defined as hovering motion at a certain altitude. For a fixed-wing aircraft, the basis state could be to fly in a circle for indefinite time, while maintaining a minimum speed.

An *unsafe* state, on the other hand, is defined as a state from which the vehicle cannot maneuver to its basis state. The safety strategy must now ensure *a priori* that the vehicle never enters an unsafe state, and can thus always return to its basis state. A straightforward way to accomplish this is to maintain a feasible "rescue path" to the basis state at all times. The vehicle is only allowed to move to another state, if there exists a rescue path starting from that state. The safety algorithm must thus look ahead in future, and check whether such a rescue path exists. If the optimal next state resulting from the RH-MILP turns out to be unsafe, the first step of the rescue path from the current state should be executed.

The computation of the rescue path is done by solving a fixed arrival time MILP (see Section 3.1) from the proposed next state, with the final state corresponding to the basis state. Since the length of the receding horizon is chosen such that the planned segment always lies within the surveillance radius, the length of the fixed horizon can be taken the same. Provided that it exists, the rescue path will then automatically lie in the known area as well. However, in case the receding horizon length is too conservative, the fixed horizon length can be taken longer, provided extra constraints are included in the MILP that ensure that the rescue path is fully contained in the surveillance region. In case the environment is static, one could also constrain the rescue path to lie in the total region that has already been explored.

Because (fuel-) optimality is of minor importance during a rescue maneuver, and because the safety check is preferably carried out in minimal time, a simple cost function can be used. A good choice is to minimize the first input. That way, the fuel consumption is still locally minimized, as typically only the first segment of a rescue path from a certain unsafe state is executed. This is a direct consequence of the algorithm described below: whenever the vehicle moves to a new state, the rescue path is updated, *i.e.* a new rescue path starting in that state is computed.

5.2 Algorithm

Schematically, the planner executes the following algorithm:

Step 1: Solve the receding horizon MILP with the current state C as the initial state. This yields the optimal next state N. If N is the desired final state F or the predefined basis state B, go to **Step 5**. Else, go to **Step 2**.

Step 2: Check whether the next state N is safe, by solving a fixed arrival time MILP with N as the initial state, and the predefined basis state B as the final state. If this MILP is feasible, go to **Step 3**. Else, go to **Step 4**.

Step 3: Let the vehicle move to N, store the rescue path and substitute C by N. Repeat **Step 1**.

Step 4: Execute the first step from the rescue path from C. This yields a new safe state S. Store the remaining part of the rescue path as the rescue path from S. Substitute C by S and repeat **Step 1**.

Step 5: Let the vehicle move to N. End the algorithm.

By maintaining a rescue path at all time steps, the algorithm guarantees *a priori* that the vehicle will never move to an unsafe state. After the first segment of a rescue path is executed, a RH-MILP from the new safe state is solved. The new RH-MILP is guaranteed to be feasible, as the rest of the rescue path is always a feasible solution. In the worst case scenario, the RH-solution corresponds to the rest of the rescue path. If the desired final state can be reached in a safe way, the algorithm ends when that final state is reached. If there exists no safe path to the destination, the algorithm will end in the basis state, by executing a rescue path to the end.

6 Examples

The safe planning algorithm presented above is now applied to the examples of Section 4.2. Figures 4 and 5 show the trajectories when safe RH-planning is carried out. In both examples, the same planning horizon is used as in the unsafe version ($3s$ and

8s respectively). For both cases, the basis state is defined as a zero velocity state with unspecified position.

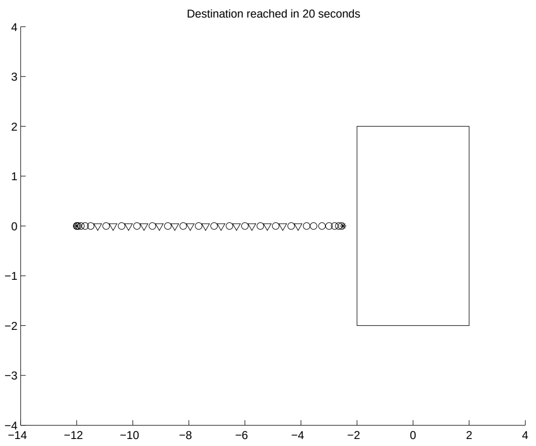

Figure 4: Collision-free trajectory with safe RH-planning. The triangles indicate the time steps at which the rescue path is executed.

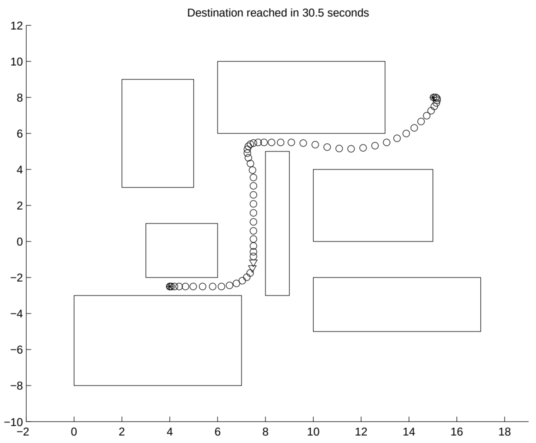

Figure 5: Collision-free trajectory with safe RH-planning and limited turn rate. The safety mechanism detects the hazardous situation at the 17th time step and slows down the vehicle.

The triangles in the figures indicate the time steps at which the vehicle transitions to the safe mode and executes the rescue path. In Figure 4, the safety mechanism continually slows down the vehicle, maintaining it at a safe speed between $0.5m/s$ and $0.6m/s$, such that it can always stop within the 3s horizon. In Figure 5, the vehicle transitions to the rescue mode for two time steps. During that time, it slows down, enabling a safe sharp turn later on.

7 Conclusion & Future work

In this paper, a safe real-time path planning algorithm was presented, that uses mixed integer linear programming in a receding horizon framework. At each time step, the planning algorithm maintains a feasible path to a naturally safe basis state, thus ensuring a safe way out whenever the optimal planning problem becomes infeasible. As such, the algorithm guarantees a priori safety. The method was illustrated by two examples: one in which the planning horizon is shorter than the time needed to stop, a second one in which the maneuverability is constrained by a limited turn rate.

Future work will concentrate on extending the safety mechanism to multi-vehicle systems. Safe basis states for such systems need to be defined, and centralized, as well as decentralized approaches will be investigated. Also, current research is aimed at including the feasibility constraints directly into the receding horizon MILP, such that an *optimal* safe path can be computed directly. Moreover, the robustness of the technique with respect to uncertainties in the dynamics and disturbances in the environment will be investigated.

References

[1] J.E. Hopcroft, J.T. Schwartz, and M. Sharir. "On the complexity of motion planning for multiple independent objects: PSPACE-hardness of the "Warehouseman's Problem"". *International Journal of Robotics Research*, 4(3):76–88, 1984.

[2] L.E. Kavraki, F. Lamiraux, and C. Holleman. "Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces". *IEEE Transactions on Robotics and Automation*, 12(4):566–580.

[3] S.M. LaValle and J.J. Kuffner. "Randomized Kinodynamic Planning". *IEEE International Conference on Robotics and Automation*, July 1999.

[4] T. Schouwenaars, B. DeMoor, E. Feron, and J. How. "Mixed Integer Programming for Multi-Vehicle Path Planning". *Proceedings of the 2001 European Control Conference*, pages 2603–2608, September 2001. Porto, Portugal.

[5] T. Schouwenaars. "Mixed Integer Programming for Optimal Collision-Free Path Planning of Autonomous Vehicles". Master's thesis, Katholieke Universiteit Leuven, Department of Electrical Engineering, Sista, Leuven, Belgium, May 2001.

[6] C. A. Floudas. *"Nonlinear and Mixed-Integer Programming - Fundamentals and Applications"*. Oxford University Press, 1995.

[7] *ILOG CPLEX User's guide*. ILOG, 1999.

[8] J. Bellingham, A. Richards, and J. How. "Receding Horizon Control of Autonomous Aerial Vehicles". *Proceedings of the 2002 American Control Conference*, May 2002. Anchorage,AK.

[9] A. Richards and J. How. "Aircraft Trajectory Planning With Collision Avoidance Using Mixed Integer Linear Programming". *Proceedings of the 2002 American Control Conference*, May 2002. Anchorage,AK.

[10] A. Bemporad and M. Morari. "Control of Systems Integrating Logic, Dynamics, and Constraints". *Automatica*, Vol.35:407–427, 1999. Pergamon/Elsevier Science, New York NY.

[11] E. Frazzoli, M. A. Dahleh, and E. Feron. "Real-Time Motion Planning for Agile Autonomous Vehicles". *AIAA Journal of Guidance, Control, and Dynamics*, 25(1):116–129, 2002.