JOURNAL OF AEROSPACE COMPUTING, INFORMATION, AND COMMUNICATION
Vol. 1, December 2004

Hybrid Model for Trajectory Planning of Agile Autonomous Vehicles

Tom Schouwenaars,[*] Bernard Mettler,[†] Eric Feron,[‡] and Jonathan How[§]
Massachusetts Institute of Technology, Cambridge, Massachusetts, 02139

This paper presents a hybrid architecture for autonomous trajectory planning of agile vehicles. A velocity control system providing the ability to accurately track trajectories is combined with a maneuver scheduler that enables execution of pre-programmed agile maneuvers. The closed-loop dynamics under this control architecture are described by a simple hybrid model, consisting of a set of constrained, linear time-invariant modes and discrete fixed-duration transitions in the state space. Given these dynamics, mixed integer linear programming (MILP) is used to compute optimal trajectories in cluttered environments. Continuous constraints and binary logic are combined to model the constraints governing the dynamics, to formulate switching rules between different velocity modes, to encode execution of agile maneuvers, and to account for obstacle avoidance. Both offline time-optimal planning and online receding horizon formulations are presented. The framework is applied in detail to a small-scale helicopter, for which several receding horizon results are given. A discussion about practical considerations regarding a real-time implementation concludes the paper.

I. Introduction

Taking advantage of the full range of vehicle maneuverability in an autonomous fashion is key to a number of potential unmanned aerial vehicles (UAV) tasks. Examples include tracking of moving targets, flying through cluttered (e.g. urban) environments, and tactical flight, such as nap of the earth. In these applications, autonomous agile maneuvering can either be necessary – for instance to effectively avoid pop-up obstacles– or may represent a competitive advantage. Besides exploiting the agile capabilities of the vehicle, autonomous guidance systems should also account for the goal of the mission and for the available information about the environment, such as the location of obstacles.

It is known, however, that motion planning is intrinsically NP-hard.[1,2] For vehicles with fast and complex dynamics, such as small-scale rotorcraft,[3] it is therefore impractical to consider the full equations of motion in the development of an autonomous trajectory planning system. A well-established idea to reduce the complexity is to first organize the vehicle dynamics through some form of control augmentation, such as velocity controllers, and to then design the guidance system using the simpler closed-loop dynamics. However, for highly agile vehicles, such an approach might restrict the performance. An alternative method is to use a "maneuver automaton" as was introduced by Frazzoli et al.[4] The framework was later applied to an X-Cell miniature helicopter as described in Refs. 5 and 6. With this approach, the vehicle is modeled as a hybrid automaton, consisting of a set of discrete equilibrium trim conditions and transitions between these trims, called maneuvers. Optimal trajectories are obtained in real- time by evaluating the possible discrete actions at each time step, i.e. to stay on the current trim trajectory or to execute a maneuver. The decision is made according to an optimal policy operating on a value function, which results from a dynamic program that is solved offline by value iteration.[7] The maneuver automaton as described above, however, has several drawbacks. These are primarily related to the fact that the vehicle dynamics are

This paper is part of the December Special Section on Intelligent Systems. Received 19 August 2004; revision received 15 November 2004; accepted for publication 23 November 2004.

[*]Research Assistant, Laboratory for Information and Decision Systems, toms@mit.edu. Student Member, AIAA
[†]Post-doctoral Associate, Laboratory for Information and Decision Systems, bmettler@mit.edu. Associate Member, AIAA.
[‡]Associate Professor, Laboratory for Information and Decision Systems, feron@mit.edu. Associate Fellow, AIAA.
[§]Associate Professor, Aerospace Controls Laboratory, jhow@mit.edu. Senior Member, AIAA.

constrained to a finite set of motion primitives. Namely, velocity is discretized into several trims with constant speeds, thus restricting the vehicle's behavior to one of these. The lack of continuous velocity modes and the discretization used in the value function can be a problem when precise navigation is required.[5,6] Moreover, since one operating region is typically discretized into multiple trim conditions with corresponding transition maneuvers, the complexity of the maneuver automaton and corresponding dynamic program increases significantly with the resolution of the discretization.

In this paper, we present an alternative approach based on a hybrid architecture that combines a velocity control system and a maneuver scheduler. Using this framework, optimal trajectory design can be formulated as a mixed integer linear program (MILP). MILP permits continuous optimization over several velocity control modes, allows inclusion of dynamic and kinematic constraints, and can incorporate binary logic such as the decision to execute a maneuver. Moreover, it allows to directly account for obstacle and collision avoidance constraints in the trajectory planning problem.[8] We will specialize the approach to the case of a small-scale rotorcraft using a MILP model based on MIT's aerobatic X-Cell helicopter.

The paper is organized as follows. Section II presents the hybrid control architecture and a corresponding high-level dynamic model. Given these dynamics, section III outlines the use of MILP for optimal guidance. In section IV, the MILP framework is applied to guidance of a small-scale helicopter, for which simulation results are presented in section V. Section VI then discusses some practical considerations regarding a real-time implementation, and section VII concludes.

II. Hybrid Control Architecture for Guidance

A. Automatic Control of Agile Vehicles

The guidance framework presented in this paper is based on the analysis of human control of highly agile, small-scale helicopters.[9] It shows two distinct operating regimes: tracking of trim trajectories and maneu-vering. The two modes are distinctly set apart in terms of control strategy and dynamic conditions:

- Tracking operations take place around trim trajectories. Control around these trajectories involves continuous feedback, and the dynamics are approximately linear.
- Maneuvering actions are of finite duration and start and end on trim trajectories. The control activity typically involves large amplitude input commands that exploit the extreme performance of the vehicle and result in large changes of its state. The dynamics across this range are typically nonlinear. Control is dominated by feed-forward actions; feedback may include discrete switching events triggered by state thresholds.

Tracking trim trajectories is a well researched area; automatic maneuvering, however, is more challenging due the highly nonlinear dynamics. Instead of applying traditional nonlinear control methods such as feedback linearization,[10] recently a control logic inspired by the above human strategies was developed for an autonomous miniature helicopter.[11] It combines angular rate controllers and a timing logic that allows tracking of pre-programmed reference trajectories. The amplitude and timing of these trajectories or maneuvers were extracted from piloted flight test experiments. Prior to and upon exit from a maneuver, gain-scheduled linear quadratic trim tracking controllers are used. Using this approach, several aerobatic maneuvers were successfully implemented, including a snaproll, a hammerhead, and a split-S.[12]

A block diagram of the switching control architecture is shown in figure 1. The velocity controllers enable the vehicle to accurately track trajectories throughout a large region of the flight envelope, whereas the maneuvers take the helicopter through its extreme range of performance. Under this control architecture, the closed-loop dynamics of the agile vehicle can be accurately described by a combination of low-order, linear time-in variant (LTI) equations of motion and discrete state transitions. An abstract representation of this structure is given in figure 2: it shows that maneuvers start and end in the linear velocity control regime.

More generally, for any type of unmanned agile vehicle, the combination of gain-scheduled LTI modes and a finite number of fast, pre-programmed maneuvers enables a broad range of behaviors that can be exploited when designing mission-specific trajectories. The benefits of this approach are several. First, the model allows for precise navigation in the velocity control mode, without compromising on agility when extreme transitions are required, such as during reactive threat or obstacle avoidance. Second, compared to the maneuver automaton approach, [4] the architecture significantly simplifies the development of a motion primitive library: the problem is reduced to selecting a few LTI modes and a small set of agile maneuvers.

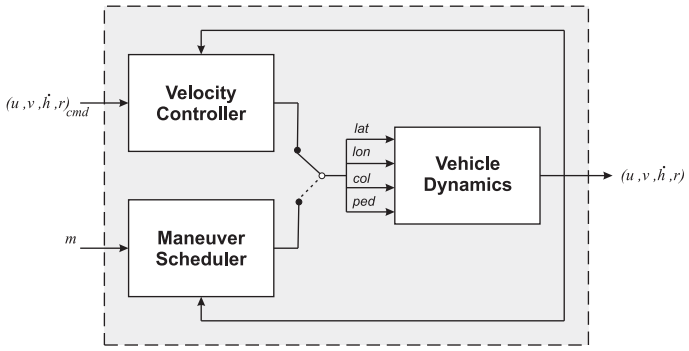

Figure 1. Hybrid control architecture: the helicopter can be controlled through a velocity control system, or through a maneuver scheduler that allows the implementation of agile maneuvers taken from a library.

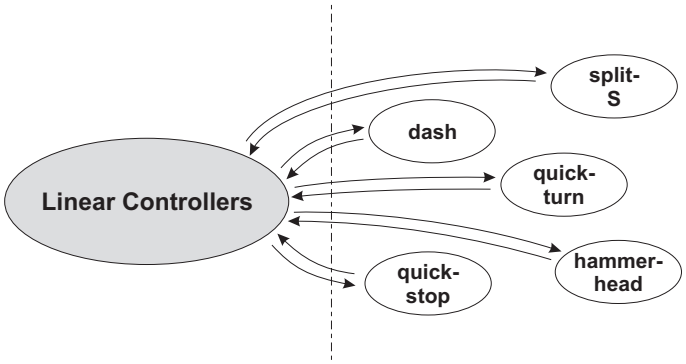

Figure 2. Abstract representation of the control architecture. The linear controllers allow continuous motion, the maneuvers are discrete transitions.

In what follows, we formalize this control structure in a mathematical framework, which can then be used to effectively formulate trajectory optimization problems.

B. Velocity Control System

Assume that the gain-scheduled velocity controllers result in L distinct, mutually exclusive LTI modes or operating regions. In general, each LTI mode l corresponds to a discrete-time state space model $\boldsymbol{x}_{i+1} = \mathbf{A}_l \boldsymbol{x}_i + \mathbf{B}_l \boldsymbol{u}_i$ in some vector space $\mathbb{R}^{n_s}$, where subscript i indicates the discrete time step. The state vector $\boldsymbol{x} \in \mathbb{R}^{n_s}$ typically contains velocity and position components in either a body-fixed and/or inertial coordinate frame. The input vector $\boldsymbol{u} \in \mathbb{R}^{n_u}$ generally consists of reference velocity commands or accelerations, again in either coordinate frame.

The key requirement is that the state space models describe the closed-loop dynamics of the vehicle in a form that is compatible with the type of desired guidance. For example, if a waypoint follower is available, a closed-loop model incorporating this controller could be considered that takes inertial positions as inputs. Alternatively, it might be desirable to directly steer the vehicle using body frame velocity commands. Since the formulation that we will present can capture both coordinate frames or a combination thereof, we will make abstraction of this distinction. Using the state space model, the solution to the trajectory planning algorithm will consist of a sequence of inputs in the appropriate form.

Besides the state space matrices $(\mathbf{A}_l, \mathbf{B}_l)$, each operating region l is characterized by a set of feasible states $\mathcal{X}_l \subseteq \mathbb{R}^{n_s}$ and feasible inputs $\mathcal{U}_l \subseteq \mathbb{R}^{n_u}$, such that when $\boldsymbol{x} \in \mathcal{X}_l$, the vehicle is in mode l and only inputs $\boldsymbol{u} \in \mathcal{U}_l$ are allowed. Among other, these constraint sets can capture bounds on velocity, acceleration, and turn rate for each LTI mode, as well as obstacle avoidance requirements. The overall constraint sets $\mathcal{X}_l$ and $\mathcal{U}_l$ are typically non-convex (e.g. because of the presence of obstacles in the environment), but subsets of these constraints, such as maximum speed, may be convex. Hence, the sets can always be approximated

by a combination of polyhedral regions and intersections of polyhedra, and be described as follows:

$$\mathcal{X}_l = \begin{cases} \boldsymbol{x}^T \boldsymbol{f}^c_{l,1} \leq 0 & \text{and} \quad \boldsymbol{x}^T \boldsymbol{f}^c_{l,2} \leq 0 \quad \dots \quad \text{and} \quad \boldsymbol{x}^T \boldsymbol{f}^c_{l,L^c_f} \leq 0 \\ \boldsymbol{x}^T \boldsymbol{f}^n_{l,1} \geq 0 & \text{or} \quad \boldsymbol{x}^T \boldsymbol{f}^n_{l,2} \geq 0 \quad \dots \quad \text{or} \quad \boldsymbol{x}^T \boldsymbol{f}^n_{l,L^n_f} \geq 0 \end{cases} \tag{1}$$

$$\mathcal{U}_l = \begin{cases} \boldsymbol{u}^T \boldsymbol{g}^c_{l,1} \leq 0 & \text{and} \quad \boldsymbol{u}^T \boldsymbol{g}^c_{l,2} \leq 0 \quad \dots \quad \text{and} \quad \boldsymbol{u}^T \boldsymbol{g}^c_{l,L^c_g} \leq 0 \\ \boldsymbol{u}^T \boldsymbol{g}^n_{l,1} \geq 0 & \text{or} \quad \boldsymbol{u}^T \boldsymbol{g}^n_{l,2} \geq 0 \quad \dots \quad \text{or} \quad \boldsymbol{u}^T \boldsymbol{g}^n_{l,L^n_g} \geq 0 \end{cases} \tag{2}$$

Here, the vectors $\boldsymbol{f}^c_{l,\cdot}$ and $\boldsymbol{f}^n_{l,\cdot}$ denote the coefficients of the linear inequalities that respectively capture the convex and non-convex state constraints associated with the operating region l. Similarly, the vectors $\boldsymbol{g}^c_{l,\cdot}$ and $\boldsymbol{g}^n_{l,\cdot}$ define the convex and non-convex input constraints.

Summarizing, when flying in a particular LTI mode l, the dynamics are given by:

$$\begin{cases} \boldsymbol{x}_{i+1} &= \mathbf{A}_l \boldsymbol{x}_i + \mathbf{B}_l \boldsymbol{u}_i \\ t_{i+1} &= t_i + \Delta t \\ \boldsymbol{x}_i &\in \mathcal{X}_l \\ \boldsymbol{u}_i &\in \mathcal{U}_l \end{cases} \tag{3}$$

where we introduced an explicit time evolution equation using the discretization step Δt. After applying the control input $\boldsymbol{u}_i$, the next state $\boldsymbol{x}_{i+1}$ can lie in the same operating region l or the vehicle might have transitioned to another mode l'.

C. Maneuver Scheduler

As mentioned before, the maneuver scheduler allows the execution of pre-programmed maneuvers that can result in rapid and extreme changes of the vehicle's state. Consider a library of P individually designed maneuvers with fixed durations ΔT_m and fixed spatial displacements with respect to the vehicle's body frame. For trajectory planning purposes, each maneuver m can then be characterized by a discrete state update equation in the appropriate reference system, as follows:

$$\begin{cases} \boldsymbol{x}_{i+1} &= \mathbf{C}_m \boldsymbol{x}_i + \mathbf{d}_m \\ t_{i+1} &= t_i + \Delta T_m \\ \boldsymbol{x}_i &\in \mathcal{X}_m \end{cases} \tag{4}$$

Here, $\mathbf{C}_m$ is a fixed matrix and $\mathbf{d}_m$ a fixed vector describing the maneuver as an affine transformation of the feasible ingress state $\boldsymbol{x}_i \in \mathcal{X}_m$. Namely, a particular maneuver m can only be initiated when the state lies within certain bounds described by $\mathcal{X}_m$. For example, a pre-programmed hammerhead maneuver can only be executed above a certain speed. In general, the feasible entry conditions can again be approximated by a set of linear inequalities, capturing both convex and non-convex constraints on the entry state:

$$\mathcal{X}_m = \begin{cases} \boldsymbol{x}^T \boldsymbol{f}^c_{m,1} \leq 0 & \text{and} \quad \boldsymbol{x}^T \boldsymbol{f}^c_{m,2} \leq 0 \quad \dots \quad \text{and} \quad \boldsymbol{x}^T \boldsymbol{f}^c_{m,M^c_f} \leq 0 \\ \boldsymbol{x}^T \boldsymbol{f}^n_{m,1} \geq 0 & \text{or} \quad \boldsymbol{x}^T \boldsymbol{f}^n_{m,2} \geq 0 \quad \dots \quad \text{or} \quad \boldsymbol{x}^T \boldsymbol{f}^n_{m,M^n_f} \geq 0 \end{cases} \tag{5}$$

Here, the vectors $\boldsymbol{f}^c_{m,\cdot}$ and $\boldsymbol{f}^n_{m,\cdot}$ again denote the coefficients of the linear inequalities corresponding to the convex and non-convex constraints associated with maneuver m. The entry state $\boldsymbol{x}_i$ and exit state $\boldsymbol{x}_{i+1}$ must lie in the operating region of some LTI mode, which can be identical, adjacent, or non-adjacent in case of a large jump in the state space. Typically, the ingress constraint set $\mathcal{X}_m$ is a subset of only one LTI operating region $\mathcal{X}_l$.

Notice that we did not consider a set of feasible control inputs corresponding to a maneuver. Since it is pre-programmed, the only relevant control input associated with a maneuver is the binary decision whether to initiate it or not. This decision is ultimately taken by the trajectory optimization algorithm. Since any details about the physical control variables will be hidden to the optimization problem, the maneuver can be abstracted as a state transition without input.

D. LTI-Maneuver Automaton

With the corresponding operating and entry constraints, the closed-loop dynamics consisting of the combination of LTI modes and finite duration maneuvers constitute a hybrid input/output automaton,[13] which we will call an LTI-maneuver automaton (LTI-MA). A graph representation of this LTI-MA is given in figure 3. During each time step, the vehicle is either in an LTI mode or executing a maneuver. State transitions between two LTI modes, i.e. those described by equations (3), take a regular time interval Δt; transitions resulting from performing a particular maneuver m have a duration ΔT_m.

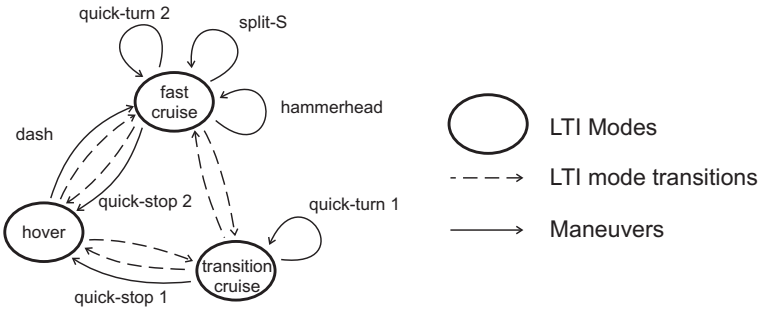

Figure 3. Graph representation of the LTI-maneuver automaton. The vehicle is either in an LTI mode, or executing a finite duration maneuver.

III. Trajectory Optimization Using MILP

A. Sequential decision process

Given the LTI-MA framework described above, our goal is to compute optimal trajectories between two waypoints corresponding to an initial and a final state. An optimal trajectory consists of a sequence of dynamically feasible states and corresponding inputs that satisfy the constraints imposed by the environment, and minimize a certain performance criterion captured by a cost function. The latter can be a measure of time or fuel, or a more complex criterion such as visibility or risk. The waypoints are typically provided by a higher level planning algorithm that optimizes a specific task or mission criterion.[14] In this paper, we will assume that such a higher planning level is in place.

The trajectory optimization problem can be viewed as a sequential decision process, where at the start of each decision step, the helicopter is flying in one of the LTI modes. The guidance problem then comes down to deciding at each decision step whether to stay in the current LTI mode l, to transition to a neighboring mode l', or to execute a certain maneuver m. However, the last option is only available if the entry conditions $\mathcal{X}_m$ for that maneuver are satisfied.

B. Mixed Integer Linear Programming

The guidance decision logic outlined above lends itself well to being formulated as a *mixed integer linear program* (MILP). MILP is a powerful mathematical programming framework that allows inclusion of integer variables and discrete logic in a continuous linear optimization problem.[15] It is commonly used in Operations Research,[16] and has more recently been introduced to the field of hybrid systems.[17] In our case, the continuous optimization is done over the states and inputs associated with the LTI modes. The discrete logic and decisions result from partitioning the state space into the distinct LTI modes and from the option of executing maneuvers when the corresponding entry conditions are satisfied.

As an illustration of how logical decisions can be incorporated in an optimization problem, consider the following example. Assume a cost function $J(\boldsymbol{x})$ needs to be minimized subject to either one of two

constraints $\ell_1(\boldsymbol{x})$ and $\ell_2(\boldsymbol{x})$ on the continuous decision vector $\boldsymbol{x}$:

$$\begin{array}{l} \min\limits_{\boldsymbol{x}} J(\boldsymbol{x}) \\ \text{subject to:} \\ \qquad \begin{array}{rcl} \ell_1(\boldsymbol{x}) & \leq & 0 \\ \text{OR} \quad \ell_2(\boldsymbol{x}) & \leq & 0 \end{array} \end{array} \tag{6}$$

By introducing a large positive number M and a binary variable b, this optimization problem can equivalently be formulated as follows:

$$\begin{array}{l} \min\limits_{\boldsymbol{x}} J(\boldsymbol{x}) \\ \text{subject to:} \\ \qquad \begin{array}{rcl} \ell_1(\boldsymbol{x}) & \leq & Mb \\ \text{AND} \quad \ell_2(\boldsymbol{x}) & \leq & M(1-b) \\ b & \in & \{0,1\} \end{array} \end{array} \tag{7}$$

When $b = 0$, constraint $\ell_1(\boldsymbol{x})$ must be satisfied, whereas $\ell_2(\boldsymbol{x})$ is relaxed. Namely, if M is chosen sufficiently large, $\ell_2(\boldsymbol{x}) \leq M(1-b)$ is always satisfied independent of the value of $\boldsymbol{x}$. The situation is reversed when $b = 1$. Since b can only take the binary values 0 or 1, at least one of the constraints $\ell_1(\boldsymbol{x})$ and $\ell_2(\boldsymbol{x})$ will be satisfied, which is equivalent to the original "OR"-formulation (6). In the special case where $J(\boldsymbol{x})$, $\ell_1(\boldsymbol{x})$ and $\ell_2(\boldsymbol{x})$ are (affine) linear expressions, problem (7) is a MILP.

The formulation can easily be extended to account for multiple constraints $\ell_k(\boldsymbol{x}),\ k = 1 \ldots K$, out of which at least N must be satisfied simultaneously. This is done as follows:

$$\begin{array}{l} \min\limits_{\boldsymbol{x}} J(\boldsymbol{x}) \\ \text{subject to:} \\ \qquad \begin{array}{rcl} \ell_k(\boldsymbol{x}) & \leq & Mb_k,\ k = 1 \ldots K \\ \sum\limits_k b_k & \leq & K - N \\ b_k & \in & \{0,1\} \end{array} \end{array} \tag{8}$$

The additional summation constraint ensures that at least N of the binary variables b_k are 0, thus guaranteeing that at least N of the inequalities $\ell_k(\boldsymbol{x}) \leq 0$ are satisfied simultaneously. More generally, using a vector $\boldsymbol{b}$ of binary variables, any polyhedron or intersection of polyhedra described by linear constraints on a continuous decision vector $\boldsymbol{x}$ can then be described as follows:

$$\mathbf{L}\boldsymbol{x} + \mathbf{M}\boldsymbol{b} + \mathbf{k} \leq 0 \tag{9}$$

The constant matrix $\mathbf{M}$ and vector $\mathbf{k}$ contain large numbers M and integer constants K and N that can encode any binary logic such as that of example (8).

C. Trajectory Optimization with the LTI-MA

We now apply the above constraint formulation principle to the trajectory planning problem using the LTI-MA dynamics.

1. Operating Region Bounds

We begin by introducing binary variables to capture the *and/or* logic in the convex and non-convex constraints (1), (2), and (5), describing the feasible state and input sets of the LTI modes and the feasible entry states of the maneuvers. They can be captured in the matrix form of inequality (9). As such, for each LTI mode l, we obtain:

$$\left\{ \begin{array}{rcl} \boldsymbol{x}_{i+1} & = & \mathbf{A}_l \boldsymbol{x}_i + \mathbf{B}_l \boldsymbol{u}_i \\ t_{i+1} & = & t_i + \Delta t \\ \mathbf{0} & \geq & \mathbf{F}_l^c \boldsymbol{x}_i + \mathbf{F}_l^b \boldsymbol{y}_{il} + \mathbf{f}_l \\ \mathbf{0} & \geq & \mathbf{G}_l^c \boldsymbol{u}_i + \mathbf{G}_l^b \boldsymbol{w}_{il} + \mathbf{g}_l \end{array} \right. \tag{10}$$

where $\boldsymbol{y}_{il}$ and $\boldsymbol{w}_{il}$ are binary vectors. The matrix $\mathbf{F}_l^c$ combines the coefficients $\boldsymbol{f}_{l,\cdot}^c$ and $\boldsymbol{f}_{l,\cdot}^n$ of the state inequalities (1), while matrix $\mathbf{F}_l^b$ and vector $\mathbf{f}_l$ encode the corresponding non-convexities using binary logic. Similarly, matrix $\mathbf{G}_l^c$ combines the coefficients $\boldsymbol{g}_{l,\cdot}^c$ and $\boldsymbol{g}_{l,\cdot}^n$ of the input inequalities (2), and matrix $\mathbf{G}_l^b$ and vector $\mathbf{g}_l$ encode the associated non-convexities.

For the maneuver entry conditions (5), the same principle yields:

$$\left\{\begin{array}{rcl} \boldsymbol{x}_{i+1} & = & \mathbf{C}_m\boldsymbol{x}_i + \mathbf{d}_m \\ t_{i+1} & = & t_i + \Delta T_m \\ \mathbf{0} & \geq & \mathbf{F}_m^c\boldsymbol{x}_i + \mathbf{F}_m^b\boldsymbol{z}_{im} + \mathbf{f}_m \end{array}\right. \tag{11}$$

where $\boldsymbol{z}_{im}$ is again a binary vector, $\mathbf{F}_m^c$ combines the coefficients $\boldsymbol{f}_{m,\cdot}^c$ and $\boldsymbol{f}_{m,\cdot}^n$, and $\mathbf{F}_m^b$ and $\mathbf{f}_m$ encode the non-convex structure of the entry constraints (5).

2. LTI Mode Switching

At the beginning of each decision step, the vehicle is in exactly one LTI mode l. Hence, with every decision step i, we can associate a binary variable b_{il} that equals 1 if the vehicle is flying in mode l at the start of that decision step. Since the L modes are mutually exclusive, only one b_{il} variable out of L can be 1 at each step i. The non-active LTI modes must then be "switched off". Using principle (8), this can be expressed as follows:

$$\forall l \in [1 \ldots L]: \left\{\begin{array}{rcl} \boldsymbol{x}_{i+1} - \mathbf{A}_l\boldsymbol{x}_i - \mathbf{B}_l\boldsymbol{u}_i & \leq & M(1-b_{il})\mathbf{1} \\ -\boldsymbol{x}_{i+1} + \mathbf{A}_l\boldsymbol{x}_i + \mathbf{B}_l\boldsymbol{u}_i & \leq & M(1-b_{il})\mathbf{1} \\ \mathbf{F}_l^c\boldsymbol{x}_i + \mathbf{F}_l^b\boldsymbol{y}_{il} + \mathbf{f}_l & \leq & M(1-b_{il})\mathbf{1} \\ \mathbf{G}_l^c\boldsymbol{u}_i + \mathbf{G}_l^b\boldsymbol{w}_{il} + \mathbf{g}_l & \leq & M(1-b_{il})\mathbf{1} \end{array}\right. \tag{12a}$$

$$\left\{\begin{array}{rcl} t_{i+1} - t_i - \Delta t & \leq & 0 \\ -t_{i+1} + t_i + \Delta t & \leq & 0 \\ \sum_{i=1}^{L} b_{il} & = & 1 \end{array}\right. \tag{12b}$$

where $\mathbf{1}$ represents a unity vector of appropriate length. Notice that the state space equality constraints of the LTI mode dynamics (10) have been replaced by a pair of positive and negative inequalities, such that they can be relaxed if the vehicle is not in mode l (i.e. when $b_{il} = 0$). Since the time evolution equation is the same for all LTI modes, however, the corresponding inequalities need not be relaxed.

3. Maneuver Execution

Similarly, we introduce a binary selection variable d_{im} for each maneuver m at each decision step i, which equals 1 if the maneuver is executed:

$$\forall m \in [1 \ldots P]: \left\{\begin{array}{rcl} \boldsymbol{x}_{i+1} - \mathbf{C}_m\boldsymbol{x}_i - \mathbf{d}_m & \leq & M(1-d_{im})\mathbf{1} \\ -\boldsymbol{x}_{i+1} + \mathbf{C}_m\boldsymbol{x}_i + \mathbf{d}_m & \leq & M(1-d_{im})\mathbf{1} \\ \mathbf{F}_m^c\boldsymbol{x}_i + \mathbf{F}_m^b\boldsymbol{z}_{im} + \mathbf{f}_m & \leq & M(1-d_{im})\mathbf{1} \\ t_{i+1} - t_i - \Delta T_m & \leq & M(1-d_{im}) \\ -t_{i+1} + t_i + \Delta T_m & \leq & M(1-d_{im}) \end{array}\right. \tag{13a}$$

$$\sum_{i=1}^{P} d_{im} \leq 1 \tag{13b}$$

The inequality $\sum_{m=1}^{P} d_{im} \leq 1$ ensures that at most one maneuver is performed at a time. This implies that if the initial conditions $\mathbf{F}_m^c\boldsymbol{x}_i + \mathbf{F}_m^b\boldsymbol{z}_{im} + \mathbf{f}_m \leq \mathbf{0}$ for a certain maneuver m are satisfied, it does not necessarily have to be executed: d_{im} can still be set to 0. However, if it is initiated, the state and time update inequalities of the LTI mode constraints (12a) and (12b) must be relaxed, since they are in that case

determined by equations (4). We therefore extend constraints (12a) and (12b) as follows:

$$\forall l \in [1 \dots L]: \begin{cases} \boldsymbol{x}_{i+1} - \mathbf{A}_l \boldsymbol{x}_i - \mathbf{B}_l \boldsymbol{u}_i & \leq & M(1-b_{il})\mathbf{1} + M(\sum_{m=1}^{P} d_{im})\mathbf{1} \\ -\boldsymbol{x}_{i+1} + \mathbf{A}_l \boldsymbol{x}_i + \mathbf{B}_l \boldsymbol{u}_i & \leq & M(1-b_{il})\mathbf{1} + M(\sum_{m=1}^{P} d_{im})\mathbf{1} \\ \mathbf{F}_l^c \boldsymbol{x}_i + \mathbf{F}_l^b \boldsymbol{y}_{il} + \mathbf{f}_l & \leq & M(1-b_{il})\mathbf{1} \\ \mathbf{G}_l^c \boldsymbol{u}_i + \mathbf{G}_l^b \boldsymbol{w}_{il} + \mathbf{g}_l & \leq & M(1-b_{il})\mathbf{1} \end{cases} \tag{14a}$$

$$\begin{array}{rcl} t_{i+1} - t_i - \Delta t & \leq & M \sum_{m=1}^{P} d_{im} \\ -t_{i+1} + t_i + \Delta t & \leq & M \sum_{m=1}^{P} d_{im} \\ \sum_{i=1}^{L} b_{il} & = & 1 \end{array} \tag{14b}$$

If no maneuver is performed, the extra relaxation term $M(\sum_{m=1}^{P} d_{im})\mathbf{1}$ will equal $\mathbf{0}$. Since all maneuvers start and end in one of the LTI modes, the LTI operating bounds need *not* be relaxed by the maneuver selection variables: even if a maneuver is executed, the vehicle will still be in some LTI mode l at the start of decision step i.

D. Planning Strategies

Given an initial state $\boldsymbol{x}_{init}$, we now want to compute an optimal trajectory to a desired final state $\boldsymbol{x}_f$. These states are typically positions with associated speeds and heading angles provided in an inertial coordinate frame. Depending on the operational objective and the available computational resources, we can consider different planning strategies.

1. *Fixed Horizon Planning*

A first approach is to solve one (off-line) MILP problem over a fixed number of decision steps, with the constraint that the vehicle must have arrived within some bound of the final state by the end of the planning horizon. This approach requires that all relevant information about the mission and environment is known ahead of time, and that during execution of the plan, the trajectory can be accurately tracked. Introducing a planning horizon of T decision steps and an arrival tolerance $\boldsymbol{\epsilon}$, the optimization problem has the following general form:

$$\min J_{fh} = \sum_{i=0}^{T-1} \left(\ell_i(\boldsymbol{x}_i, \boldsymbol{u}_i, \boldsymbol{b}_i) + \sum_{l=1}^{L} b_{il}\ell_{il} + \sum_{m=1}^{P} d_{im}\ell_{im} \right) \tag{15a}$$

$$\text{subject to} \begin{cases} (13a), (13b), \ i = 0 \dots T-1 \\ (14a), (14b), \ i = 0 \dots T-1 \\ \boldsymbol{x}_T - \boldsymbol{x}_f \leq \boldsymbol{\epsilon} \\ -\boldsymbol{x}_T + \boldsymbol{x}_f \leq \boldsymbol{\epsilon} \\ \boldsymbol{x}_0 = \boldsymbol{x}_{init} \end{cases} \tag{15b}$$

Here $\ell_i(\boldsymbol{x}_i, \boldsymbol{u}_i, \boldsymbol{b}_i)$ is a general (piece-wise) linear cost term associated with the ith decision step, with $\boldsymbol{b}_i$ some binary decision vector; ℓ_{il} and ℓ_{im} are constant cost factors, respectively associated with being in LTI mode l or executing maneuver m during decision step i. For the problem to be feasible, the horizon length T must be an upper bound on the number of steps needed to reach the desired state.

2. *Shortest Arrival Time*

A special case of a fixed horizon planning problem is to compute the shortest time trajectory between the states $\boldsymbol{x}_{init}$ and $\boldsymbol{x}_f$. The shortest time corresponds to the minimum weighted number of decision steps in which the final state can be reached. The weight of each step is the actual duration of the action taken during that step. To minimize the arrival time, we can then introduce binary variables r_i that select the step at which the final state is reached. In addition, an extra equality constraint is needed that enforces the helicopter to actually reach the desired state at one of the decision steps in the planning horizon.[18]

We again consider an horizon of T decision steps and an arrival treshold $\boldsymbol{\epsilon}$. A shortest time trajectory can then be computed as follows:

$$\min J_{st} = \sum_{i=0}^{T} r_i i\,\Delta t + \sum_{i=0}^{T-1}\sum_{m=1}^{P} d_{im}(\Delta T_m - \Delta t) \tag{16a}$$

$$\text{subject to} \quad \begin{cases} (13a), (13b),\ i = 0 \dots T-1 \\ (14a), (14b),\ i = 0 \dots T-1 \\ \boldsymbol{x}_i - \boldsymbol{x}_f \leq \boldsymbol{\epsilon} + M(1-r_i)\mathbf{1},\ i = 0 \dots T \\ -\boldsymbol{x}_i + \boldsymbol{x}_f \leq \boldsymbol{\epsilon} + M(1-r_i)\mathbf{1},\ i = 0 \dots T \\ \sum_{i=0}^{T} r_i = 1 \\ \boldsymbol{x}_0 = \boldsymbol{x}_{init} \end{cases} \tag{16b}$$

Since only one of the r_i binary variables equals 1, the first term in the cost function yields the optimal number of decision steps needed to reach the final state, weighted by Δt. If the optimal action at the ith decision step is to stay in the LTI regime, the weight of the step corresponds to the discretization step Δt. However, if the optimal action is to perform a maneuver ($d_{im} = 1$), the weight of the decision step is the maneuver duration ΔT_m. Since the first term in J_{st} already accounts for a weight Δt, the latter is subtracted from the actual maneuver time ΔT_m.

3. Receding Horizon Planning

A drawback of MILP is that the computation time increases at least polynomially with the number of variables and constraints. Therefore, the fixed horizon approach is mainly suited for offline computation of trajectories. For real-time applications, it can only be applied to relatively small problems, i.e. to problems with a reduced set of LTI modes and maneuvers and/or a limited number of decision steps in which the final state can be reached.

Offline trajectory planning, however, has several disadvantages. First, once the trajectory has been computed, it does not allow for changes in the vehicle dynamics or for modifications in the environment during execution of the plan. As such, a nominal offline planning strategy is not robust to uncertainties or disturbances. Moreover, all necessary information has to be available beforehand, i.e. before the vehicle starts its mission. This is often impossible and would preclude applications such as reconnaissance or terrain exploration. In addition, it is our intention to use the LTI-MA framework in a *real-time* guidance loop, where the full range of the vehicle's dynamic capabilities can be exploited in a reactive fashion.

The above limitations can be effectively addressed using a receding horizon planning strategy.[8,19] The path of the vehicle is then computed iteratively and composed of a sequence of locally (time-)optimal segments. At a certain iteration, a MILP with an appropriate cost function is solved online for T future steps, thus providing the reference trajectory and actions for the next T steps. 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 (e.g. as resulting from onboard sensor information). However, only a subset of these T actions is actually implemented. The process is then repeated periodically, and a new set of commands is computed for each time window. As such, new information about the environment can be incorporated in the optimization problem at each iteration, enabling reactive planning capabilities that are crucial when the environment changes or is explored in real-time.

If the length of the planning horizon is relatively short and the problem complexity low enough to solve the fixed horizon problem (15a)-(15b) or (16a)-(16b) in real-time, a first approach to a receding horizon implementation is to solve a fixed horizon problem at each time step. As the vehicle moves closer to the goal, the upper bound T on the number of required decision steps can then be reduced at each iteration. If the waypoints are relatively far apart, however, and the horizon length is consequently relatively long, this method becomes computationally too expensive for use in real-time.

Therefore, at a certain iteration with current state $\boldsymbol{x}_{init}$, we use the following, more general form for the

receding horizon optimization problem:

$$\min J_{rh} = \sum_{i=0}^{T-1} \left(\ell_i(\boldsymbol{x}_i, \boldsymbol{u}_i, \boldsymbol{b}_i) + \sum_{l=1}^{L} b_{il}\ell_{il} + \sum_{m=1}^{P} d_{im}\ell_{im} \right) + \ell_T(\boldsymbol{x}_T, \boldsymbol{x}_f, \boldsymbol{b}_T) \tag{17a}$$

$$\text{subject to} \begin{cases} (13a), (13b), \ i = 0 \dots T-1 \\ (14a), (14b), \ i = 0 \dots T-1 \\ \boldsymbol{x}_0 = \boldsymbol{x}_{init} \end{cases} \tag{17b}$$

Since the final state $\boldsymbol{x}_f$ might not be reached within T decision steps from the current state $\boldsymbol{x}_{init}$, the arrival constraints are replaced by a terminal cost $\ell_T(\boldsymbol{x}_T, \boldsymbol{x}_f, \boldsymbol{b}_T)$ in the objective function. According to Bellman's principle of optimality,[20] the ideal terminal cost is the exact cost-to-go (e.g. time-to-go) from the last state $\boldsymbol{x}_T$ in the planning horizon to the desired state $\boldsymbol{x}_f$. However, computing the exact cost-to-go generally requires solving a fixed horizon problem from the last state in the planning horizon, thus defeating the benefits of using a receding horizon policy. Instead, heuristic methods are often used to generate estimates of the cost-to-go that guarantee some form of stability (e.g. reaching the goal without getting trapped in local minima of the cost function) and some level of performance (e.g. bounds on suboptimality).[19,21] Notice, however, that to be applicable to the MILP framework, these terminal cost estimates must have a piece-wise linear form.

IV. Small-Scale Helicopter Example

We now apply the mathematical framework from section III to a small-scale rotorcraft modeled after MIT's aerobatic X-Cell helicopter.[22] To keep the formulation concise, we present an approximate inertial frame model. A physically more comprehensive approach using body-fixed velocities can be found in Ref. 23, in which additional binary logic is introduced to encode the associated nonlinear kinematics.

A. Helicopter LTI Modes

1. *Continuous Time Version*

The velocity control augmentation system of the X-Cell is described in Ref. 3 and features the following command variables: body axis forward velocity u_{cmd} and side velocity v_{cmd}, climb rate $\dot{h}_{cmd}$ and yaw rate r_{cmd}. The yaw rate command is mechanized to work as a turn rate command, both at hover and forward flight. In hover, the helicopter uses tail rotor control to turn on the spot. In forward flight, lateral cyclic and tail rotor control are mixed by the control law to achieve coordinated turns, i.e., turns with zero side slip. The control system uses gain scheduling to linearize the closed-loop dynamics around several operating velocities, resulting in different LTI modes. As such, the closed-loop dynamics of the X-Cell in each LTI mode l can be accurately modeled by the following decoupled, first-order differential equations:[24]

$$\begin{aligned} \dot{u} &= -\tfrac{1}{\tau_{u_l}} u + \tfrac{1}{\tau_{u_l}} u_{cmd} \\ \dot{v} &= -\tfrac{1}{\tau_{v_l}} v + \tfrac{1}{\tau_{v_l}} v_{cmd} \\ \ddot{h} &= -\tfrac{1}{\tau_{h_l}} \dot{h} + \tfrac{1}{\tau_{h_l}} \dot{h}_{cmd} \\ \dot{r} &= -\tfrac{1}{\tau_{r_l}} r + \tfrac{1}{\tau_{r_l}} r_{cmd} \end{aligned} \tag{18}$$

in which u is the body-fixed forward velocity, v is the body-fixed lateral or "side slip" velocity, $\dot{h}$ is the climb rate, and r is the turn rate. Each operating region l is characterized by specific time constants $\tau_{._l}$ and limits on speed, acceleration and control variables. For example, the turn rate response is quicker in hover than in forward flight, and side slip velocities are not permitted in cruise mode. Figure 4 shows a graphical example of this partitioning: the three principal regions are 1) the hover mode, where side slip and turns on the spot are allowed, 2) the transition mode, where a small amount of side slip is tolerated, and 3) the cruise mode, where turns are fully coordinated.

Unfortunately, the inertial kinematics associated with this model are nonlinear. We therefore approximate the body-fixed dynamics (18) by the following model in an inertial (x, y, z) coordinate frame, in which x and

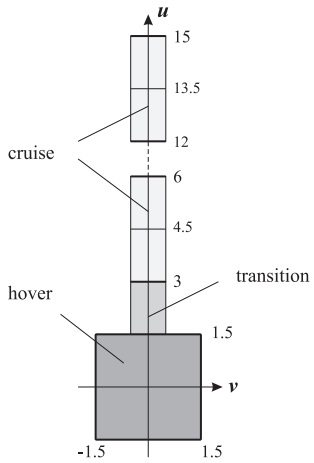

Figure 4. Example of operating regions of the velocity control mode, shown in function of the body axis forward (u) and lateral velocity (v).

y are the coordinates in the horizontal plane and z is the altitude:

$$\begin{aligned} \ddot{x}(t) &= -\tfrac{1}{\tau_l}\dot{x}(t) + \tfrac{k_l}{\tau_l}\dot{x}_{cmd}(t) \\ \ddot{y}(t) &= -\tfrac{1}{\tau_l}\dot{y}(t) + \tfrac{k_l}{\tau_l}\dot{y}_{cmd}(t) \\ \ddot{z}(t) &= -\tfrac{1}{\tau_z}\dot{z}(t) + \tfrac{k_z}{\tau_z}\dot{z}_{cmd}(t) \end{aligned} \tag{19}$$

Each LTI mode l is characterized by a time constant τ_l and gain k_l associated with the planar motion. For the vertical motion, we consider the same time constant τ_z and gain k_z for all modes, indicating that there is only one climb/descent mode which is fully decoupled from the horizontal motion. In state space form $\dot{\boldsymbol{x}}(t) = \mathbf{A_{cl}}\boldsymbol{x}(t) + \mathbf{B_{cl}}\boldsymbol{u}(t)$, we obtain:

$$\begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{z}(t) \\ \ddot{x}(t) \\ \ddot{y}(t) \\ \ddot{z}(t) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -\frac{1}{\tau_l} & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{\tau_l} & 0 \\ 0 & 0 & 0 & 0 & 0 & -\frac{1}{\tau_z} \end{bmatrix} \begin{bmatrix} x(t) \\ y(t) \\ z(t) \\ \dot{x}(t) \\ \dot{y}(t) \\ \dot{z}(t) \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{k_l}{\tau_l} & 0 & 0 \\ 0 & \frac{k_l}{\tau_l} & 0 \\ 0 & 0 & \frac{k_z}{\tau_z} \end{bmatrix} \begin{bmatrix} \dot{x}_{cmd}(t) \\ \dot{y}_{cmd}(t) \\ \dot{z}_{cmd}(t) \end{bmatrix} \tag{20}$$

The input vector $\boldsymbol{u}$ now contains inertial reference speeds $\dot{x}_{cmd}$, $\dot{y}_{cmd}$, and $\dot{z}_{cmd}$, instead of body-fixed velocities; the state vector $\boldsymbol{x}$ is composed of the inertial position vector $[x\ y\ z]^T$ and velocity vector $[\dot{x}\ \dot{y}\ \dot{z}]^T$. Accordingly, after discretizing time, the control sequence corresponding to a trajectory will consist of inertial reference velocities that can be transformed into equivalent body-fixed velocity commands. Alternatively, the resulting inertial positions can be given as inputs to a waypoint tracking controller.

Notice, however, that the planar dynamics are isotropic in both coordinates and ignore limits on side slip and turn rate. To correct for this and distinguish between the different operating regions, we introduce additional constraints $\boldsymbol{x} \in \mathcal{X}$ and $\boldsymbol{u} \in \mathcal{U}$ on the state and input vectors respectively. First, the different LTI modes are scheduled with the magnitude of the planar inertial velocity vector $\boldsymbol{v} = [\dot{x}\ \dot{y}]^T$. Each operating region l is thus delimited by a minimum speed $v_{min,l}$ and a maximum speed $v_{max,l}$:

$$v_{min,l} \leq ||\boldsymbol{v}|| \leq v_{max,l} \tag{21}$$

where for the hover mode ($l = 1$), we have $v_{min,1} = 0$. The magnitude of the horizontal velocity command vector $\boldsymbol{v}_{cmd} = [\dot{x}_{cmd}\ \dot{y}_{cmd}]^T$, however, is the same for all modes:

$$0 \leq ||\boldsymbol{v}_{cmd}|| \leq v_{max} \tag{22}$$

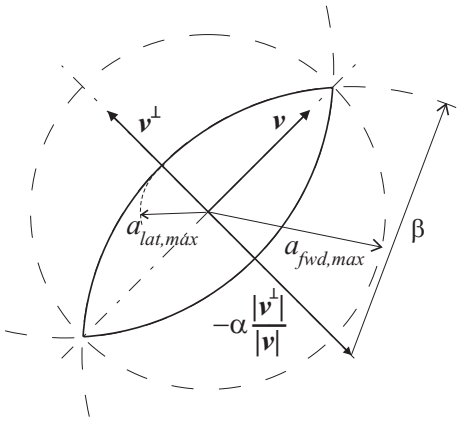

Figure 5. Two circle approximation of the elliptic constraint on forward and lateral acceleration.

where v_{max} is the overall maximum velocity of the helicopter. Similarly, the bounds on the climb rate command $\dot{z}_{cmd}$ and the resulting acceleration $\ddot{z}$ are identical for all LTI modes:

$$\dot{z}_{min} \leq \dot{z}_{cmd} \leq \dot{z}_{max} \tag{23a}$$

$$\ddot{z}_{min} \leq \ddot{z} \leq \ddot{z}_{max} \tag{23b}$$

where $\dot{z}_{min}$ and $\ddot{z}_{min}$ are negative numbers corresponding to descent.

Second, each LTI mode is characterized by specific bounds on turn rate and side slip. In hover mode, quick turns are possible and the limits on forward and lateral acceleration are of similar magnitude. In cruise mode, however, the helicopter flies in an airplane-like fashion in which no side slip is permitted and the achievable turn rate is much lower. The latter is then inversely proportional to the forward velocity. Both slide slip and turn rate constraints, however, can be captured by one mode-dependent geometric profile in which the horizontal inertial acceleration vector $\boldsymbol{a} = [\ddot{x} \;\; \ddot{y}]^T$ must lie.

A reasonably good approximation is to consider elliptical areas that – in the most general case– are delimited along their principal axes by mode-dependent maximum forward and lateral accelerations ($a_{fwd,max,l}$ and $a_{lat,max,l}$). For the hover mode, this ellipse will be approximately circular. In cruise, however, $a_{fwd,max,l}$ is typically larger than $a_{lat,max,l}$, corresponding to the fact that a helicopter can typically accelerate or decelerate faster than it can turn. This results in a long and narrow ellipse. Assuming coordinated turns, the ellipse should at all times be aligned with the planar velocity vector $\boldsymbol{v}$, such that it tracks changes in heading. This can be achieved by approximating the ellipse as the intersection of two circles whose centers lie along the line that goes through the origin and is parallel to the orthogonal complement $\boldsymbol{v}^\perp = [-\dot{y} \;\; \dot{x}]^T$ of $\boldsymbol{v}$. The geometric construction is illustrated in figure 5 for the cruise flight case $a_{fwd,max,l} \geq a_{lat,max,l}$. Using appropriate parameters α_l and β_l for each LTI mode l, these circles can be formulated as:

$$||\boldsymbol{a} - \alpha_l \frac{\boldsymbol{v}^\perp}{||\boldsymbol{v}||}|| \leq \beta_l \tag{24a}$$

$$||\boldsymbol{a} + \alpha_l \frac{\boldsymbol{v}^\perp}{||\boldsymbol{v}||}|| \leq \beta_l \tag{24b}$$

where α_l and β_l are determined by the values of $a_{fwd,max,l}$ and $a_{lat,max,l}$ as follows:

$$\alpha_l = \frac{a_{fwd,max,l}^2 - a_{lat,max,l}^2}{2a_{lat,max,l}} \tag{25a}$$

$$\beta_l = \sqrt{\left(\frac{a_{fwd,max,l}^2 - a_{lat,max,l}^2}{2a_{lat,max,l}}\right)^2 + a_{fwd,max,l}^2} = \sqrt{\alpha^2 + a_{fwd,max,l}^2} \tag{25b}$$

In case $a_{fwd,max,l} \leq a_{lat,max,l}$, $\boldsymbol{v}^\perp$ should be replaced by $\boldsymbol{v}$ in equalities (24a-b) and the role of $a_{fwd,max,l}$ and $a_{lat,max,l}$ be interchanged in equations (25a-b). When $a_{fwd,max,l} = a_{lat,max,l}$, the previous expressions result in a circular acceleration profile.

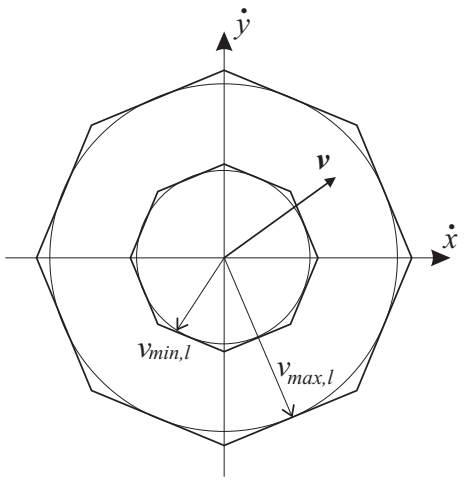

Figure 6. Approximation of minimum and maximum velocity bounds by polygons.

2. Discrete Time Version

To make use of the above dynamics in our MILP framework, the state space models (20) need to be discretized. Using the bilinear transform with a sample time Δt, the discrete state space model for each LTI mode l becomes: $\boldsymbol{x}_{i+1} = \mathbf{A}_l \boldsymbol{x}_i + \mathbf{B}_l \boldsymbol{u}_i$, with

$$\begin{array}{rcl} \mathbf{A}_l & = & \left(\mathbf{I} - \frac{\Delta t}{2}\mathbf{A}_{\mathbf{c}l}\right)^{-1}\left(\mathbf{I} + \frac{\Delta t}{2}\mathbf{A}_{\mathbf{c}l}\right) \\ \mathbf{B}_l & = & \left(\mathbf{I} - \frac{\Delta t}{2}\mathbf{A}_{\mathbf{c}l}\right)^{-1}\mathbf{B}_{\mathbf{c}l}. \end{array} \tag{26}$$

Next, as depicted in figure 6, all nonlinear inequality constraints characterizing each LTI mode l are approximated by N-sided polygons. The upper bounds of the velocity constraints (21)-(22) then correspond to the vectors lying inside such a polygon, which can be expressed as follows:

$$\forall n \in [1 \ldots N]: \qquad \dot{x}_i \sin\left(\frac{2\pi n}{N}\right) + \dot{y}_i \cos\left(\frac{2\pi n}{N}\right) \leq v_{max,l} \tag{27}$$

$$\dot{x}_{cmd,i} \sin\left(\frac{2\pi n}{N}\right) + \dot{y}_{cmd,i} \cos\left(\frac{2\pi n}{N}\right) \leq v_{max} \tag{28}$$

The minimum velocity $v_{min,l}$ delimiting mode l from below in constraint (21) can be handled by ensuring that the speed vector lies *outside* the corresponding polygon. By introducing N binary variables c_{iln} and applying principle (8), we can encode this non-convex constraint as follows:

$$\forall n \in [1 \ldots N]: \quad \dot{x}_i \sin\left(\frac{2\pi n}{N}\right) + \dot{y}_i \cos\left(\frac{2\pi n}{N}\right) \geq v_{min,l} - M c_{iln} \tag{29a}$$

$$\sum_{n=1}^{N} c_{iln} \leq N - 1 \tag{29b}$$

where M is again a sufficiently large number.

For the climb and descent acceleration constraints (23b), we use an Euler discretization of the derivative:

$$\ddot{z}_{min} \leq \frac{\dot{z}_{i+1} - \dot{z}_i}{\Delta t} \leq \ddot{z}_{max} \tag{30}$$

Applying the same principle and the polygonal approximation to the planar inertial constraints (24a-b), we obtain:

$$\forall n \in [1 \ldots N]: \quad \left(\frac{\dot{x}_{i+1} - \dot{x}_i}{\Delta t} + \alpha_l \frac{\dot{y}_i}{v_0}\right) \sin\left(\frac{2\pi n}{N}\right) + \left(\frac{\dot{y}_{i+1} - \dot{y}_i}{\Delta t} - \alpha_l \frac{\dot{x}_i}{v_0}\right) \cos\left(\frac{2\pi n}{N}\right) \leq \beta_l \tag{31a}$$

$$\left(\frac{\dot{x}_{i+1} - \dot{x}_i}{\Delta t} - \alpha_l \frac{\dot{y}_i}{v_0}\right) \sin\left(\frac{2\pi n}{N}\right) + \left(\frac{\dot{y}_{i+1} - \dot{y}_i}{\Delta t} + \alpha_l \frac{\dot{x}_i}{v_0}\right) \cos\left(\frac{2\pi n}{N}\right) \leq \beta_l \tag{31b}$$

Because these constraints will typically be formulated over a receding planning horizon T, the unknown magnitude of the velocity vector $\boldsymbol{v}_i = [\dot{x}_i \ \dot{y}_i]^T$ is replaced by v_0, the *known* horizontal speed at the initial time step in the horizon (except when $v_0 = 0$). A a result, when $||\boldsymbol{v}_i|| > v_0$ the circles will lie further away from each other, thus reducing the feasible region and corresponding maximum forward and lateral acceleration. On the other hand, if $||\boldsymbol{v}_i|| < v_0$, the circles will move closer together, thereby increasing the allowable accelerations. However, since only the action corresponding to the first decision step will be implemented (at which the acceleration constraints are exact), this under- or overestimation of the available acceleration in the remaining steps will be compensated for at the next iteration. In the fixed arrival time case, v_0 could be replaced by v_l, the forward velocity around which the dynamics are linearized in mode l (except for hover, for which $v_l = 0$).

Still, if this reshaping of the acceleration profile increases the bounds on forward or lateral acceleration by too much, the following magnitude constraint $||\boldsymbol{a}|| \leq \max(a_{fwd,max,l}, a_{lat,max,l})$ can be added:

$$\forall n \in [1 \ldots N]: \quad \left(\frac{\dot{x}_{i+1} - \dot{x}_i}{\Delta t}\right) \sin\left(\frac{2\pi n}{N}\right) + \left(\frac{\dot{y}_{i+1} - \dot{y}_i}{\Delta t}\right) \cos\left(\frac{2\pi n}{N}\right) \leq \max(a_{fwd,max,l}, a_{lat,max,l}) \quad (32)$$

B. Helicopter Maneuvers

Maneuvers are designed to exploit the extreme performance and agility of the vehicle: they typically take advantage of the full control input range and result in large state excursions. The availability of such maneuvers plays an essential role in reactive threat and obstacle avoidance. Table 1 gives an overview of maneuvers that could be designed; figure 7 shows the corresponding trajectories. Both split-S and hammerhead have already been implemented on MIT's helicopter.[12,25] Other maneuvers that have since been considered include dash, quick-stop (or deceleration) and quick-turn maneuvers. The split-S and hammerhead can be used to quickly reverse the direction of flight; compared to a level-flight U-turn, they are faster and require no lateral displacements. However, both require a minimum entry speed, whereas a U-turn can be performed at any velocity. Also, the split-S results in an altitude drop, while the hammerhead typically ends at the initial or a higher altitude.

Table 1. Description of sample maneuvers that could be implemented on a rotorcraft-type vehicle.

Maneuver	Usage
Dash to cruise	rapid acceleration from hover to one of the cruise conditions
Quick-stop	rapid transition from cruise to a full stop (hover)
Quick-turn	rapid turn resulting in a pre-determined heading change
Split-S	reversal of the flight direction with altitude loss
Hammerhead	reversal of the flight direction with altitude gain or zero altitude change

Each maneuver m is characterized by its duration ΔT_m, entry and exit speed $v_{in,m}$ and $v_{ex,m}$, resulting spatial displacement $[\Delta x_h \ \Delta y_h \ \Delta z]_m^T$ and change in heading $\Delta\psi_m$. As shown in figure 8, the displacement $[\Delta x_h \ \Delta y_h]_m^T$ and heading change $\Delta\psi_m$ are defined with respect to the body-fixed frame at the start of the maneuver. From these body frame parameters, a fixed affine transformation of the inertial state vector can be extracted for each maneuver. Assuming that a maneuver can only be initiated in level flight with $\dot{z}_i = 0$ and zero side slip, the characterizing constants derived from the body frame parameters are:

$$\begin{aligned}
\gamma_m &= -\arctan\left(\frac{\Delta x_{h,m}}{\Delta y_{h,m}}\right) \\
\delta_m &= -\Delta\psi_m \\
c_m &= \frac{\sqrt{(\Delta x_{h,m})^2 + (\Delta y_{h,m})^2}}{v_{init,m}} \\
d_m &= \frac{v_{ex,m}}{v_{in,m}}
\end{aligned}$$

with γ_m defined between $-180°$ and $180°$. The state transition resulting from maneuver m is then given by

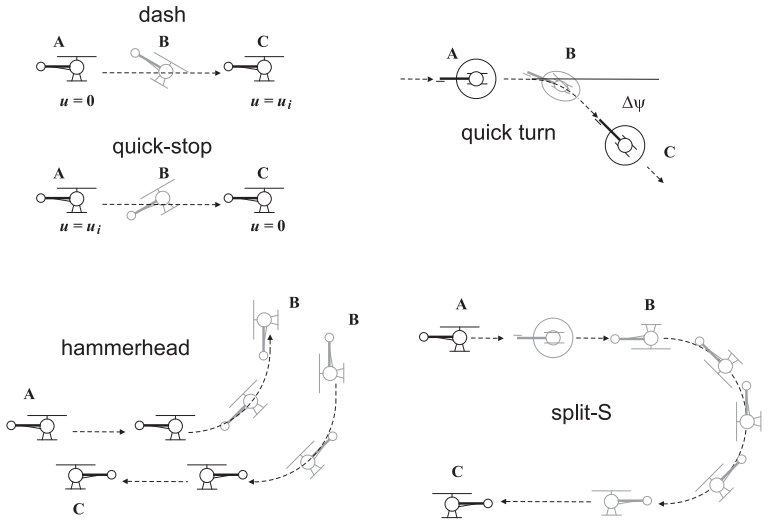

Figure 7. Helicopter trajectories for sample maneuvers.

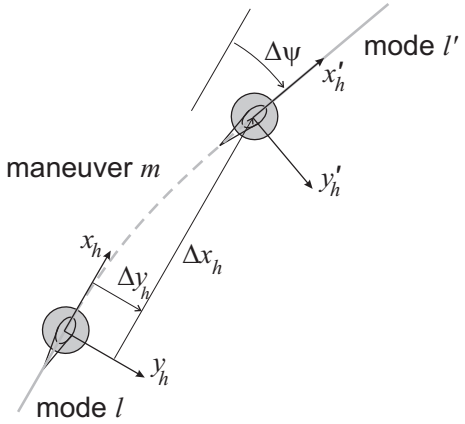

Figure 8. Change in helicopter position and orientation resulting from a maneuver (shown in dashed line), as observed from the body-fixed frame

the following affine transformation:

$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \\ z_{i+1} \\ \dot{x}_{i+1} \\ \dot{y}_{i+1} \\ \dot{z}_{i+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & c_m \cos\gamma_m & -c_m \sin\gamma_m & 0 \\ 0 & 1 & 0 & c_m \sin\gamma_m & c_m \cos\gamma_m & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & d_m \cos\delta_m & -d_m \sin\delta_m & 0 \\ 0 & 0 & 0 & d_m \sin\delta_m & d_m \cos\delta_m & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ z_i \\ \dot{x}_i \\ \dot{y}_i \\ \dot{z}_i \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \Delta z_m \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{33}$$

which combines rotation, scaling and translation to express the exit state $\boldsymbol{x}_{i+1}$ as a function of the entry state $\boldsymbol{x}_i$. The entry conditions $\boldsymbol{x}_i \in \mathcal{X}_m$ of maneuver m are a fixed planar initial speed $||\boldsymbol{v}_i|| = v_{in,m}$, zero

acceleration $\boldsymbol{a}_i$, and zero climb rate $\dot{z}_i$. The entry velocity constraint can be expressed as follows:

$$\forall n \in [1 \ldots N]: \quad \dot{x}_i \sin\left(\frac{2\pi n}{N}\right) + \dot{y}_i \cos\left(\frac{2\pi n}{N}\right) \leq v_{in,m} \tag{34a}$$

$$\dot{x}_i \sin\left(\frac{2\pi n}{N}\right) + \dot{y}_i \cos\left(\frac{2\pi n}{N}\right) \geq v_{in,m} - M c_{imn} \tag{34b}$$

$$\sum_{n=1}^{N} c_{imn} \leq N - 1 \tag{34c}$$

with c_{imn} binary variables. The acceleration and climb rate conditions become:

$$\begin{aligned} \dot{x}_i - \dot{x}_{i-1} &\leq 0 \\ -\dot{x}_i + \dot{x}_{i-1} &\leq 0 \\ \dot{y}_i - \dot{y}_{i-1} &\leq 0 \\ -\dot{y}_i + \dot{y}_{i-1} &\leq 0 \\ \dot{z}_i &\leq 0 \\ -\dot{z}_i &\leq 0 \end{aligned} \tag{35}$$

which are relaxed by $M(1 - \sum_{m=1}^{P} d_{im})$ if no maneuver is executed.

C. Obstacle Avoidance

By constraining the trajectory points to lie outside of polyhedral regions, obstacle avoidance can be handled by mixed integer linear constraints as well.[8] Although any obstacle shape approximated by polyhedrons can be dealt with, for simplicity of exposition, we consider rectangular obstacles k with lower left corner $(x_{min,k}, y_{min,k}, z_{min,k})$ and upper right corner $(x_{max,k}, y_{max,k}, z_{max,k})$. As is common practice in the field of robot motion planning, these obstacles are enlarged with the dimensions of the vehicle, such that the vehicle itself can be treated as a point.[26] Furthermore, only obstacles that lie within the maximum distance the helicopter can travel over the duration of the planning horizon should be accounted for. Denote the number of these relevant obstacles as S. Then, introducing binary variables f_{ikn} for each time step i and each obstacle k, the obstacle avoidance constraints can be expressed as follows:

$$\forall k \in [1 \ldots S]: \quad \begin{aligned} x_i &\leq x_{min,k} + M f_{ik1} \\ y_i &\leq y_{min,k} + M f_{ik2} \\ z_i &\leq z_{min,k} + M f_{ik3} \\ -x_i &\leq -x_{max,k} + M f_{ik4} \\ -y_i &\leq -y_{max,k} + M f_{ik5} \\ -z_i &\leq -z_{max,k} + M f_{ik5} \\ \sum_{n=1}^{6} f_{ikn} &\leq 5 \end{aligned} \tag{36}$$

These inequalities encode the requirement that each trajectory point (x_i, y_i, z_i) must lie in at least one of the outer halfplanes defined by the faces of the S obstacles.

Because the trajectory consists of a discrete sequence of positions, however, there is still no guarantee that the corresponding continuous path does not intersect with the obstacles between two subsequent points. Therefore, the obstacles must be extended by a safety boundary that corresponds to the largest distance that can be traveled during a decision step. If $v_{max}\Delta t$ is large or when maneuvers are considered, this safety boundary can be relatively big compared to the size of the obstacles, thus making the obstacle environment denser than might be acceptable.

However, if the vehicle is in the LTI regime, the boundary can be reduced by determining intermediate trajectory points through linear interpolation. For maneuver transitions, on the other hand, an additional avoidance check can be carried out for J sample points along the *actual* maneuver trajectory. Although these points are not known a priori, they can be expressed as affine transformations of the position vector

$\boldsymbol{p}_i = [x_i\ y_i\ z_i]^T$ and horizontal inertial velocity $\boldsymbol{v}_i = [\dot{x}_i\ \dot{y}_i]^T$ at the start of the maneuver. Let $\tilde{\boldsymbol{p}}_{mj}$ be the jth point along the trajectory of maneuver m. Similarly to equation (33), by introducing a maneuver-specific constant matrix $\mathbf{P}_{mj}$ and vector $\mathbf{p}_{mj}$ for each sample point $\tilde{\boldsymbol{p}}_{mj}$, we can write:

$$\tilde{\boldsymbol{p}}_{mj} = \mathbf{P}_{mj} \begin{bmatrix} \boldsymbol{p}_i \\ \boldsymbol{v}_i \end{bmatrix} + \mathbf{p}_{mj} \tag{37}$$

At each decision step i, the obstacle avoidance check for maneuver m and obstacle k can then be expressed as follows:

$$\begin{aligned} \forall k \in [1 \ldots S], j \in [1 \ldots J]: \qquad \mathbf{P}_{mj} \begin{bmatrix} \boldsymbol{p}_i \\ \boldsymbol{v}_i \end{bmatrix} + \mathbf{p}_{mj} &\leq \begin{bmatrix} x_{min,k} \\ y_{min,k} \\ z_{min,k} \end{bmatrix} + M \begin{bmatrix} f_{ijk1} \\ f_{ijk2} \\ f_{ijk3} \end{bmatrix} \\ -\mathbf{P}_{mj} \begin{bmatrix} \boldsymbol{p}_i \\ \boldsymbol{v}_i \end{bmatrix} + \mathbf{p}_{mj} &\leq -\begin{bmatrix} x_{max,k} \\ y_{max,k} \\ z_{max,k} \end{bmatrix} + M \begin{bmatrix} f_{ijk4} \\ f_{ijk5} \\ f_{ijk6} \end{bmatrix} \\ \sum_{n=1}^{6} f_{ijkn} &\leq 5 \end{aligned} \tag{38}$$

where the same binaries are used for each maneuver m. By adding terms $M(1 - d_{im})$, the constraints will be relaxed if the maneuver is not executed.

D. Receding Horizon Formulation

Given the above helicopter model, we are interested in guiding the helicopter between waypoints in the fastest possible way, thereby avoiding obstacles. The exact shortest time between two states can be computed using the fixed arrival time approach and objective function (16a). However, for real-time applications, we resort to a receding horizon strategy using the following heuristic piece-wise linear cost function. It aims at designing a *fast* trajectory between the initial waypoint $\boldsymbol{p}_{init} = \boldsymbol{p}_0$ in the horizon and the desired one $\boldsymbol{p}_f$, mimicking a shortest time objective:

$$\min J_h = \sum_{i=0}^{T} -q\boldsymbol{v}_i^T(\boldsymbol{p}_f - \boldsymbol{p}_{init}) + r|\boldsymbol{p}_T - \boldsymbol{p}_f| \tag{39}$$

The first term tries to maximize the scalar product of the inertial velocity with the vector that is pointing from the initial position to the desired one. The effect is twofold: it will speed the helicopter up, while turning it toward the right direction. In addition, the term $r|\boldsymbol{p}_T - \boldsymbol{p}_f|$ tries to minimize the 1-norm distance towards the goal from the last position in the planning horizon. Both terms thus work towards attaining the same goal, with q and r weight factors emphasizing either one effect.

Since the position error term only accounts for the final step of the planning horizon, the helicopter is allowed to speed up away from the goal to enable the execution of a maneuver that will eventually bring it closer than a regular coordinated turn would. This trade-off between acting immediately or "investing" in an overall more efficient maneuver is also captured by the scalar product. For example, consider a case in which the helicopter must reverse direction as quickly as possible, e.g. because of a pop-up threat ahead of it. Assume that it can either make a U-turn or perform a hammerhead. Since the latter requires a minimum entrance speed, depending on the initial velocity, one action might be faster than the other. First speeding up in a straight line to perform the hammerhead will generate scalar product terms that initially increase the objective function, but enable a sudden decrease when the maneuver is executed. Making a U-turn, on the other hand, involves only cost-decreasing terms, but might be slower overall. When there is a bound on lateral displacement, however, e.g. when flying through a street lined with buildings, the nominally fastest action might be infeasible. The MILP problem will then be optimized over the alternatives.

Using cost function (39), the full MILP problem that needs to be solved at each receding horizon iteration

is then:

$$\min J_h = \sum_{i=0}^{T} -q\boldsymbol{v}_i^T(\boldsymbol{p}_f - \boldsymbol{p}_0) + r|\boldsymbol{p}_T - \boldsymbol{p}_f| \tag{40}$$

$$\text{subject to} \begin{cases} \boldsymbol{x}_{i+1} - \mathbf{A}_l\boldsymbol{x}_i - \mathbf{B}_l\boldsymbol{u}_i \le M(1 - b_{il} + \sum_{m=1}^{P} d_{im})\mathbf{1} & l = 1\dots L,\ i = 0\dots T-1 \\ -\boldsymbol{x}_{i+1} + \mathbf{A}_l\boldsymbol{x}_i + \mathbf{B}_l\boldsymbol{u}_i \le M(1 - b_{il} + \sum_{m=1}^{P} d_{im})\mathbf{1} & l = 1\dots L,\ i = 0\dots T-1 \\ \boldsymbol{x}_{i+1} - \mathbf{C}_m\boldsymbol{x}_i - \mathbf{d}_m \le M(1 - d_{im})\mathbf{1} & m = 1\dots P,\ i = 0\dots T-1 \\ -\boldsymbol{x}_{i+1} + \mathbf{C}_m\boldsymbol{x}_i + \mathbf{d}_m \le M(1 - d_{im})\mathbf{1} & m = 1\dots P,\ i = 0\dots T-1 \\ \sum_{i=1}^{L} b_{il} = 1 & i = 0\dots T-1 \\ \sum_{i=1}^{P} d_{im} \le 1 & i = 0\dots T-1 \\ \boldsymbol{x}_0 = \boldsymbol{x}_{init} & \\ \\ (23a) & i = 0\dots T-1 \\ (27) + M(1 - b_{il}) & l = 1\dots L,\ i = 0\dots T-1 \\ (28) & i = 0\dots T-1 \\ (29a), (29b) + M(1 - b_{il}) & l = 1\dots L,\ i = 0\dots T-1 \\ (30) & i = 0\dots T-1 \\ (31a), (31b), (32) \ + M(1 - b_{il}) + M\sum_{m=1}^{P} d_{im} & l = 1\dots L,\ i = 0\dots T-1 \\ (34a), (34b), (34c) + M(1 - d_{im}) & m = 1\dots P,\ i = 0\dots T-1 \\ (35) + M(1 - \sum_{m=1}^{P} d_{im}) & i = 1\dots T-1 \\ (36) & i = 0\dots T \\ (38) + M(1 - d_{im}) & m = 1\dots P,\ i = 0\dots T-1 \end{cases}$$

V. Results

A. Helicopter and Problem Parameters

We now use the full receding horizon MILP formulation (40) to compute real-time reference trajectories for some example scenarios. The helicopter parameters for the LTI modes and maneuvers are based on those of MIT's X-Cell and are given in tables 2 and 3. We considered two LTI regimes: 1) a hover mode with forward and lateral velocities up to 3 m/s, and 2) a forward flight cruise mode up to 20 m/s, in which no side slip is allowed. The maximum forward acceleration for both modes was set to 3 m/s^2, the maximum lateral acceleration for hover and cruise to 3.14 m/s^2 and 3.49 m/s^2 respectively, corresponding to maximum turn rates of 60 deg/s and 10 deg/s at the limiting velocities of 3 m/s and 20 m/s.

Table 2. Parameters of LTI modes

Mode	v_{min} (m/s)	v_{max} (m/s)	$a_{fwd,max}$ (m/s^2)	$a_{lat,max}$ (m/s^2)	τ (s)	k
Hover	0	3	3	3.14	1	1
Cruise	3	20	3	3.49	1	1

The maneuver library contains a hammerhead and a split-S with entry speeds of 15 m/s and 18 m/s respectively. The body-fixed parameters describing them were obtained by averaging the actual values resulting from autonomous executions of these maneuvers on the X-Cell. Notice that an ideal execution would not exhibit a lateral displacement. However, since these imperfect maneuvers clearly illustrate the trade-off decisions the MILP optimization makes in the various scenarios, we chose to keep them. In addition, we focused on 2D trajectory planning and ignored the changes in altitude resulting from a maneuver, thus assuming that the helicopter is flying sufficiently high.

A planning horizon of $T = 5$ decision steps was used with $\Delta t = 1$ s. As such, each iteration of the planning problem had to be solved within a second. Using CPLEX 9.0 on a Pentium 4 with 2.2 Ghz clock frequency, optimal or good suboptimal feasible solutions could always be found within this hard real-time limit. The computation times ranged from 0.3 s to the full 1.0 s, but were about 0.6 s on average. The settings of

Table 3. Parameters of pre-programmed maneuvers

Maneuver	Δx_h(m)	Δy_h(m)	Δz (m)	$\Delta\psi$ (deg)	ΔT(s)	v_{in}(m/s)	v_{ex} (m/s)
Split-S	6.5	10	-30	180	5	15	18
Hammerhead	-20	-6	0	180	7	18	18

other parameters were as follows: $N = 16$ for the polygonal approximation of circular constraints, $J = 3$ for the obstacle avoidance checks between trajectory points, a 5 m safety boundary around the obstacles, and $q = r = 1$ for the weights in the cost function.

B. Scenarios

In each of the following scenarios, the helicopter is initially in the origin (0 m, 0 m), flying east at 6 m/s or 12 m/s through a corridor such as a street lined with buildings. It is then given the task to reverse its direction of flight and fly towards location (-100 m, 0 m) as quickly as possible. This command would typically be issued by a higher level decision unit, and could for example result from the detection of a threat ahead of the helicopter.

In the first scenario, depicted in figure 9, the vehicle is flying at 12 m/s. The optimal course of action is to speed up to 15 m/s and execute the split-S, resulting in a quick direction reversal with a 18 m/s exit speed. Conformable to the LTI-MA dynamics, the maneuver is considered as a discontinuous displacement, shown in dashed line. Upon exiting the split-S, the helicopter further accelerates to its maximum speed of 20 m/s. The full sequence to get near the waypoint at (-100 m, 0 m) takes 12 s. Solving once for the exact shortest time trajectory using strategy (16a)-(16b) yielded the same result. However, the computation took 310 s (for a $T = 14$ step horizon), making the approach inapplicable to online optimization.

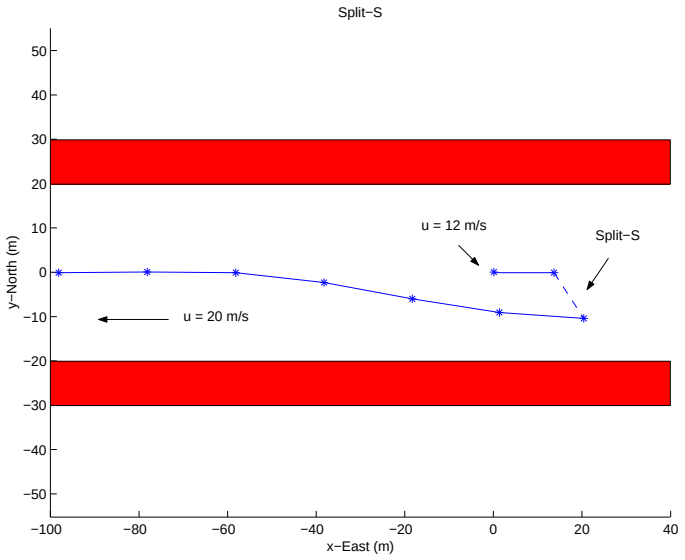

Figure 9. Scenario 1: The helicopter is initially in $(0, 0)$ flying at 12 m/s and needs to reverse direction towards $(-100, 0)$ as quickly as possible. It decides to execute the split-S.

In the second scenario, shown in figure 10, the initial speed is 6 m/s. The helicopter now decides to slow down to the hover mode in which it can turn around faster. It now takes 6 s to make the full U-turn and 10 s total to reach the waypoint at a speed of 20 m/s. The exact time-optimal trajectory looked very similar and lasted equally long, but took 471 s to compute (again using 14 decision steps as an upper bound).

Next, we considered two scenarios in which an additional obstacle was placed in the environment. The resulting trajectories are plotted in figures 11 and 12, for the two starting velocities of 12 m/s and 6 m/s respectively. Because of the restrictive bound on turn rate in the cruise mode, executing the split-S as in

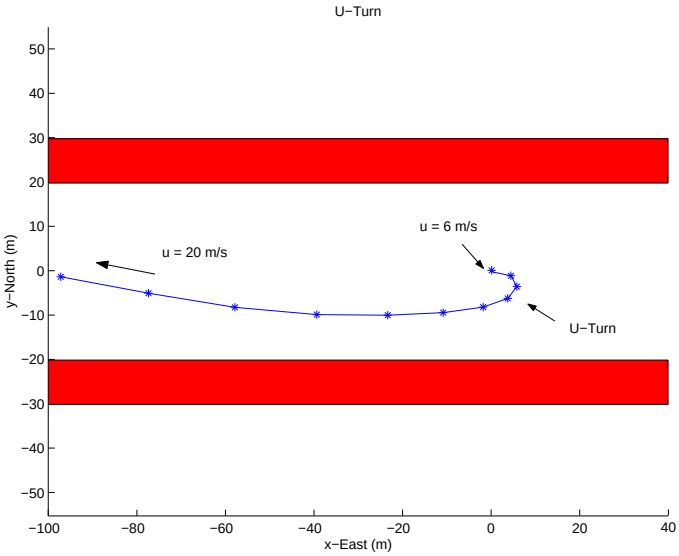

Figure 10. Scenario 2: The helicopter is initially in $(\mathbf{0}, \mathbf{0})$ flying at 6 m/s and needs to reverse direction towards $(-\mathbf{100}, \mathbf{0})$ as quickly as possible. It decides to make a U-turn.

scenario 1 would prohibit the vehicle from safely avoiding the obstacle (which is enlarged with a 5 m safety boundary): the helicopter would be facing it with a speed of 18 m/s and insufficient distance to turn away. The trajectory optimization therefore results in the vehicle changing heading during the first time step. That way it is not pointing towards the obstacle after executing the split-S, and has enough space to turn around it. In addition, the helicopter slows down to avoid the southern wall. In the U-turn case of figure 4, the vehicle simply makes a wider coordinated turn.

In the last two examples, shown in figures 13 and 14, we narrowed the width of the corridor to 20 m and 10 m respectively. The helicopter starts at 12 m/s in both cases. In the 20 m scenario, the vehicle first turns northeast such that it can still safely execute the split-S. In the 10 m scenario, however, because of the 5 m safety boundaries along the walls, the only feasible option is to slow down and make a turn on the spot.

Notice that in the above examples, the vehicle was not constrained to hit the waypoint exactly. If desired, however, this could be achieved by switching to a different cost function and/or by including additional arrival constraints as soon as the waypoint is reachable within the planning horizon. The option of changing cost functions between two iterations leads us to the discussion in the next section about different planning modes.

VI. Practical Considerations

When planning in a receding horizon fashion, it may be possible to take some decisions outside of the actual trajectory optimization problem. Decoupling specific elements of the planning problem by introducing a decision hierarchy can save significant computation time. This can be performed by tailoring the formulation of the trajectory optimization to the requirements set by a specific context. In some situations, it may indeed be possible to recognize *a priori* certain features of the planning problem based on the vehicle's operating condition and aspects of the environment. For example, in scenario 6 from above, the split-S could be ruled out as a feasible option and removed from the problem formulation ahead of time. More generally, if it is possible to determine whether the vehicle state and environment preclude certain types of maneuvers or LTI conditions, a planning mode with a pruned set of maneuvers and LTI modes can be used, thus reducing the problem complexity and required computation time. Furthermore, an appropriate discretization level can be used, depending on whether the context requires less or more precise trajectories.

These considerations are of importance in the implementation of a real-time guidance system. Exploiting a priori knowledge can also be used in the computation of a heuristic cost-to-go function. We believe that the most effective implementation will be one in which an online receding horizon approach is combined

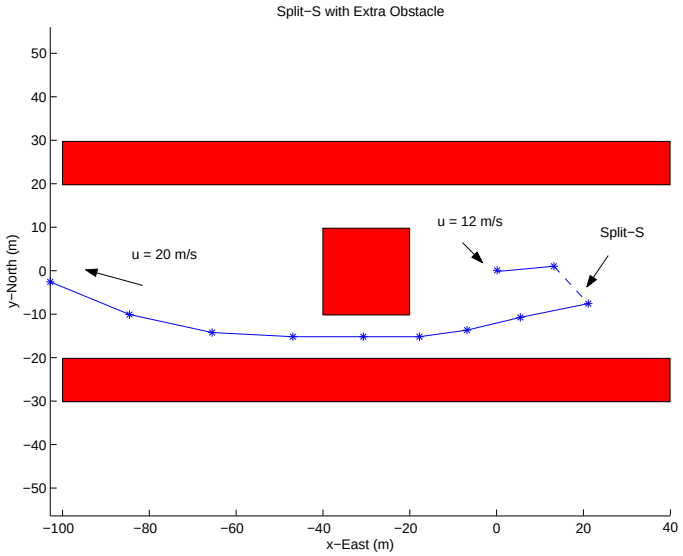

Figure 11. Scenario 3: The helicopter is initially in $(0, 0)$ flying at 12 m/s and needs to reverse direction towards $(-100, 0)$ as quickly as possible. To avoid the obstacle in the middle, it decides to change heading before executing the split-S.

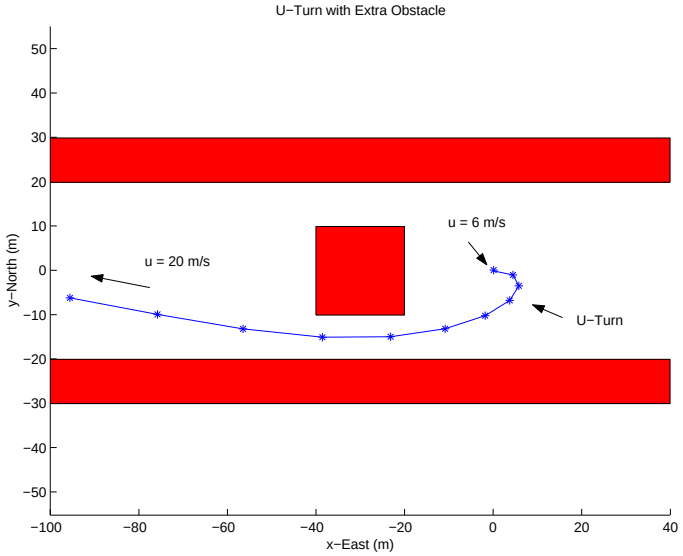

Figure 12. Scenario 4: The helicopter is initially in $(0, 0)$ flying at 6 m/s and needs to reverse direction towards $(-100, 0)$ as quickly as possible. To avoid the obstacle in the middle, it decides to make wider a U-turn.

with an offline cost-to-go calculation. Several (parameterized) cost-to-go functions could be pre-computed for different, often recurring situations, such as turning around a street corner in an urban environment. These cost-to-go functions can then be incorporated in the online optimization problem as terminal costs at the end of the planning horizon.

Such different planning modes would also allow to switch between inertial and body-fixed frame models of the vehicle. An inertial model, for instance, could be used to compute "high resolution" waypoints in a cluttered environment, which are then followed by solving MILP problems in a body-fixed reference frame. The MILP optimization itself could then act as the waypoint controller.

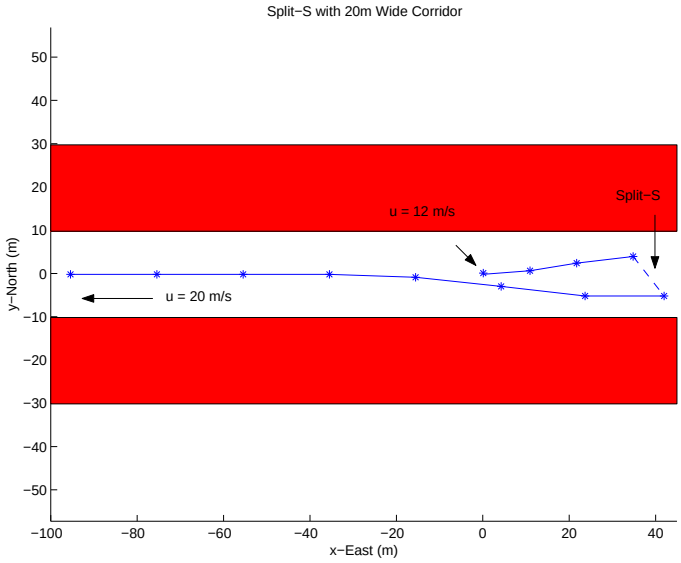

Figure 13. Scenario 5: The helicopter is initially in $(0, 0)$ flying at 12 m/s and needs to reverse direction towards $(-100, 0)$ as quickly as possible. Because of the narrower corridor, the vehicle first turns northeast before executing the split-S.

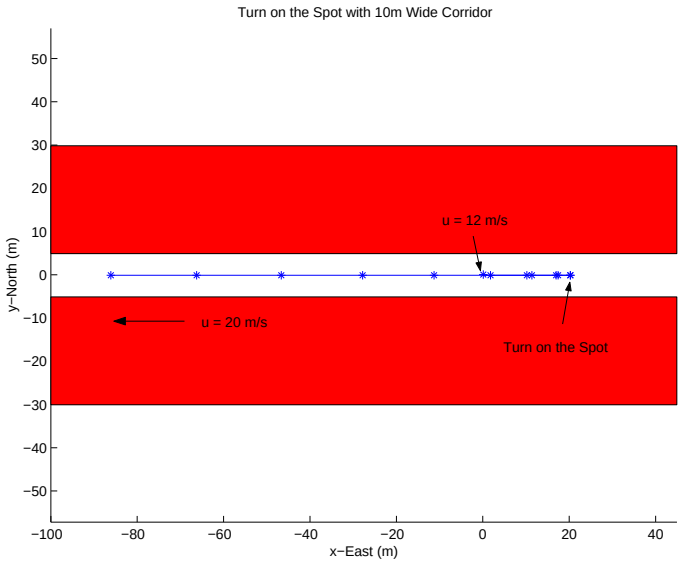

Figure 14. Scenario 6: The helicopter is initially in $(0, 0)$ flying at 12 m/s and needs to reverse direction towards $(-100, 0)$ as quickly as possible. The only feasible option is to slow down and make a turn on the spot.

VII. Conclusion

This paper presented a hybrid architecture for autonomous trajectory planning of agile vehicles by combining multiple velocity control modes with a maneuver scheduler. The former provide the flexibility to precisely navigate between waypoints in a cluttered environment, while the latter enables execution of pre-programmed maneuvers at the limit of the vehicle capabilities. The closed-loop dynamics under this control architecture were described by a simple hybrid model consisting of a set of LTI modes and discrete, fixed-duration state transitions. Using this description of the dynamics, we formulated optimal trajectory planning through a cluttered environment as a mixed integer linear program. The framework was worked out in detail

for the case of a small-scale helicopter model based on MIT's aerobatic X-Cell. Several receding horizon scenarios were presented illustrating the real-time applicability of the framework. In addition, some practical considerations regarding an efficient implementation were discussed.

Acknowledgments

This work was funded by NASA Grant NAG 2-1552, ONR Grant N00014-03-1-0171, and AFOSR Grant F33615-01-C-1850. The authors would like to thank Dr. Vladislav Gavrilets for his input in characterizing the closed-loop dynamics and maneuver parameters of the X-Cell helicopter.

References

[1]Reif, J., "Complexity of the Mover's Problem and Generalizations," *Proc. 20th IEEE Symposium on Foundations of Computer Science*, 1979, pp. 224–241.

[2]Hopcroft, J., Schwartz, J., and Sharir, M., "On the Complexity of Motion Planning for Multiple Independent Objects: PSPACE-Hardness of the Warehouseman's Problem," *International Journal of Robotics Research*, Vol. 4, No. 3, 1984, pp. 76–88.

[3]Gavrilets, V., Mettler, B., and Feron, E., "Nonlinear Model for a Small-Size Acrobatic Helicopter," *Proc. AIAA Guidance, Navigation, and Control Conference*, No. AIAA 2001-4333, Montreal, Canada, Aug. 2001.

[4]Frazzoli, E., Dahleh, M. A., and Feron, E., "Real-Time Motion Planning for Agile Autonomous Vehicles," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 25, No. 1, 2002, pp. 116–129.

[5]Mettler, B., Valenti, M., Schouwenaars, T., Frazzoli, E., and Feron, E., "Rotorcraft Motion Planning for Agile Maneuvering," *Proc. 58th Annual Forum of the American Helicopter Society*, Montreal, Canada, June 2002.

[6]Schouwenaars, T., Mettler, B., Feron, E., and How, J., "Robust Motion Planning Using a Maneuver Automaton with Built-in Uncertainties," *Proc. 2003 American Control Conference*, Denver, CO, June 2003.

[7]Bertsekas, D., *Dynamic Programming and Optimal Control*, Vol. 1, Athena Scientific, Belmont, MA, 2nd ed., 2000.

[8]Schouwenaars, T., DeMoor, B., Feron, E., and How, J., "Mixed Integer Programming for Multi-Vehicle Path Planning," *Proc. European Control Conference (ECC'01)*, Porto, Portugal, Sept. 2001.

[9]Gavrilets, V., Frazzoli, E., Mettler, B., Piedmonte, M., and Feron, E., "Aggressive Maneuvering of Small Autonomous Helicopters: A Human-Centered Approach," *International Journal of Robotics Research*, Oct. 2001, pp. 795–807.

[10]Slotine, J.-J. and Li, W., *Applied Nonlinear Control*, Prentice Hall, Upper Saddle River, NJ, 1991.

[11]Gavrilets, V., Mettler, B., and Feron, E., "Human-Inspired Control Logic for Automated Maneuvering of Miniature Helicopter," *AIAA Journal of Guidance, Control and Dynamics*, Vol. 27, No. 5, 2004.

[12]Gavrilets, V., Martinos, I., Mettler, B., and Feron, E., "Aggressive Maneuvering Flight Tests and Simulation Results of a Miniature Robotic Helicopter," *Proc. 8th International Symposium on Experimental Robotics*, Sant'Angelo d'Ischia, Italy, July 2002.

[13]Lynch, N., Segala, R., and Vaandrager, F., "Hybrid I/O Automata Revisited," *Hybrid Systems: Computation and Control*, edited by M. D. Benedetto and A. Sangiovanni-Vincentelli, Vol. 2034 of *Lecture Notes in Computer Science (LNCS)*, Springer Verlag, 2001, pp. 403–417.

[14]Martinos, I., Schouwenaars, T., DeMot, J., and Feron, E., "Hierarchical Cooperative Multi-Agent Navigation using Mathematical Programming," *Proc. 41th Annual Allerton Conference on Communication, Control, and Computing*, Monticello, IL, Oct. 2003.

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

[16]Taha, H. A., *Operations Research, An Introduction*, Macmillan Publishing Company, New York, 4th ed., 1987.

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

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

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

[20]Bellman, R., *Dynamic Programming*, Princeton University Press, Princeton, NJ, 1957.

[21]Mayne, D., Rawlings, J., Rao, C., and Scokaert, P., "Constrained Model Predictive Control: Stability and Optimality," *Automatica*, Vol. 36, 2000, pp. 789–814.

[22]Sprague, K., Gavrilets, V., Martinos, I., Dugail, D., Mettler, B., and Feron, E., "Design and Applications of an Avionics System for a Miniature Acrobatic Helicopter," *Proc. 20th Digital Avionics Systems (DASC)*, Daytona Beach, FL, Oct. 2001.

[23]Schouwenaars, T., Feron, E., and How, J., "Hybrid Model for Receding Horizon Guidance of Agile Autonomous Rotorcraft," *Proc. 16th IFAC Symposium on Automatic Control in Aerospace*, St. Petersburg, Russia, June 2004.

[24]Gavrilets, V., *Autonomous Aerobatic Maneuvering of Miniature Helicopters*, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA, 2003.

[25]Gavrilets, V., Martinos, I., Mettler, B., and Feron, E., "Flight Test and Simulation Results for an Autonomous Acrobatic Helicopter," *Proc. 21st Digital Avionics Conference (DASC)*, Irvine, CA, Oct. 2002.

[26]Latombe, J.-C., *Robot Motion Planning*, Kluwer Academic Publishers, 1991.