Abstract: This paper presents a hybrid architecture for full-envelope autonomous rotorcraft guidance by combining a velocity control mode with a maneuver scheduler. The former provides the flexibility to accurately track trajectories, while the latter enables the execution of pre-programmed maneuvers at the limit of the vehicle capabilities. 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 finite-time transitions in the state space. Mixed integer linear programming (MILP) is then used to compute time optimal trajectories between waypoints. It is used to model the constraints governing the dynamics, to include binary logic encoding the execution of maneuvers, and to account for obstacle avoidance. Several scenarios are presented and an overview of ongoing research is given.

1 Introduction

Taking advantage of the full range of vehicle maneuverability in an autonomous fashion is key to a number of potential unmanned rotorcraft tasks. Because of the complexity of the vehicle dynamics, however, it is impractical to consider the full equations of motion in the development of a guidance system. The idea of first organizing the vehicle dynamics through some form of control augmentation, such as velocity controllers, and then designing the guidance system on the simpler closed-loop dynamics, is well established. However, for extremely agile vehicles such an approach may restrict the performance.

An alternative method is to use a “maneuver automaton” as was introduced by Frazzoli et al. in [6]. The framework was later applied to MIT’s X-Cell miniature helicopter as described in [11, 14]. With this approach, the vehicle is modeled as a hybrid automaton, consisting of a set

---

1Research Assistant, Laboratory for Information and Decision Systems, toms@mit.edu
2Post-doctoral Associate, Laboratory for Information and Decision Systems, bmettler@mit.edu
3Associate Professor, Laboratory for Information and Decision Systems, feron@mit.edu
4Associate Professor, Space Systems Laboratory, jhow@mit.edu
of discrete equilibrium trim conditions and transitions between these trims, called maneuvers. The latter can be both smooth or highly agile transitions. Optimal trajectories are obtained in real-time by evaluating the possible discrete actions at each time step, i.e. to either stay in the current trim 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 [3].

The maneuver automaton as described above, however, has several drawbacks. These are primarily related to the fact that the vehicle dynamics are constrained to a finite set of motion primitives. Namely, a continuous velocity mode is discretized into several trims with constant velocities, 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 an issue when precise navigation is required [11, 14]. 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 [15].

The paper is organized as follows. Section 2 presents the hybrid control architecture and corresponding dynamic model. Section 3 then motivates the use of MILP for optimal guidance, which is detailed for the given architecture in Section 4. Section 5 outlines the application of MILP to obstacle avoidance, and Section 6 presents some scenarios using a model inspired by MIT’s autonomous X-Cell helicopter. Next, in Section 7, the feasibility of a real-time implementation using receding horizon planning is discussed. We conclude with a survey of some shortcomings of the presented approach and motivate further research in Section 8.

2 Hybrid Control Architecture of Guidance

2.1 Hybrid architecture

The guidance framework presented in this paper uses a hybrid control architecture that combines a velocity control augmentation system and a maneuver scheduler. The former allows the vehicle to accurately track trajectories throughout most of the flight envelope, while the latter allows the execution of fast, pre-programmed maneuvers that take the vehicle through its extreme range of performance. A block-diagram of the architecture is shown in Figure 1. Under this control framework, the closed-loop dynamics of the vehicle can be accurately described by a combination of linear time invariant (LTI) low-order equations of motion and discrete state transitions.

The benefits of this approach are several. First, the model allows for precise waypoint navigation in the velocity control mode, without compromising on agility when extreme transitions are required. Second, the architecture significantly simplifies the development of the motion
2.2 Velocity control system

The architecture outlined above is motivated by the control logic of MIT's acrobatic X-Cell miniature helicopter [16]. Its velocity control augmentation system is described in [9]. Since the dynamics of the augmented helicopter change with the body axis forward speed $v$, the turn rate response is quicker in hover than in cruise flight. We consider several distinct operating regions: hover mode, where a small amount of side slip is tolerated, and the cruise mode, where turns are fully coordinated.

Since the kinematics of most vehicles are nonlinear, previous work using MILP for trajectory generation was based on approximate linear dynamic models in an inertial coordinate frame [15, 12]. In our first attempt to tackle full-envelope motion planning, we approximate the helicopter dynamics by inertial LTI state space models. The state variables then consist of the inertial position $(x, y, z)$ and inertial velocity $(v_x, v_y, v_z)$. Moreover, for optimization purposes, we consider discrete time models. Accordingly, a trajectory resulting from the optimization algorithm will consist of a sequence of inertial state vectors, which can then serve as a reference input to a tracking control system.

In the velocity control mode, the altitude rate dynamics are largely decoupled from the longitudinal and lateral ones. For simplicity of exposition, we therefore choose to focus on the planar dynamics. Since the kinematics of most vehicles are nonlinear, previous work using MILP for trajectory generation was based on approximate linear dynamic models in an inertial coordinate frame [15, 12]. In our first attempt to tackle full-envelope motion planning, we approximate the helicopter dynamics by inertial LTI state space models. The state variables then consist of the inertial position $(x, y, z)$ and inertial velocity $(v_x, v_y, v_z)$. Moreover, for optimization purposes, we consider discrete time models. Accordingly, a trajectory resulting from the optimization algorithm will consist of a sequence of inertial state vectors, which can then serve as a reference input to a tracking control system.

In the velocity control mode, the altitude rate dynamics are largely decoupled from the longitudinal and lateral ones. For simplicity of exposition, we therefore choose to focus on the planar dynamics. The framework, however, can be easily extended to account for altitude and climb rate. More specifically, we model the helicopter as a 2D double integrator over the whole velocity range. Using forward Euler discretization with a time step $\Delta t$, the corresponding
Figure 2: Operating regions of the velocity control mode shown in function of the body axis forward \((u)\) and side velocity \((v)\).

discrete state space equations are given by:

\[
\begin{bmatrix}
  x_{i+1} \\
  y_{i+1} \\
  v_{x,i+1} \\
  v_{y,i+1}
\end{bmatrix} =
\begin{bmatrix}
  1 & 0 & \Delta t & 0 \\
  0 & 1 & 0 & \Delta t \\
  0 & 0 & 1 & 0 \\
  0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x_i \\
  y_i \\
  v_{x,i} \\
  v_{y,i}
\end{bmatrix} +
\begin{bmatrix}
  0 & 0 \\
  0 & 0 \\
  \Delta t & 0 \\
  0 & \Delta t
\end{bmatrix}
\begin{bmatrix}
  a_{x,i} \\
  a_{y,i}
\end{bmatrix}
\]

(1)

in which the input variables \(a_{x,i}\) and \(a_{y,i}\) are the inertial accelerations at the \(i\)th time step.

To distinguish between the different operating regions, we need to introduce additional constraints that account for limits on side slip and turn rate. In hover mode, we consider circular bounds on the magnitudes of the inertial velocity vector \(v = (v_x, v_y)\) and acceleration \(a = (a_x, a_y)\). They are bounded by the maximum speed \(v_{h,max}\) allowed in hover and the total maximum acceleration \(a_{max}\) respectively:

\[
||v|| \leq v_{h,max}
\]

(2)

\[
||a|| \leq a_{max}
\]

(3)

As will become clear in Section 4, these inequalities need to be linearized. We achieve this by approximating the circular constraints (2) and (3) by the edges of a \(K\)-sided polygon [12]:

\[
\forall k \in [1...K] : \quad v_x \sin \left( \frac{2\pi k}{K} \right) + v_y \cos \left( \frac{2\pi k}{K} \right) \leq v_{h,max}
\]

(4)

\[
a_x \sin \left( \frac{2\pi k}{K} \right) + a_y \cos \left( \frac{2\pi k}{K} \right) \leq a_{max}
\]

(5)
This is illustrated in Figure 3 for the velocity constraint. Also, note that we left out the time index \(i\) to simplify the notation.

In cruise flight, however, the helicopter flies in an airplane-like fashion, i.e. no side slip is permitted. Moreover, the maximum turn rate \(r_{\text{max}}\) is inversely proportional to the body axis forward velocity \(u\). Namely, \(r_{\text{max}} = \frac{g}{2u}\), where \(g = 9.81\text{m/s}^2\). Both requirements, however, can be approximately accounted for by constraining the inertial acceleration vector to lie in a narrow elliptic region that is parallel to the inertial speed vector. This limits the lateral acceleration, thereby enforcing the turn rate constraint without compromising longitudinal forward acceleration and deceleration. Since the maximum turn rate scales inversely with the forward velocity, the shape of the ellipse needs to be specified for each operating region in the partition. Within one operation region, however, the limit on turn rate will be assumed constant.

For the purpose of later optimization, the elliptic constraint is approximated by the intersection of two circles, whose centers lie along the line going through the origin and parallel to the orthogonal complement \(\mathbf{v}_\perp = (-v_y, v_x)\) of the velocity vector \(\mathbf{v}\). This is illustrated in Figure 4. Using appropriate scaling factors \(\alpha_l\) and \(\beta_l\) to produce the circles for each cruise mode \(l\), we obtain:

\[
\| \mathbf{a} - \alpha_l \mathbf{v}_\perp \| \leq \beta_l a_{\text{max}} \quad (6)
\]
\[
\| \mathbf{a} + \alpha_l \mathbf{v}_\perp \| \leq \beta_l a_{\text{max}} \quad (7)
\]
Again, a linearized version of these quadratic constraints is given by:

\[ \forall k \in [1...K] : (a_x + \alpha_k v_y) \sin \left( \frac{2\pi k}{K} \right) + (a_y - \alpha_k v_x) \cos \left( \frac{2\pi k}{K} \right) \leq \beta a_{\max} \quad (8) \]

\[ (a_x - \alpha_k v_y) \sin \left( \frac{2\pi k}{K} \right) + (a_y + \alpha_k v_x) \cos \left( \frac{2\pi k}{K} \right) \leq \beta a_{\max} \quad (9) \]

2.3 Maneuver scheduler

The maneuver scheduler allows the execution of pre-programmed maneuvers that result in a rapid and extreme change of the helicopter’s state, flight path, or position. Typically, these are designed to exploit the full performance and agility of the vehicle, i.e., they take advantage of the full control input range and result in large coordinated state excursions. The availability of such maneuvers will play an essential role in the reactive threat and obstacle avoidance capabilities of the vehicle. Table 1 gives an overview of maneuvers that could be designed, and Figure 5 shows the corresponding vehicle trajectories.

Both split-S and hammerhead have already been implemented on MIT’s helicopter [7, 8]. Other maneuvers that are being 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 U-turn, they are faster and ideally require no lateral displacements. However, both require a minimum entrance 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 altitude or higher.

Each maneuver is designed individually and programmed in the control system. For path planning purposes, a maneuver is characterized by its duration \( \Delta T \), initial and exit speed \( v_{\text{init}} \)
Table 1: Description of sample maneuvers that could be implemented on a rotorcraft-type vehicle.

<table>
<thead>
<tr>
<th>Maneuver</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dash to cruise</td>
<td>rapid acceleration from hover to one of the cruise conditions</td>
</tr>
<tr>
<td>Quick stop</td>
<td>rapid transition from cruise to a full stop (hover)</td>
</tr>
<tr>
<td>Quick Turn</td>
<td>rapid turn resulting in a pre-determined heading change</td>
</tr>
<tr>
<td>Split-S</td>
<td>reversal of the flight direction with negative altitude loss</td>
</tr>
<tr>
<td>Hammerhead</td>
<td>reversal of the flight direction with positive or no altitude loss</td>
</tr>
</tbody>
</table>

and $v_{\text{exit}}$, resulting spatial displacement $[\Delta x, \Delta y, \Delta z]$ and change in heading $\Delta \psi$. Both the displacement and heading change are defined with respect to the body-fixed frame at the start of the maneuver, as shown in Figure 6.

From these body frame parameters, a fixed linear transformation of the inertial state vector can be extracted for each maneuver $m$. We assume here that a maneuver can only be initialized
Figure 6: Change in helicopter position and orientation resulting from a maneuver (shown in dashed line), as observed from the body-fixed frame in level flight with zero side slip. We then define the following derived parameters:

\[
\begin{align*}
\gamma_m &= -\arctan\left(\frac{\Delta x_m}{\Delta y_m}\right) \\
\delta_m &= -\Delta \psi_m \\
c_m &= \sqrt{(\Delta x_m)^2 + (\Delta y_m)^2} \\
d_m &= \frac{v_{\text{exit},m}}{v_{\text{init},m}}
\end{align*}
\]  

(10) (11) (12) (13)

where \(\gamma_m\) is defined between \(-180^\circ\) and \(180^\circ\). The state transition resulting from maneuver \(m\) is then given by the following linear transformation:

\[
\begin{bmatrix}
x_{\text{exit}} \\ y_{\text{exit}} \\ v_{x,\text{exit}} \\ v_{y,\text{exit}}
\end{bmatrix} = \begin{bmatrix}
1 & 0 & c_m \cos \gamma_m & -c_m \sin \gamma_m \\
0 & 1 & c_m \sin \gamma_m & c_m \cos \gamma_m \\
0 & 0 & d_m \cos \delta_m & -d_m \sin \delta_m \\
0 & 0 & d_m \sin \delta_m & d_m \cos \delta_m
\end{bmatrix} \begin{bmatrix}
x_{\text{init}} \\ y_{\text{init}} \\ v_{x,\text{init}} \\ v_{y,\text{init}}
\end{bmatrix}
\]

(14)

which combines rotation, scaling and translation to express the exit state as a function of the initial state. For the remainder of the paper, we will use the following shorthand notation:

\[
x_{\text{exit}} = C_m x_{\text{init}}
\]

(15)
3 Trajectory Optimization using MILP

3.1 Sequential decision process

Given the framework described above, our goal is to compute shortest time trajectories between two waypoints, corresponding to an initial and a final state. Hence, an optimal trajectory will consist of a sequence of states that minimizes the total transition time between a start and end position and velocity vector. The waypoints are typically provided by a higher level planning algorithm that optimizes a certain task or mission criterion. In this paper, we will assume that such a higher planning level is in place.

Since — for planning purposes—, maneuvers are considered to be discrete transition entities between two LTI-modes, 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-regimes. Hence, the guidance problem comes down to deciding at each decision step whether to stay in the current LTI-mode, to transition to a neighboring LTI-regime, or to execute a certain maneuver. However, the last option is only available if the initial conditions for that maneuver are satisfied. When the helicopter is flying in an LTI-mode, the decision steps correspond to normal discrete time steps. In the maneuver execution mode, however, the single decision step in which a maneuver is executed, corresponds to a number of time steps equivalent to the actual duration of the maneuver. This difference should be accounted for in the formulation of the trajectory optimization problem.

3.2 Mixed Integer Linear Programming

The guidance decision logic outlined above, lends itself well to be formulated as a mixed integer linear program (MILP). MILP is a powerful mathematical framework that allows the inclusion of integer variables and discrete logic in a continuous linear optimization problem [5]. It is commonly used in Operations Research [17], and has more recently been introduced to the field of hybrid systems [2]. In our case, the continuous optimization is done over the state and inputs in the LTI velocity control mode; the discrete logic is introduced by the partitioning of the state space and the option of executing a maneuver when its initial conditions are satisfied.

As an illustration on how logical decisions can be incorporated in an optimization problem, consider the following simple example. A cost function $J(x)$ needs to be minimized subject to either one of two constraints $\ell_1(x), \ell_2(x)$ on the continuous decision variable $x$:

$$
\min_x J(x) \\
\text{subject to:} \\
\ell_1(x) \leq 0 \\
\text{OR } \ell_2(x) \leq 0
$$

(16)

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

\[
\begin{align*}
\min_{x} & \quad J(x) \\
\text{subject to:} & \quad \ell_1(x) \leq Mb \\
& \quad \text{AND} \quad \ell_2(x) \leq M(1-b) \\
& \quad b \in \{0,1\}
\end{align*}
\]  

(17)

When \( b = 0 \), constraint \( \ell_1(x) \) must be satisfied, whereas \( \ell_2(x) \) is relaxed. Namely, if \( M \) is chosen sufficiently large, \( \ell_2(x) \leq M(1-b) \) is always satisfied independent of the value of \( x \). The situation is reversed when \( b = 1 \). Since \( b \) can only take the binary values 0 or 1, at least one of the constraints \( \ell_1(x) \) and \( \ell_2(x) \) will be satisfied, which is equivalent to the original “OR”-formulation (16). In the special case where \( J(x) \), \( \ell_1(x) \) and \( \ell_2(x) \) are (affine) linear expressions, problem (17) is a MILP.

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

\[
\begin{align*}
\min_{x} & \quad J(x) \\
\text{subject to:} & \quad \ell_k(x) \leq M b_k, \; k = 1 \ldots K \\
& \quad \sum_k b_k \leq K - L \\
& \quad b_k \in \{0,1\}
\end{align*}
\]  

(18)

The additional summation constraint ensures that at least \( L \) of the binary variables \( b_k \) are 0, thus guaranteeing that at least \( L \) of the inequalities \( \ell_k(x) \leq 0 \) are satisfied simultaneously.

4 Rotorcraft MILP Formulation

We now apply the mathematical framework described in the previous section to the optimal guidance problem for the helicopter.

4.1 LTI-mode switching

Assume that the velocity control mode is partitioned into \( L \) operating regions, denoted by indices \( l \). Although we considered the same double integrator model in all regimes, in general the \( L \) LTI-modes can have different state space matrices \( A_l \) and \( B_l \) and a general input vector \( u \). In our case, \( u \) only contains the accelerations \( (a_x, a_y) \). Each LTI-mode \( l \) has a minimum velocity \( v_{\text{min},l} \) and a maximum velocity \( v_{\text{max},l} \) that define the operating region. For the hover mode \( (l = 1) \) we have \( v_{\text{min},h} = 0 \). A transition between two LTI-modes is triggered when the corresponding velocity bounds are exceeded.

As discussed before, the helicopter must be in exactly one LTI-mode at the beginning of each decision step. Hence, with every decision step \( i \), we can associate a binary variable \( b_{il} \) that
equals 1 when the helicopter is flying in mode \( l \). 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 thus need to be relaxed, which can be expressed as follows:

\[
\forall l \in [1 \ldots L]: \quad x_{i+1} - A_l x_i - B_l u_i \leq M(1 - b_{il}) \\
-x_{i+1} + A_l x_i + B_l u_i \leq M(1 - b_{il}) \\
\sum_{i=1}^{L} b_{il} = 1
\]  

(19)

The mode selection variable \( b_{il} \) is assigned through the value of the velocity magnitude, for which inequalities (4) mark the upper bound. Satisfying the minimum velocity criterion \( ||v|| \geq v_{\text{min},l} \), however, is a non-convex constraint. The velocity vector must now lie outside the \( K \)-polygon associated with \( v_{\text{min},l} \). For this purpose, it is sufficient that the vector lies in at least one of the outer halfplanes defined by the edges of the polygon. The situation is depicted in Figure 3. Using \( K \) binary variables \( c_{ilk} \) for each decision step \( i \) and mode \( l \), the minimum velocity constraint for mode \( l \) can be expressed as follows:

\[
\forall k \in [1 \ldots K]: \quad v_{xi} \sin \left( \frac{2 \pi k}{K} \right) + v_{yi} \cos \left( \frac{2 \pi k}{K} \right) \geq v_{\text{min},l} - Mc_{ilk} \\
\sum_{k=1}^{K} c_{ilk} \leq K - 1
\]  

(20)

Combining the set of inequalities (19) with the velocity constraints (4) and (20), the acceleration constraints (5) and (8), and introducing the necessary “big M” relaxations yields the complete LTI-mode switching logic for the \( i^{th} \) decision step:

\[
\forall l \in [1 \ldots L]: \quad x_{i+1} - A_l x_i - B_l u_i \leq M(1 - b_{il}) \\
-x_{i+1} + A_l x_i + B_l u_i \leq M(1 - b_{il}) \\
\sum_{i=1}^{L} b_{il} = 1
\]  

\[
l = 1: \quad \forall k \in [1 \ldots K]: \quad v_{xi} \sin \left( \frac{2 \pi k}{K} \right) + v_{yi} \cos \left( \frac{2 \pi k}{K} \right) \leq v_{\text{max},h} + M(1 - b_{1l}) \\
a_{xi} \sin \left( \frac{2 \pi k}{K} \right) + a_{yi} \cos \left( \frac{2 \pi k}{K} \right) \leq a_{\text{max}} + M(1 - b_{1l})
\]  

(21)

\[
\forall l \in [2 \ldots L]: \quad \forall k \in [1 \ldots K]: \quad v_{xi} \sin \left( \frac{2 \pi k}{K} \right) + v_{yi} \cos \left( \frac{2 \pi k}{K} \right) \leq v_{\text{max},l} + M(1 - b_{il}) \\
-v_{xi} \sin \left( \frac{2 \pi k}{K} \right) - v_{yi} \cos \left( \frac{2 \pi k}{K} \right) \leq -v_{\text{min},l} + M(1 - b_{il}) + Mc_{ilk} \\
\sum_{k=1}^{K} c_{ilk} \leq K - 1
\]

\[
(a_x + \alpha_l v_y) \sin \left( \frac{2 \pi k}{K} \right) + (a_y - \alpha_l v_x) \cos \left( \frac{2 \pi k}{K} \right) \leq \beta_l a_{\text{max}} + M(1 - b_{il}) \\
(a_x - \alpha_l v_y) \sin \left( \frac{2 \pi k}{K} \right) + (a_y + \alpha_l v_x) \cos \left( \frac{2 \pi k}{K} \right) \leq \beta_l a_{\text{max}} + M(1 - b_{il})
\]

4.2 Maneuver execution

Consider a set of \( P \) maneuvers denoted by indices \( m \) and characterized by transition matrices \( C_m \) as in Eq. (15). Maneuver \( m \) can only be executed when its initial conditions are satisfied,
namely, in our case, when the required initial speed \( v_{\text{init},m} \) is attained. This check can be performed by the same principle that was used to derive the inequalities (4) and (20) for the LTI-mode bounds. In this case, however, the inner and outer polygons coincide:

\[
\forall m \in [1 \ldots P] : \quad x_{i+1} - C_m x_i \leq M(1 - d_{im}) \\
-x_{i+1} + C_m x_i \leq M(1 - d_{im}) \\
\sum_{m=1}^{P} d_{im} \leq 1
\]

\[
\forall k \in [1 \ldots K] : \quad v_{xi} \sin \left( \frac{2\pi k}{K} \right) + v_{yi} \cos \left( \frac{2\pi k}{K} \right) \leq v_{\text{init},m} + M(1 - d_{im}) \\
-v_{xi} \sin \left( \frac{2\pi k}{K} \right) - v_{yi} \cos \left( \frac{2\pi k}{K} \right) \leq -v_{\text{init},m} + M(1 - d_{im}) + M \epsilon_{imk} \\
\sum_{k=1}^{K} \epsilon_{imk} \leq K - 1
\]

(22)

If the binary selection variable \( d_{im} = 1 \), maneuver \( m \) is executed at the \( i \)-th decision step. The inequality \( \sum_{m=1}^{P} d_{im} \leq 1 \) expresses the fact that at most one maneuver can be executed at a time. Note that if the initial conditions for a certain maneuver are satisfied, it does not necessarily have to be executed. However, if the guidance algorithm decides to perform a maneuver, the state space inequalities of the LTI-modes in (19) need to be relaxed, since the state transition is now given by (15). We therefore extend the constraints (19) as follows:

\[
\forall l \in [1 \ldots L] : \quad x_{i+1} - A_l x_i - B_l u_i \leq M(1 - b_{il}) + M \sum_{m=1}^{P} d_{im} \\
-x_{i+1} + A_l x_i + B_l u_i \leq M(1 - b_{il}) + M \sum_{m=1}^{P} d_{im} \\
\sum_{i=1}^{L} b_{il} = 1
\]

(23)

If no maneuver is performed, the extra relaxation term \( M \sum_{m=1}^{P} d_{im} \) will equal 0.

4.3 Time optimal trajectory

Given an initial state \( x_{\text{init}} \), we now want to compute the time optimal trajectory to a desired final state \( x_{\text{final}} \). 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 introduce binary variables \( t_i \) that select the step at which the final state is reached. An additional equality constraint is needed that enforces the helicopter to actually reach the desired state at one of the decision steps in the planning horizon [12]. More specifically, consider an horizon of \( T \) decision steps, where \( T \) is a heuristic upper bound on the number of steps needed to reach the desired state. The arrival requirement can then be formulated as follows:

\[
\forall i \in [0 \ldots T] : \quad x_i - x_{\text{final}} \leq M(1 - t_i) \\
x_{\text{final}} - x_i \leq M(1 - t_i) \\
\sum_{i=0}^{N} t_i = 1 \\
x_0 = x_{\text{init}}
\]

(24)
The shortest trajectory time can now be expressed as the minimum of the following cost function:

\[
J = \sum_{i=0}^{T} t_i \Delta t + \sum_{i=0}^{T-1} \sum_{m=1}^{P} c_{im}(\Delta T_m - \Delta t)
\]  

(25)

Since only one of the \(t_i\) 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 \(\Delta t\). If the optimal action at the \(i\)th decision step is to stay in LTI-regime, the weight of the step corresponds to the discretization step \(\Delta t\). However, if the optimal action is to perform a maneuver \((c_{im} = 1)\), the weight of the decision step is the maneuver duration \(\Delta T_m\). Since the first term in \(J\) already accounts for a weight \(\Delta t\), the latter is subtracted from the actual maneuver time \(\Delta T_m\).

Finally, to generate smooth and natural trajectories, we need to add an input regularization term to the cost function \(J\) in (25). Namely, since in our case the \(x-, y-\) (and \(z-\)) dynamics are fully decoupled, a straight trajectory could have the same duration as one that wiggles around it. To avoid this effect, we modify the cost function as follows:

\[
\tilde{J} = \sum_{i=0}^{T} t_i \Delta t + \sum_{i=0}^{T-1} \sum_{m=1}^{P} c_{im}(\Delta T_m - \Delta t) + \epsilon \sum_{i=0}^{T-1} \sum_{n=1}^{n_u} |u_{in}|
\]

(26)

in which \(\epsilon\) is a very small number and \(n_u\) denotes the dimension of the input vector. This will ensure that among the set of time optimal paths, the one with the least number of lateral (and altitude) excursions is chosen. This path will typically correspond to the shortest distance trajectory in that set.

Note that the regularization term contains the absolute value of the input variables rather than the variables themselves. As such, \(\tilde{J}\) is a piece-wise linear cost function that at first sight does not fit the MILP framework. However, by introducing auxiliary variables and extra constraints, an equivalent linear formulation can be generated. Details can be found in [15, 4].

The cost function (26) subject to the constraint sets (24), (22) and (21) modified with the relaxed state space inequalities (23), then form the full MILP for time-optimal trajectory generation.

### 5 Obstacle avoidance using MILP

If obstacles are present in the environment, the optimization algorithm must account for these in computing the optimal trajectory. As was shown in [15], obstacle avoidance can be handled by mixed integer linear constraints as well. As is common practice in the field of robot motion planning, the obstacles are enlarged with the dimensions of the vehicle, such that the vehicle itself can be treated as a point [10]. Consider now for simplicity of exposition a rectangular obstacle in a 2-dimensional space, with lower left corner \((x_{\min}, y_{\min})\) and upper right corner \((x_{\max}, y_{\max})\). To avoid the obstacle, each trajectory point \((x_i, y_i)\) of the point mass helicopter must satisfy at least one of the following inequalities:

\[
\begin{align*}
    x_i &\leq x_{\min} \\
    \text{OR} & -x_i \leq -x_{\max} \\
    y_i &\leq y_{\min} \\
    \text{OR} & -y_i \leq -y_{\max}
\end{align*}
\]

(27)
These inequalities represent the four halfplanes defined by the edges of the rectangular obstacle. As discussed before, by introducing binary variables \( f_{ik} \) and a sufficiently large positive number \( M \), the logical constraints (27) are equivalent to:

\[
\begin{align*}
    x_i & \leq x_{\min} + M f_{i1} \\
    -x_i & \leq -x_{\max} + M f_{i2} \\
    y_i & \leq y_{\min} + M f_{i3} \\
    -y_i & \leq -y_{\max} + M f_{i4} \\
\sum_{k=1}^{4} f_{ik} & \leq 3
\end{align*}
\]

(28)

The last inequality ensures that at least one of the original “OR”-constraints is active, thereby guaranteeing that the trajectory point \((x_i, y_i)\) lies outside the rectangle. This set of inequalities must be formulated for each time step \( i = 1 \ldots T \), and for each obstacle in the operating region of the vehicle. Note that this method can be extended to a 3D environment and to arbitrarily shaped obstacles which are then approximated by a polygon or polyhedron.

However, because the trajectory consists of a discrete sequence of positions, there is no guarantee that the corresponding continuous trajectory does not cut through 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. When maneuvers are considered, however, the required safety boundary can be relatively large compared to the size of the obstacle. Alternative methods are therefore to carry out the avoidance check for sampled points along the maneuver trajectory, or to contain the maneuver in a box that should not overlap with any of the obstacles when the maneuver is being executed. These approaches are currently still being investigated.

6 Examples

For the examples presented in this section, we use a model inspired by MIT’s autonomous X-Cell miniature helicopter. We consider a hover mode up to 4 m/s, a maximum forward velocity of 20 m/s, and a maximum acceleration and deceleration of 3 m/s\(^2\). For simplicity, we consider only one forward flight mode with constant double circle constraints, enforcing a maximum turn rate of 25 deg/s. We include two maneuvers: a hammerhead and a split-S, for which the parameters are given in Table 2. These values are rough averages of the actual values resulting from autonomous execution of these maneuvers on the X-Cell. Note that an ideal execution would not result in a lateral displacement. Also, as discussed before, we do not account for the altitude change in the examples.

In a first scenario, the helicopter starts in hover at the origin and is pointing east. It subsequently has to fly through a waypoint \((40, 0)\) at a speed \(v_x = 12\) m/s, through \((10, 15)\) at \(v_x = -15\) m/s, and must end in hover at position \((-40, 15)\). The time optimal trajectory is depicted in Figure 7. After flying through the first waypoint, the helicopter slows down, executes a sharp U-turn, and then speeds up to reach 15 m/s at the second waypoint. With a discretization step of 0.5s, the total flight duration is 21s.
Table 2: Parameters for pre-programmed maneuvers

<table>
<thead>
<tr>
<th>Maneuver</th>
<th>$\Delta x (m)$</th>
<th>$\Delta y (m)$</th>
<th>$\Delta z (m)$</th>
<th>$\Delta \psi (deg)$</th>
<th>$\Delta T (s)$</th>
<th>$v_{init} (m/s)$</th>
<th>$v_{init} (m/s)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Split-S</td>
<td>6.5</td>
<td>10</td>
<td>0</td>
<td>180</td>
<td>5</td>
<td>15</td>
<td>18</td>
</tr>
<tr>
<td>Hammerhead</td>
<td>-20</td>
<td>-6</td>
<td>-40</td>
<td>180</td>
<td>7</td>
<td>18</td>
<td>18</td>
</tr>
</tbody>
</table>

Figure 7: Example of optimal waypoint navigation. The helicopter must fly through point (40, 0) at 12 m/s and through (10, 15) at $v = 15$ m/s.

In a second scenario, the waypoints are the same, but the required speed to fly through (40, 0) is now 14 m/s. As can be seen in Figure 8, in this case it is faster to execute the hammerhead maneuver. To enable the latter, the helicopter first needs to accelerate to 15 m/s. Since the exit speed is 18 m/s, it also needs to decelerate back to 15 m/s after execution of the hammerhead. The extra time these actions require, however, is compensated by the relatively short time it takes to reverse heading. The total trajectory now takes 22s. Note that the complete hammerhead trajectory itself is abstracted and shown by a dashed line representing the discrete state transition of the maneuver. To allow for a comparison, we ran the same scenario but with no maneuvers available to the helicopter. The naturally optimal action is then to perform a U-turn, resulting in a trajectory that takes 22.5s.

A third scenario is to quickly reverse direction of flight; this could be required if there is a threat ahead. In this case, we do not constrain the position of the final state, but only require the velocity to reach a particular value. More specifically, assume that the helicopter is
initially flying east through the origin at \( v_x = 10 \text{ m/s} \). Given the task to change velocity to \( v_x = -20 \text{ m/s} \) as quickly as possible, the optimal action sequence consists of accelerating to 15 m/s, executing the split-S, and further accelerating to 20 m/s from the 18 m/s exit velocity. The entire sequence takes 8s and is shown in Figure 9.

Figure 10 presents the same direction reversal scenario, but with a constraint on lateral displacement. The helicopter now has to stay within a space of width 4m, which accounts for any safety boundaries resulting from the actual displacement when executing a maneuver. This geometry could for example represent a city street aligned with buildings. Since the split-S cannot be executed in this confined area, the helicopter opts for the hammerhead, resulting in a total time of 10.5s. Note that it first has to make a lateral displacement and has to reorient itself slightly to fit the hammerhead between the boundaries. Further reduction of the available width to 3m will eventually result in a full stop, followed by a turn on the spot and a subsequent acceleration as the only feasible option to reverse direction of flight. This is shown in Figure 11. The total transition now takes 11s.

Figure 8: Second example of optimal waypoint navigation. The helicopter must now fly through point \((40,0)\) at 14 m/s, and executes the hammerhead maneuver to quickly change direction afterwards.
Figure 9: Time optimal trajectory to reverse flight direction using the split-S.

Figure 10: Time optimal trajectory to reverse flight direction using the hammerhead when bounds on lateral displacement are given.
7 Receding Horizon Implementation

A significant drawback of MILP is that the computation time increases at least polynomially with the number of variables and constraints. Therefore, the minimum time approach presented in this paper is mainly only 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 limited number of decision steps and a reduced set of LTI-modes and maneuvers.

Offline trajectory planning, however, has several disadvantages. First, it does not allow for changes in the vehicle dynamics or for modifications in the obstacle field during flight. As such, an offline MILP planning strategy is not robust to uncertainties. Moreover, all necessary mission information has to be available beforehand, i.e., before the vehicle starts its execution. During the execution phase, the navigation control system of the vehicle is restricted to tracking the precomputed path. As such, offline computation defeats the purpose of the hybrid control architecture aimed at enabling reactive threat avoidance.

These limitations can be effectively addressed using a receding horizon (RH) planning strategy [1, 15, 13]. In this case, the path of the vehicle is computed iteratively and is 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, 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. However, only a subset of these $T$ actions is actually implemented. Instead, the process is repeated periodically and a new set of commands is computed for each time window. Note
that this way, new information about the environment can be incorporated at each iteration, which is crucial when the environment changes or is explored in real-time.

Since the planning horizon is limited by the available computational power, the initial and final state of the local MILP segment must be sufficiently close to each other for the problem to be feasible. However, when the waypoints generated by the higher level mission planner lie relatively far apart, the receding horizon algorithm itself must generate intermediate waypoints for the local segments. Based on the principle of optimality [3], this is typically done by using an estimate of the time-to-go from the intermediate waypoint to the desired final waypoint. The construction of such a time-to-go estimate can be done offline and be stored in a lookup table. Such an approach using MILP was presented by Bellingham et al. in [1]. However, at this point, it is still unclear how to incorporate a time-to-go estimate when agile maneuvering capabilities are included in the dynamics. It is a topic of ongoing research, as are heuristics that exploit the use of as much a priori information as possible to speed up the online computation at each iteration.

8 Issues, Current and Future Work

The work presented in this paper is part of an ongoing evaluation of MILP as a solution technique for real-time motion planning of autonomous vehicles. As such, the inertial LTI model combined with fixed maneuvers as used in this paper is a first approach to tackle the complex problem of full flight envelope motion planning using MILP. However, the model has several shortcomings.

A first drawback is the approximate formulation of limited turn rate and zero side slip constraints in cruise flight. Second, the current model does not permit to keep track of inertial heading while planning a path, because this involves nonlinear expressions. A third shortcoming is the need for an accurate reference state tracking system, which should track both inertial position and velocity. For these reasons, a linear model that is closer to the actual body-fixed frame dynamics and control inputs is desired.

Current efforts are therefore focused towards using a low-order LTI-model in the body-fixed frame. MIT’s X-Cell helicopter lends itself well to such an approach. Its velocity control augmentation system features the following command variables: body axis forward velocity $u_{cmd}$ and side velocity $v_{cmd}$, altitude rate $h_{cmd}$ 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. As such, the dynamics of the X-Cell can be accurately modeled by the following decoupled, first-order LTI equations:

$$\begin{align*}
\dot{u} &= -\frac{1}{\tau_{u4}} u + \frac{1}{\tau_{u4}} u_{cmd} \\
\dot{v} &= -\frac{1}{\tau_{v4}} v + \frac{1}{\tau_{v4}} v_{cmd} \\
\dot{h} &= -\frac{1}{\tau_{h4}} h + \frac{1}{\tau_{h4}} h_{cmd} \\
\dot{r} &= -\frac{1}{\tau_{r4}} r + \frac{1}{\tau_{r4}} r_{cmd} \tag{29}
\end{align*}$$

One benefit of using this model is that the lower and upper bounds on velocity, acceleration, and control inputs can be expressed by single inequalities, instead of the multiple polygonal
constraints with binary variables in the inertial model. Moreover, the optimal inputs computed using MILP can be used as reference commands to the velocity control system. As such — provided that it is solved fast enough — MILP can directly be applied as a closed-loop guidance method, for which a receding horizon implementation is a natural approach.

However, the inertial kinematics associated with this body frame model are nonlinear and given by:

\[

t_N = u \cos \psi - v \sin \psi \\
vt_E = u \sin \psi + v \cos \psi \\
vt_D = -\dot{h} \\
\dot{\psi} = r
\]

Therefore, to use MILP for guidance and obstacle avoidance, we need to approximate the sine and cosine by piece-wise constant functions, which involves the use of binary variables. This body-fixed frame approach, with the use of receding horizon planning in closed-loop, is a topic of current and future research, and will be detailed in future publications.

9 Conclusion

In this paper, we presented a hybrid architecture for autonomous full-envelope rotorcraft guidance by combining multiple velocity control modes with a maneuver scheduler. The former provide the flexibility to accurately track trajectories, while the latter enables the 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 finite-time state transitions. Using this framework, we formulated time optimal trajectory planning including obstacle avoidance constraints as a mixed integer linear program. Several example scenarios were given, and a receding horizon approach was discussed. We concluded with an overview of the shortcomings of the current proposed framework and a motivation for ongoing and future research.

10 Acknowledgements

This work was funded by NASA Grant NAG 2-1552 and the Office of Naval Research (ONR) Grant N00014-03-1-0171. The authors would like to thank Vladislav Gavrilets for his input in characterizing the closed-loop dynamics and maneuver parameters of the X-Cell helicopter. Tom Schouwenaars is thankful to Arthur Richards for their discussion on the double circle constraints for lateral acceleration.

References


