Emergency Landing Planning for Damaged Aircraft

Nicolas Meuleau[*] and **Christian Plaunt** and **David E. Smith**
NASA Ames Research Center
M.S. 269-3
Moffet Field, California 94035-1000
{nicolas.f.meuleau,christian.j.plaunt,david.smith}@nasa.gov

Abstract

Considerable progress has been made over the last 15 years on building adaptive control systems to assist pilots in flying damaged aircraft. Once a pilot has regained control of a damaged aircraft, the next problem is to determine the best site for an emergency landing. In general, the decision depends on many factors including the actual control envelope of the aircraft, distance to the site, weather en route, characteristics of the approach path, characteristics of the runway or landing site, and emergency facilities at the site. All of these influence the *risk* to the aircraft, to the passengers and crew, and to people and property on the ground. We describe an ongoing project to build and demonstrate an emergency landing planner that takes these various factors into consideration and proposes possible routes and landing sites to the pilot, ordering them according to estimated risk. We give an overview of the system architecture and input data, describe our preliminary modeling of risk, and describe how we search the space of landing sites and routes.

Introduction

On July 19, 1989, United flight 232, a DC-10 enroute from Denver to Chicago, suffered an uncontained failure of the fan blades in the number two (rear) engine. The resulting debris severed hydraulic lines in the airplane's tail resulting in loss of all hydraulic fluid, and consequent loss of all aircraft control surfaces. Miraculously, a DC-10 flight instructor on board the aircraft was able to regain some semblance of control using differential thrust from the two remaining engines. An emergency landing was subsequently attempted at Sioux City, IA. Because of the high landing speed, high descent rate, and limited control, the aircraft broke up on impact, but 10 of 11 crew members and 175 of the 285 passengers survived the accident (NTSB 1990).

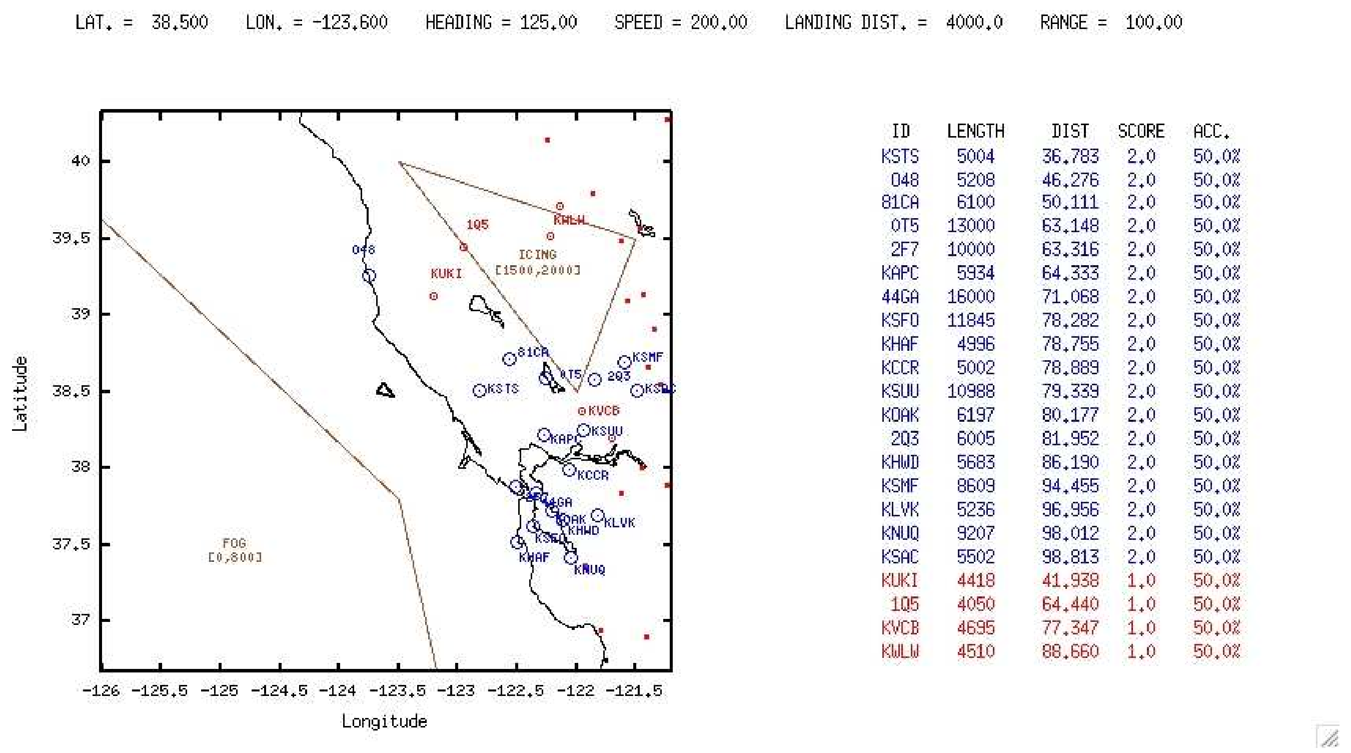
This accident, and others involving structural damage to aircraft, motivated research on adaptive control systems, aimed at allowing a pilot to continue to fly a damaged aircraft using stick inputs. The adaptive controller translates those inputs into novel combinations of thrust vectoring and control surface movements in order to achieve the pilots intent. Testing of these controllers in full motion simulation, and in test aircraft has been very successful so far (see for example (Burcham *et al.* 1996; Burken & Burcham 1997; Gundy-Burlet *et al.* 2004)). As a result, such control systems are being seriously considered for next generation military and civil transport aircraft. This capability, while quite remarkable, only addresses the first piece of the problem – regaining control of the aircraft. Once this is achieved, the next problem is to determine the best site for an emergency landing. In general, the decision depends on many factors including the actual control envelope of the aircraft, distance to the site, weather en route, characteristics of the approach path, characteristics of the runway or landing site, and emergency facilities available at the site. All of these influence the *risk* to the aircraft, to the passengers and crew, and to people and property on the ground. A purely secondary consideration is airline and passenger convenience.

Although pilots are highly trained in emergency procedures, structural damage and the consequent changes in flight characteristics strain the limits of their intuition and ability to assess different possible options. It would therefore be very useful to have an automated system that could, in seconds, generate and evaluate different possible emergency landing plans, and present the best options to the pilot. Furthermore, as the flight progresses, it is necessary to continually update and reevaluate the set of options to take into account the changing location, altitude and velocity of the aircraft, subsequent degradation or failures that change the predicted control envelope, and updated weather and airport information.

Fundamentally, this problem is a 3D path planning problem involving dynamics (aircraft speed and direction), with complex optimization criterion. It may, for example be possible for the aircraft to fly through a region of moderate turbulence, but because of the limited control authority, there is increased risk of loss of control. It might also be easier (as it was for United 232) for the aircraft to make right turns rather than left turns, or to handle a right crosswind, rather than a left crosswind.

Traditionally, difficult path planning problems have been solved using either discretization of the space, or by generating probabilistic road maps. For this problem, discretization often results in paths that have turns in them, even if the

[*]Carnegie Mellon University



```
LAT. =  38.500    LON. = -123.600    HEADING = 125.00    SPEED = 200.00    LANDING DIST. =  4000.0    RANGE =  100.00
```

ID	LENGTH	DIST	SCORE	ACC.
KSTS	5004	36.783	2.0	50.0%
048	5208	46.276	2.0	50.0%
81CA	6100	50.111	2.0	50.0%
0T5	13000	63.148	2.0	50.0%
2F7	10000	63.316	2.0	50.0%
KAPC	5934	64.333	2.0	50.0%
44GA	16000	71.068	2.0	50.0%
KSFO	11845	78.282	2.0	50.0%
KHAF	4996	78.755	2.0	50.0%
KCCR	5002	78.889	2.0	50.0%
KSUU	10988	79.339	2.0	50.0%
KOAK	6197	80.177	2.0	50.0%
2Q3	6005	81.952	2.0	50.0%
KHWD	5683	86.190	2.0	50.0%
KSMF	8609	94.455	2.0	50.0%
KLVK	5236	96.956	2.0	50.0%
KNUQ	9207	98.012	2.0	50.0%
KSAC	5502	98.813	2.0	50.0%
KUKI	4418	41.938	1.0	50.0%
1Q5	4050	64.440	1.0	50.0%
KVCB	4695	77.347	1.0	50.0%
KWLW	4510	88.660	1.0	50.0%

Figure 1: A display of reachable emergency landing sites for a disabled aircraft (small center triangle). In this example, sites are ranked according to distance and runway length.

actual travel could be in a straight line. Since turning increases risk, this artificially biases the search against paths that do not go 0, 45, or 90 degrees to the axes of the grid. The generation of probabilistic roadmaps is possible in this domain, although there is little guarantee that the space of potentially good paths will be systematically explored with this technique. Instead we have opted for a different, more systematic method of generating roadmaps that relies on the typical characteristics of obstacles in this domain. The planning search is then a modified A* that searches for paths of low risk in this roadmap.

In the sections that follow, we give an overview of the system architecture, describe how obstacle information is obtained, describe how we asses risk, and describe the details of our prototype path planning algorithm.

Architecture

The Emergency Landing Planner is one component of the Integrated Flight Planning and Guidance (IFPG) subsystem of the Integrated Resilient Aircraft Control (IRAC) Project (see Figure 2).

In the IFPG architecture, when some sort of damage or failure occurs that impairs the aircraft in some significant way, several things happen. First, the Adaptive Flight Control subsystem helps the pilot retain or regain control of the aircraft. While this is happening, the IFPG subsystem dynamically gathers data from the Integrated Intelligent Flight

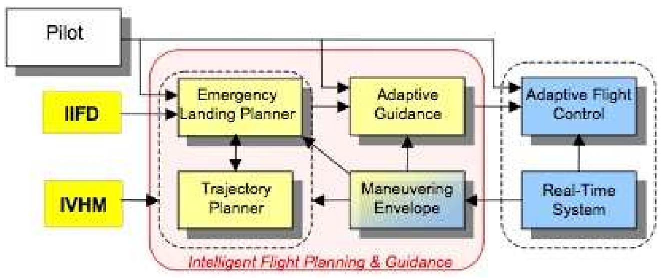


Figure 2: An overview of the IFPG Architecture in IRAC, including the Emergency Landing Planner.

Deck (IIFD) for airports and obstacles, Integrated Vehicle Health Management (IVHM) for aircraft health, and the Maneuvering Envelope subsystem for aircraft control limitations in order to construct the 3D planning problem to be solved by the Emergency Landing Planner.

As the Emergency Landing Planner finds usable solutions that do not violate any of the obstacle or controllability constraints, it consults the Trajectory Planner to refine these solutions into more detailed flight plans.[1] The pilot then

[1]The trajectory planner has a much more detailed but computationally more expensive model of aircraft performance and dynamics.

chooses from among the proposed solutions for the flight plan.

This IFPG emergency planning architecture allows for flexibility in the amount of autonomy delivered by the IFPG subsystem. The pilot can choose any of the solutions proposed by the IFPG subsystem based on experience, and on the information and predictions delivered by the IIFD, IVHM, and Adaptive Flight Control subsystems.

The planning problem to be solved consists of the following:

1. The start state, consisting of the current position, altitude and velocity of the aircraft.
2. The control envelope for the damaged aircraft, including airspeed range, allowable bank angles, descent range, and control responsiveness.
3. The potential landing sites within the estimated landing range of the aircraft, together with the characteristics of those landing sites, such as urban density, runway length, weather conditions and emergency facilities.
4. All of the "obstacles" that must be considered while flying to any landing site. Some of these may be hard obstacles like terrain, but others may be regions with weather conditions that simply present increased risk.

Within these constraints, the Emergency Landing Planning searches, based on explicit modeling preferences, for the best solutions that can be found. These are then presented to the pilot (see Figure 1) as possible landing sites.

Obstacles

To determine the best routes and landing sites the planner has to consider a set of dynamic and static *obstacles* as part of planning. These obstacles are derived from various online sources. There are five rough categories of obstacle data which, in the current work, cover the continental US, northern Mexico and southern Canada. These obstacles consist of:

- Terrain elevations
- Urban development
- Radar observations (thunderstorms)
- Special use airspaces and temporary flight restrictions
- Icing and turbulence observations

These physical and administrative phenomena, when they are relevant to the planning process, generate corresponding *obstacles*. The obstacles can be either *hard* or *soft* and are columnar in nature. They are generated by taking the 2D boundary of the phenomenon, forming a *convex hull* around it, and recording the *floor*, *ceiling* and *risk* associated with the obstacle.

The terrain elevations are used to generate hard "terrain obstacles" for the area within the landing range of the aircraft. The obstacles are derived from data gathered by the Navy Fleet Numerical Oceanography Center (FNOC) at Monterey, CA.[2] This data consists of a grid of elevations at $10'$ intervals, for both latitude and longitude.

[2]http://www.ngdc.noaa.gov/mgg/global/relief/ETOPO5/TOPO

Urban development data also comes from the FNOC data and is used as an estimation of population density along the approach path for each possible landing site. Areas of high population density are soft obstacles, corresponding to regions of increased risk (for persons and property on the ground) when the aircraft is at low altitude within 10 miles of the landing site.

The radar observations being used are the standard weather precipitation observations as reported by the National Weather Service (NWS). They are grouped into *cells*, *lines*, and *areas* of rain showers and thunder storms. They are retrieved at incident time[3] and updated dynamically as needed. They are reported as a location, size, direction of movement, speed, maximum elevation, severity, and, in the case of lines and areas, a saturation percentage. These observations are used to generate soft obstacles with risk determined by saturation, severity, and controllability of the aircraft.

Special Use Airspaces (SUAs) and Temporary Flight Restrictions (TFRs) are, as their names imply, areas where flight is restricted or prohibited. The SUAs are derived from an FAA publication[4] and are relatively static. They consist mainly of airspace reserved for military purposes, and range in size from a couple of nautical miles to covering significant portions of a state. SUAs are given as descriptions of an area boundary, a designated floor and ceiling, and times during which the given restriction is in force.

TFRs are similar to SUAs, but generally designate non-military restrictions. They consist of such events as air shows, major sporting events, large public gatherings, fire fighting activity, rescue operations, volcanic activity, VIPs, security restrictions, and so on. However, they are far more dynamic than SUAs, and are therefore retrieved and updated dynamically from the FAA[5]. They are represented in much the same way as SUAs, with a boundary, floor, ceiling, and enforcement times. Since an aircraft in distress is allowed to violate airspace restrictions, we treat SUA and TFR obstacles as soft, with risk associated with military and/or law enforcement activity.

Pilot Reports (PIREPs) of icing and turbulence are generated continually, and depending on severity, constitute regions of higher risk. PIREPs are retrieved dynamically[6] and updated frequently. Each icing and turbulence report consists of a location, floor, ceiling, intensity, and time of observation. Turbulence and icing reports generate soft obstacles with risk dictated by severity and controllability of the airplane.

All of the obstacles described above are *columner* in nature – that is, they have a 2D boundary, a floor and a ceiling. We represent them using a convex polygon together with their floor and ceiling altitudes, and an associated risk.

[3]From http://www.atmos/albany.edu/weather/data1/surface/rad

[4]"Special Use Airspace", JO 7400.8P, http://faa.gov/airports_airtraffic/air_traffic/publications

[5]http://tfr.faa.gov/tfr2/list.html

[6]http://weather.aero/pireps

Assessing Risk

There is risk associated with various phases of an emergency landing – en route, approach, landing, and emergency response. We use *expected loss of life* as our measure of risk, because it allows us to take into consideration casualties on the ground, as well as passengers and crew. It is difficult to give precise estimates of risk, given all the uncertainty associated with aircraft capabilities, pilot performance and weather. However, our purpose here is simply to provide a rough means of ranking alternatives for presentation to the pilot, and to allow the pilot to focus on the most critical factors affecting the decision on landing site.

En Route Risk

The primary factors influencing risk en route to the landing site are: controllability of the aircraft, distance and time to the site, complexity of the flight path (e.g. number of turns), weather along the path (thunderstorms, icing, and turbulence) and risk of further deterioration in aircraft performance and handling. The en route portion of the flight is generally at sufficiently high altitude (in the US) that terrain is not an issue.

We represent controllability in terms of the probability of loss of control for different flight regimes. Initially, we consider:

P_{stable} : we assume that the probability of *not* losing control while traversing an edge decreases exponentially with the length of the edge. That is, the probability of keeping control is $(1 - P_{stable})^D$, where $P_{stable} \in (0;1)$ and D is the distance traversed in nautical miles (nm). Intuitively, P_{stable} represents the probability of failing at each infinitely small step of the traverse or, loosely speaking, the probability of failure *per nm*.

P_{turn} as the probability of loss of control when a turn is initiated.

Thus for a flight path of total length D with T turns, the risk would be:

$$Risk = B\left(1 - \left(\overline{P_{stable}}\right)^D \left(\overline{P_{turn}}\right)^T\right)$$

where $\overline{P_X} \equiv (1 - P_x)$ and B is the total number of crew and passengers.

Risk associated with weather is also assumed to follow an exponential law with respect to the length traversed, but of course it depends on the severity of the weather, as well as the controllability of the aircraft. Initially, we take the probability of loss of control per nautical mile due to weather to be:

$$P_W = 1 - \left(\overline{P_{stable}} * \overline{P_w}\right)^S$$

where S is the severity of the thunderstorm, icing, or turbulence, with 1 being light, and 5 being extreme. Initially, we have chosen P_w, the inherent risk per nm associated with light weather to be .1. For a normally functioning transport aircraft, ($P_{stable} \approx 0$), this means that the chance of loss of control when flying through extreme thunderstorms, icing or turbulence is $.59/nm$. This may prove to be too high, but for now, it biases the planner away from routes through significant weather obstacles.

Risk of further deterioration in aircraft capabilities is difficult to assess. Ultimately, we assume that this will be given to us by the control and diagnostic systems, and will depend on the nature of the damage or failures. For now, we assume that the probability of loss of control due to further degradation, P_{degr}, is relatively low, but non-zero. Specifically, we assume $P_{degr} = .02/nm$. This introduces an additional mild bias for selecting shorter routes and closer landing sites.

Using all these factors (P_{stable}, P_{turn}, P_W and P_{degr}) we can assess the risk of any given route whether it goes through or around weather obstacles.

Approach Risk

When the aircraft reaches the approach environment (low altitude in the vicinity of the landing site) several additional risk factors come into play: urban development along the approach path, ceiling and visibility, wind shear, and risk due to configuration changes (deployment of flaps, slats, landing gear, etc.). At present, we ignore wind shear and configuration changes, but allow for the fact that the probability of loss of control per nautical mile may be different in the approach configuration. We refer to this probability as P_{appr} and assume that it is provided to us.

In general, the probability of loss of control on the approach depends on weather conditions as well as the controllability in the approach configuration. Assuming the aircraft navigation equipment is functioning normally, there is little additional risk associated with approaches where the ceiling is at least 1000 ft above the ground. However, if the ceiling is below 200 ft, the risk is high.[7] We take the probability of loss due to the ceiling as:

$$P_{Ceil} = \begin{cases} 0 & \text{if } Ceil \geq 1000 \\ 1 - \frac{Ceil-200}{800} & \text{if } 1000 > Ceil > 200 \\ 1 & \text{if } Ceil \leq 200 \end{cases}$$

Similarly, there is little risk when the visibility is greater than 3 miles, and extreme risk when it is less than half a mile. We therefore take the probability of loss due to ceiling as:

$$P_{Vis} = \begin{cases} 0 & \text{if } Vis \geq 3 \\ 1 - \frac{Vis-.5}{2.5} & \text{if } 3 > Vis > .5 \\ 1 & \text{if } Vis \leq .5 \end{cases}$$

We assume that loss of control of a transport category aircraft over a densely populated area will cause loss of life within at least a .1 square mile area. Thus, if the approach path takes us a distance D over population density N the risk is:

$$Risk = (.1N + B)\left(1 - \overline{P_{Ceil}} * \overline{P_{Vis}} * \left(\overline{P_{appr}}\right)^D\right)$$

Thus, if controllability is still high (e.g. loss of one engine), and weather is good, there is little bias against approach paths over heavily populated areas. However, if controllability or weather is poor, there is additional bias for low population density approach paths.

[7] Actually, we use the published Decision Height (DH) or Minimum Descent Altitude (MDA) for the approach. DH is normally 200ft for a category I precision approach.

Runway Risk

The primary risk factors associated with runway choice are runway length, width, braking condition, and relative wind. In general, the length required is determined by landing speed, braking condition and relative wind. Landing speed may be much higher than normal if controllability is low, but we assume this is given to us as part of the control envelope. As a general rule, and aircraft needs about 40ft of runway for each knot of speed at touchdown, thus:

$$\textit{Runway-reqd} \approx 40 * (\textit{Approach-speed} - \textit{Headwind})$$

This can go up by as much as a factor of two if braking quality is poor (water, snow or ice on the runway). It can also go up considerably if pitch or speed controllability is poor, so these two factors need to be added to the above equation. If runway length is sufficient, as computed above, there is no additional risk; risk increases as the runway length is reduced below that. We therefore compute probability of loss of life due to runway overrun as:

$$P_{\textit{length}} = 1 - \frac{\textit{Rnwy-length}}{\textit{Rnwy-reqd}}$$

when *Rnwy-reqd* $>$ *Rnwy-length*. Thus, if the runway length is zero, the result is considered equivalent to a crash. Additional factors are required if the runway width is low, or if the crosswind is too high, given aircraft controllability. So overall risk due to the runway can be assessed as:

$$\begin{aligned} P_{\textit{rnwy}} &= 1 - \overline{P_{\textit{length}}} * \overline{P_{\textit{width}}} * \overline{P_{\textit{xwind}}} \\ \textit{Risk}_{\textit{rnwy}} &= B * P_{\textit{rnwy}} \end{aligned}$$

Airport Risk

Finally, there is risk associated with the availability of emergency facilities at the airport and in the surrounding community. If facilities are absent this may result in greater loss of life if the aircraft loses control on landing, or runs off the end of the runway. If we represent emergency facilities as being a number between zero (no facilities) and 1 (good facilities) we can estimate the runway risk as follows

$$\textit{Risk}_{\textit{rnwy}} = B * \left(1 - \left(\overline{P_{\textit{rnwy}}}\right)^{(2-\textit{facil})}\right)$$

Thus, if emergency facilities are good, no additional risk is incurred, but if they are poor, the risk increases somewhat.

Taken together, all of this risk information for route, approach, runway and airport facilities allow us to evaluate different possible emergency landing plans. Currently, we are not actively using all of this information to guide the search for good paths, but this is an obvious next step.

Path Planning

The role of path planning is to determine the best path from the current position of the plane to any candidate landing site. There has been extensive previous work on path planning and obstacle avoidance. See (Choset *et al.* 2004) for a survey of this field. Our research in the IRAC project lead us to experiment with several approaches, including cell decomposition and roadmaps. In this paper, we report on our current approach, which is a roadmap algorithm.

Roadmaps

A roadmap is a topological representation of the environment that captures the connectivity of the free space. Formally, it is a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where vertices $v \in \mathcal{V}$ represent specific locations in the environment and edges $e \in \mathcal{E}$ represent free paths between neighboring locations. We assume that a vertex $v_0 \in \mathcal{V}$ representing the starting state is included in the graph, as well as a vertices $v_g \in \mathcal{V}$ representing goal locations. Each edge $e = (v, v') \in \mathcal{E}$ is associated with a distance or cost $d(v, v')$. In most of our exposition, we assume that $d(v, v')$ is the Euclidean distance between v and v'. More interesting cost functions are discussed in the end of this section.

Several types of roadmaps have been proposed in the literature. In this work, we adapt one of the earliest and most common techniques: the *visibility graph*. This graph is defined in two-dimensional space. Remember that obstacles are represented as 2D polygons with an associated floor, ceiling and risk. For the purpose of building the initial visibility graph, we consider all obstacles above a certain risk level, neglect the floors and ceilings of the obstacles, and use only their two-dimensional polygonal representation.[8] The nodes $\mathcal{V}$ of the visibility graph include: the start location v_0, the destination v_g, and all the obstacle vertices (corners between two edges of the polygons). The edges $\mathcal{E}$ are straight lines between vertices that do not traverse any obstacle (see Fig. 3). In 2D, the visibility graph is guaranteed to contain the shortest path from the start to the goal. Unfortunately, this property does not hold if the same approach is applied in higher dimensions.

The *reduced visibility graph* or *tangent graph* is a subgraph of the visibility graph that is also guaranteed to contain the the shortest 2D path. Because it contains fewer edges, it is easier to solve. It is based on the observation that the shortest path traverses only edges that are tangent to obstacles. Therefore, non tangent edges may be safely removed from $\mathcal{E}$ (see Fig. 3). Determining the set of tangent edges can be a difficult problem (Liu & Arimoto 2004). Instead of computing this set exactly, we eliminate edges whose extremities are not "locally tangent" to an obstacle. Consider an edge e arriving in vertex v, which represents the angle between two sides s and s' of the polygonal obstacle. Then, e is locally tangent in v if s and s' fall on the same side on the line passing through e. A tangent edge may not contain an extremity that is not locally tangent (but the converse is not true). Therefore, we can safely eliminate edges with an extremity that is not locally-tangent. This eliminates fewer edges than the real tangent graph, however, as testing for local tangency is very cheap (Cormen *et al.* 2001), it is a good overall compromise.

So far, we have limited our reasoning to a two-dimensional framework. In our emergency landing framework, obstacles have a floor and a ceiling, and it is sometimes possible to fly above or below some of them. To account for this possibility, the tangent graph is augmented

[8]In order to account for aircraft dynamics and turn radius, we expand the size of each obstacle by an amount determined by aircraft controllability.

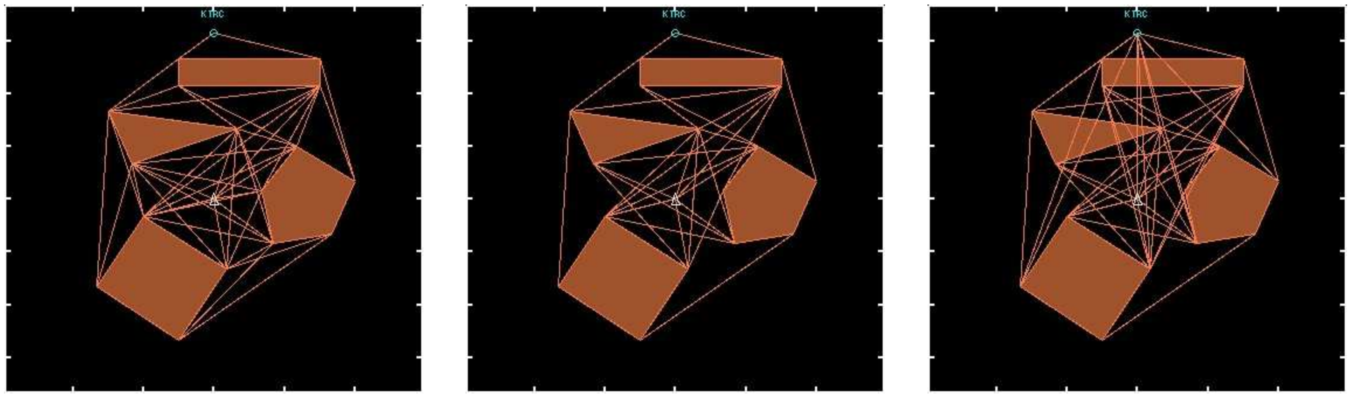


Figure 3: Three types of roadmaps (from left to right): visibility graph (58 edges), tangent graph (45 edges), and extended tangent graph (69 edges). The plane is represented by the small triangle in the center of the figure, and the targeted landing site is the blue circle labeled "KTRC" in the top of the figure.

with a set of edges and vertices called *secondary*.

To build secondary edges, we first consider connecting v_0 to v_g. We enumerate all obstacles that intersect the segment between these two vertices, and all ground-level variations along this segment. As shown in Fig. 4, the path between v_0 to v_g is divided in a series of *slices* inside of which the ground level is constant and the same set of obstacles is traversed. We represent this cut through the 3D space by chain of (secondary) vertices and edges: there is one secondary edge for each slice in Fig. 4, and one secondary vertex between each two consecutive slices.

Similar operations are repeated for connecting different pairs of vertices in the 2D map:

- v_0 or v_g to any corner of an obstacle, if the segment intersects another obstacle and does not have any non-locally tangent extremity;
- two corners of different obstacles, if the segment intersects a third obstacle and does not have any non-locally tangent extremity.

Next, *primary* edges (those originally present in the tangent graph) are replaced by chains of secondary edges to account for ground level variations along those edges (if the ground level is constant along the edge, it is left untouched).

We call the resulting graph an *extended tangent graph* (see Fig. 3). We can associate with each edge e of $\mathcal{E}$ the set of altitudes that are free of obstacles for all points in e. It is represented as a finite set of altitude intervals and denoted $\mathcal{U}_e = \{(f_i, c_i), i = 1, 2, \ldots, k_e\}$. Notations f and c stand for *floor* and *ceiling* as intervals (f_i, c_i) represent *tunnels* along e through which the plane can navigate. If e traverses no obstacle, then $\mathcal{U}_e$ contains a single interval ranging over all possible altitudes. For instance, slice 3 in Fig. 4 contains three tunnels and slice 8 only one tunnel.

The extended tangent graph contains more edges than the tangent graph, and often contains more edges than the visibility graph. Although it is not guaranteed to contain the shortest path in 3D space, it allows for some possibilities of movement that are not represented in the 2D graphs, such as going above or below an obstacle. In the next paragraph, we

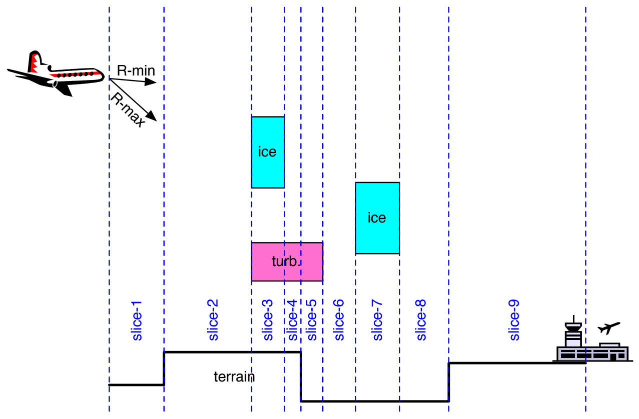


Figure 4: Cut of the 3D space along the segment from the plane (v_0) to the targeted landing site (v_g). The cut is divided into 9 slices where ground elevation and obstacles are constant. Each slice is represented by a secondary edge in the extended tangent graph. Secondary vertices's are added between each two consecutive secondary edges.

show how to determine, given the vertical maneuverability of the plane, what trajectories are possible in the 3D cut of Fig. 4.

Hybrid A*

The extended tangent graph $\mathcal{G}$ is a 2D topological map that accounts for some opportunities of movement in the 3D space, such as going under or above an obstacle. In our prototype planner, this graph is exploited by a 3D path planning algorithm called Hybrid A* (HA*). HA* is an extension of the A* algorithm that can handle a form of continuous state variables. It can also be seen as a deterministic special case of the HAO* algorithm (Mausam *et al.* 2005; Meuleau *et al.* 2008). Table 1 summarizes the relations between these algorithm.

The basic principle of HA* is to reason about hybrid states $s = (v, h)$, where v is a vertex of $\mathcal{G}$ and $h \in \mathbb{R}$.

	discrete states	hybrid states
deterministic	A*	HA*
stochastic	AO*	HAO*

Table 1: Relationship between algorithms.

Being in state (v, h) represents being at location v and altitude h. States are called hybrid because they have a discrete component n and a continuous component h. The 3D path planning problem is formalized as the problem of finding a sequence of hybrid states leading from the plane to the targeted runway which are all particular hybrid states. An important characteristic of this problem is that there are (uncountably) infinitely many states that are reachable from a given position, which must be accounted for during optimization.

HA* addresses this issue by computing finite partitions of the (infinite) hybrid-state space. Then it performs standard A* search in the space of partitions. The partitions are build on the fly, as the search progresses.

Formally, HA* associates with each vertex v of $\mathcal{G}$ a finite set of intervals $\mathcal{R}_v = \{(l_i, u_i), i = 1, 2, \ldots, k_v\}$ representing the altitudes at which v can be reached from the current position of the plane. For convenience, we denote an altitude interval by the triple $[v, l, u]$ where v is the edge of $\mathcal{G}$ to which the interval is attached, and l and u are the bounds on altitude.

The sets $\mathcal{R}_v$, $v \in \mathcal{V}$, may be computed incrementally, by propagating altitude intervals into $\mathcal{G}$. An interval I_0 is created to represent the initial situation of the plane: if v_0 is the vertex representing the plane and h_0 is the current altitude, then $I_0 = [v_0, h_0 - \epsilon/2, h_0 + \epsilon/2]$, where ϵ is any very small positive number. This seed is added to $\mathcal{R}_{v_0}$ and then pushed through every edge starting in v_0, which creates new altitude intervals that are propagated through the graph in turn.

Given an edge $e = (v, v')$ of length $d(v, v')$ and a tunnel $(f, c) \in \mathcal{U}_e$, *pushing* an altitude interval $[v, l, u]$ through (f, c) consists of computing a new altitude interval $[v', l', u']$ such that:

$$\begin{aligned} u' &= \min\{\min\{u, c\} - \rho_n d(v, v'),\ c\} \\ l' &= \max\{l - \rho_x d(v, v'),\ f\} \end{aligned} \qquad (1)$$

where $\rho_x > 0$ is the maximum descent rate of the plane and ρ_n its minimum descent rate. (If the plane can still climb, then $\rho_n < 0$.) The first line of Eqn. 1 may be understood as follows: if the set of reachable altitudes in v is (l, u), then when we enter the tunnel (f, c) leading from v to v', our maximum altitude is $\min\{u, c\}$. During the traversal of the tunnel, our altitude may increase by the amount $-\rho_n d$ at most (which may be positive or negative depending on ρ_n). Finally, our altitude when we exit the tunnel may not exceed c. The second line of Eqn. 1 is derived from a symmetric equation:

$$l' = \max\{\max\{l, f\} - \rho_x d, f\}$$

taking into account the fact $\rho_x > 0$. Note that if the new interval is inconsistent ($l' > u'$), then it is discarded.

Algorithm 1 Hybrid-A* for 3D shortest path

```
1: // Initialization
2: Create $I_0 = [v_0, h_0 - \epsilon/2, h_0 + \epsilon/2]$ representing the plane. Set $g(I_0) = 0$. Compute $h(I_0)$ as the Euclidean distance from $v_0$ to $v_g$. Set $f(I_0) = g(I_0) + h(I_0)$. Add $I_0$ to OPEN.
3: // Main Loop
4: while OPEN $\neq \emptyset$ do
5:    Pick $I = \arg\max_{I' \in \text{OPEN}} [f(I')]$.
6:    if the target is included in $I$ then
7:       return(success).
8:    end if
9:    Remove $I$ from OPEN, add $I$ to CLOSED.
10:   for all edges $e = (v, v') \in \mathcal{G}$ such that $g(I) + d(v, v')$ is lesser than the plane range do
11:      for all tunnels $(f_i, c_i) \in \mathcal{U}_e$ do
12:         Compute interval $I' = [v', l', u']$ obtained by pushing $I$ through $(f_i, c_i)$ using Eqn. 1.
13:         if $I'$ is consistent ($l' <= u'$) then
14:            Set $g(I') = g(I) + d(v, v')$. Compute $h(I')$ as the Euclidean distance from $v'$ to $v_g$. Set $f(I') = g(I') + h(I')$.
15:            MERGE($I'$).
16:         end if
17:      end for
18:   end for
19: end while
20: return(failure).
```

Pushing an altitude interval (l, u) through an edge e of $\mathcal{G}$ consists of pushing it through every tunnel of $\mathcal{U}_e$, and then taking the union of the resulting intervals. It creates at most as many interval as there are tunnels in $\mathcal{U}_e$. This operation is the basis of the reachability analysis performed by HA*.

The HA* algorithm is presented in Alg. 1. It is a very similar to standard A*, the main difference being that vertices $v \in V$ are replaced by altitude intervals. It impacts the algorithm in the following ways:

- The OPEN and CLOSED lists contain altitude intervals instead of nodes. The algorithm stops when the target *is included in* the most promising interval picked from OPEN (line 6).
- Instead of listing all possible successors of a node, HA* pushes an interval through an edge to get all its successor intervals (lines 11 to 13).
- When a new node is created, regular A* goes through a series of tests to check whether this node is already present in OPEN or CLOSED. In HA*, the situation is more complex because a newly created interval I may intersect partly with intervals in OPEN, partly with intervals in CLOSED, and partly with none of the two. Therefore, the test is replaced by a call to the function MERGE, described in Alg. 2. This procedure ensures that each hybrid state included in the new interval I receives the same treatment as discrete states in standard A*.

HA* exhibits the same convergence properties as A*, that

Algorithm 2 MERGE(I), where $I = [v, l, u]$

```
1: // Intersection with intervals in OPEN
2: for all intervals $I' = [v, l', u'] \in$ OPEN such that $(l, u) \cap (l', u') \neq \emptyset$ do
3:     Call $I''$ the interval of $v$ defined by $(l, u) \cap (l', u')$.
4:     if $g(I) < g(I')$ then
5:         Replace $I'$ with $I' \setminus I''$ in OPEN.
6:         Set $g(I'') = g(I)$ and $h(I'') = h(I)$.
7:         Add $I''$ to OPEN.
8:     end if
9: end for
10: // Intersection with intervals in CLOSED
11: for all intervals $I' = [v, l', u'] \in$ CLOSED such that $(l, u) \cap (l', u') \neq \emptyset$ do
12:     Call $I''$ the interval of $v$ defined by $(l, u) \cap (l', u')$.
13:     if $g(I) < g(I')$ then
14:         Replace $I'$ with $I' \setminus I''$ in CLOSED.
15:         Add $I''$ to OPEN.
16:         Set $g(I'') = g(I)$ and $h(I'') = h(I)$.
17:     end if
18: end for
19: // Intervals intersecting with none of OPEN and CLOSED
20: Compute the set difference of $I$ and every interval in OPEN and CLOSED. The result is a set of altitude intervals called $\Delta$.
21: for all intervals $I' = [v, l', u'] \in \Delta$ do
22:     Add $I'$ to OPEN.
23:     Set $g(I') = g(I)$ and $h(I') = h(I)$.
24: end for
25: delete($I$);
```

is, it is guaranteed to be optimal given the heuristic is admissible. A proof of this statement is obtained by adapting the proof of convergence of HAO* (Meuleau *et al.* 2008) to the particular case of deterministic problems.

Status and Future Work

The path planning algorithm described above focuses on minimizing the Euclidean length of the path to the target in the 2D map. For our application, this is not the best way to generate good candidate routes. Instead we would like to generate routes that minimize the total risk as described earlier. The HA* algorithm is very general and can be used with multiple sorts of cost. In the case of a probabilistic cost (like risk) this requires assigning costs to edges based on the logarithm of the probability of retaining control for the edge. The hardest part is to find an efficient heuristic when costs no longer have Euclidian structure.

We are currently working on extending the algorithm so that traversing certain obstacles becomes an option. Of course, this will generally incur a risk penalty. Technically, this amounts to adding more edges to the visibility graph, and defining a rule for pushing altitude intervals through an obstacle, in such way that the cost function is constant over each resulting interval.

Ultimately, the Emergency Landing Planner and the other components shown in Figure 2 will be integrated and tested using a full motion 767 simulator on a number of different damage models and scenarios. At present, the Emergency Landing Planner calls the detailed trajectory planner but otherwise functions as standalone software. In doing the integration, there are a number of user interface issues associated with presenting alternatives to the pilot, and allowing the pilot to change the ranking criteria. Ultimately this will require input from pilots and other experts on human factors in the cockpit.

Acknowledgments

This work was supported by the Intelligent Resilient Aircraft Control program of the NASA Aeronautics Research Mission Directorate. We thank John Kaneshige for help with the FLTZ trajectory planning software, and the full motion simulator. We thank United Captain Mietek Steglinski for discussion on the factors most relevant to deciding between alternative emergency landing sites.

References

Burcham, F. W.; Maine, T. A.; Fullerton, C. G.; and Webb, L. D. 1996. Development and flight evaluation of an emergency digital flight control system using only engine thrust on an F-15 airplane. Technical Report TP-3627, NASA.

Burken, J. J., and Burcham, F. W. 1997. Flight-test results of propulsion-only emergency control system on MD-11 airplane. *J. Guidance, Controls and Dynamics* 20(5):980–987.

Choset, H.; Lynch, K.; Hutchinson, S.; Kantor, G.; Burgard, W.; Kavraki, L.; and Thrun, S. 2004. *Principles of Robotic Motion: Theory, Algorithms, and Implementation*. Cambridge, MA: MIT Press.

Cormen, T.; Leiserson, C.; Rivest, R.; and Stein, C. 2001. *Introduction to Algorithms, Second Edition*. Cambridge, MA: MIT Press. Chap. 33.

Gundy-Burlet, K.; Krishnakumar, K.; Limes, G.; and Bryant, D. 2004. Augmentation of an intelligent flight control system for a simulated C-17 aircraft. *JACIC* 1(12):526–542.

Liu, Y., and Arimoto, S. 2004. Computation of the tangent graph of polygonal obstacles by moving-line processing. *IEEE Trans. on Robotics and Automation* 823–830.

Mausam; Benazera, E.; Brafman, R.; Meuleau, N.; and Hansen, E. 2005. Planning with continuous resources in stochastic domains. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, 1244–1251.

Meuleau, N.; Benazera, E.; Brafman, R.; Hansen, E.; and Mausam. 2008. A heuristic approach to planning with continuous resources in stochastic domains. *Journal of AI Research*. To appear.

NTSB. 1990. Aircraft accident report – United Airlines flight 232. Technical Report NTSB/AAR-90/06, National Transportation Safety Board.