Automated Design of Noise-Minimal, Safe Rotorcraft Trajectories

Robert A. Morris
NASA Ames Research Center, CA (USA)
robert.a.morris@nasa.gov

K. Brent Venable
University of Padova, Italy
kvenable@math.unipd.it

James Lindsey
Monterey Technologies, CA)
NASA Ames Research Center(USA)
james.e.lindsey@nasa.gov

ABSTRACT

NASA and the international community are investing in the development of a commercial transportation infrastructure that includes the increased use of rotorcraft, specifically helicopters and aircraft such as a 40-passenger civil tilt rotors. Rotorcraft have a number of advantages over fixed wing aircraft, primarily in not requiring direct access to the primary fixed wing runways. As such they can operate at an airport without directly interfering with major air carrier and commuter aircraft operations. However, there is significant concern over the impact of noise on the communities surrounding the transportation facilities. In this paper we propose to address the rotorcraft noise problem by exploiting powerful search techniques coming from artificial intelligence, coupled with simulation and field tests, to design trajectories that are expected to improve on the amount of ground noise generated. This paper investigates the use of simulation based on predictive physical models to facilitate the search for low-noise trajectories using a class of automated search algorithms called *local search*. A novel feature of this approach is the ability to incorporate constraints into the problem formulation that addresses passenger safety and comfort.

INTRODUCTION

The ability to predict rotorcraft ground noise is important in determining and assessing environmental noise impact. The noise generated by rotorcraft can limit their usage and restrict operations, particularly near cities and populated regions. The two primary approaches commonly used to reduce rotorcraft noise are to make vehicle design modifications and to make changes in operational flight procedures. The latter have the advantage that they can often be implemented to achieve significant noise reductions at a lower cost than new design efforts.

The problem motivating this work is the design of low-noise approach trajectories for rotorcraft in order to reduce surrounding community noise. This is an important component in developing a transportation infrastructure that is based on an increased use of rotorcraft, specifically helicopters and aircraft such as a 40-passenger civil tilt rotor. Rotorcraft have a number of advantages over fixed wing aircraft, primarily in not requiring direct access to the primary fixed wing runways. As such they can operate at an airport without directly interfering with major air carrier and commuter aircraft operations. There is significant concern over the impact of noise on the communities surrounding the transportation facilities. One way to address the rotorcraft noise problem is to automatically design flight profiles which can be evaluated with respect to noise in simulation or through field tests.

Computer modeling capabilities for developing low noise procedures have received much attention over the last 15 years. These models, when paired with an automated optimization approach, can facilitate the design of new approach trajectories for improving the environmental impact.

The objective of this paper is to address the *Trajectory Noise Optimization Problem* (TNOP), as introduced in (Ref. 8), for designing noise minimal rotorcraft approach trajectories. The model includes a graphical representation of the computational search space based on the state of the aircraft and the control decisions made by the pilot; a representation of constraints that identify trajectories that are 'flyable' based on pilot-elicited rules of comfort and safety of the aircraft; a noise simulator tool (Rotorcraft Noise Model, RNM) for calculating the effects of sound propagation over varying ground terrain, enabling the quantitative assessment of the overall ground noise produced by a given trajectory; cost functions that aggregate and quantify the cumulative noise level to allow for trajectories to be compared and ordered based on the noise they produce; and an optimizing search approach using local search. The local search uses a neighborhood function based on a simple exchange of control decisions. The search is initialized using a seed solution manually crafted by a pilot based on standard approach procedures.

BACKGROUND

Noise and how it is Measured

Noise is unwanted sound. Sound is variation in air pressure detectable by the human ear in the form of vibration of the ear drum. The decibel is a ratio that compares the sound pressure of the sound source of interest (e.g., the rotorcraft overflight) to a reference pressure (the quietest sound we can hear). Humans can detect sound pressure over a wide range, 10^{-9} to 10^{-3} pounds per square inch (psi). Because the range of sound pressures is very large, we use logarithms to simplify the expression to a smaller range, and express the resulting value in decibels (dB).

Presented at the American Helicopter Society 68th Annual Forum, Fort Worth, Texas, May 1–3, 2012.

Sound can be broken down into frequencies (low, medium, high). The ear is more sensitive to mid- and high frequency sounds, so we find noise in these ranges more annoying. The process of *A-weighting* approximates the sensitivity of the human ear and helps to assess the relative loudness of various sounds.

Sound levels vary with time, which is important if we are interested in the noise associated with a certain event of interest (e.g. an approaching rotorcraft). To take exposure duration into account, the most common measure is the *Sound Exposure Level (SEL)*. *SEL* 'summarizes' the variable energy level of an event with arbitrary duration by mapping it to an event of one second duration with the same overall energy and a constant energy level. *SEL* provides a comprehensive way to describe noise events for use in modeling and comparing noise environments. Computer noise models base their computations on *SEL* values.

The US Federal Aviation Administration (FAA) considers a 1.5 dB the minimum significant change where cumulative exposure is above 65 DNL. Any abatement strategy that promises over 5 dB change in noise level is considered definitely beneficial. As we show later, we will use this value in assessing and comparing noise cost functions for trajectories.

Helicopter noise sources include the main rotor, the tail rotor, the engine(s), and the drive systems. The most noticeable acoustical property of helicopters is the modulation of sound by the relatively slow-turning main rotor. The resulting sound can become impulsive in character and is referred to as BVI (Blade Vortex Interaction Noise). Impulsive noise occurs during high-speed forward flight as a result of blade thickness and compressible flow on the advancing blade. This causes the blades airloads to fluctuate rapidly. These fluctuations result in impulsive noise with shock waves that can propagate forward. At lower airspeeds, and typically during a descent, rotor impulsive noise can occur when a blade intersects its own vortex system or that of another blade. This type of noise is BVI noise. When this happens, the blade experiences locally high velocities and rapid angle-of-attack changes. This tends to produce a sound that is loud and very annoying in character (Ref. 1), (Ref. 6).

Rotorcraft Noise Model Simulation Tool

The Rotorcraft Noise Model (RNM) (Ref. 4) is a simulation program that predicts how the sound of a rotorcraft will propagate through the atmosphere and accumulate on the ground. RNM is capable of calculating cumulative noise exposures such as A-weighted *SEL*. Given a flight trajectory and other parameters describing the rotorcraft and the environment, RNM simulation produces predictive noise data in various formats. Of interest here is the generation of *ground noise contour plots*: a set of values representing ground noise exposure using A-weighted *SEL* over a designated grid of x-y points around the evaluated trajectory. Figure 2 shows an example of such a plot, where each color corresponds to a dB level (redder and lighter colors noisier). These plots provide the data used to compute the aggregate cost functions used during search. The number of data points computed to generate such a contour is a tunable parameter in RNM: we call this parameter *data resolution*. Specifically, data resolution sets the distance between two arbitrary data points; a higher distance means fewer data points, more 'gaps' to fill in with the same value. The result is a coarser measure of noise, but because of the fewer values the simulator runs faster. By contrast, more data points means a higher resolution prediction of noise, but RNM runs at an (exponentially) slower rate as resolution increases. This isn important consideration in this work, since higher resolution means in general less time for the optimizer to search for a high-quality solution, thus diminishing its performance.

The input to RNM consists of

- a set of computational parameters, including identity of rotorcraft, and the dimensions and resolution of a grid that will display output noise (discussed further below);
- a specification of points of interest; and
- a specification of the flight trajectory, including position, velocity and orientation.

RNM combines a model of sound propagation through the atmosphere with a database of noise data either experimentally or analytically generated. The database is comprised of a set of *sound spheres*. Points on the sphere are described in terms of a radius from the source and two spherical angles. A sphere is associated with one noise source and one flight condition (flight path angle, nacelle angle (for tilt-rotors) and airspeed). There may be more than one sphere for the same flight condition; for example, spheres for different locations on the rotorcraft. Figure 1 shows an example sound hemisphere.

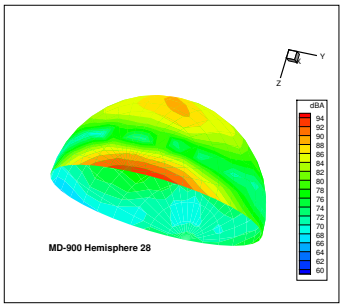

Fig. 1. A sound hemisphere of an MD-900 helicopter.

There are three main computational components of the RNM simulation:

- *Input Module*: Linear interpolation over the input trajectory as a pre-processing step. Input data are interpolated (if required) to a default of 2 second spacing. The user may specify other time increments if desired.

- *Source Database Lookup and Selection*: Selecting and interpolating over the sound spheres to determine the best representative of the noise generating for a given location and flight condition in the input trajectory; and
- *Source to receiver propagation*: Accumulating and storing the sound for a given receiver.

The second and third components are executed for each trajectory point, sound source, flight operation and receiver location.

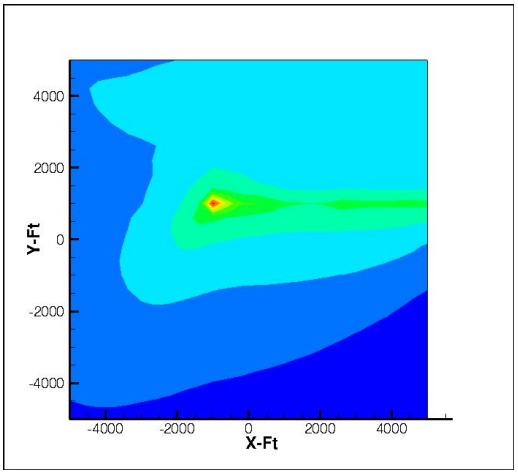

Fig. 2. A Noise Contour Plot.

Trajectory Optimization

The field of trajectory optimization has a long history, with many applications in aerospace and robotics. Informally, a trajectory optimization problem consists of a set of *states*, a vector of *control decisions*, a start and goal state, a cost function, and a set of constraints. A state represents locations (i.e. points in a 3D space), velocity and heading. A control decision is a vector representing change in velocity, altitude, heading, and in turn radius.

In addition to noise, trajectories have been optimized with respect to time, fuel, path length and obstacle avoidance. Methods of solving trajectory optimization problems range from numerical methods (Ref. 3) to non-linear programming problems (Ref. 5) or dynamic programming (Ref. 7). In addition, path planning methods from robot motion planning has been used (Ref. 9). Randomized optimization methods such as simulated annealing and genetic algorithms have also been applied in the work by Xue and Atkins (Ref. 11). The latter bears the most similarity to the work described here, but has a number of important differences. There, the search space in modeled with a k-ary tree approach where each branch represents a change in the value of a parameter (e.g. path angle and acceleration) and the branching factor is restricted to at most k. We, instead, consider box-shaped trajectories, inspired by standard flying practices, which have a more restricted shape but yet cannot be modeled in the framework used in (Ref. 11). Moreover, the noise produced by a trajectory is evaluated in (Ref. 11) using a verified noise database, whereas we use RNM as an evaluation tool. Finally, the local search techniques employed are different, as we use a standard hill-climbing procedure whereas in (Ref. 11) simulated annealing was used.

The optimization problem of interest here, which we call the *Trajectory Noise Optimization Problem (TNOP)*, is stated informally as follows: *given a set of states and control actions, find a path (trajectory) that minimizes expected ground noise subject to a set of dynamic constraints, and constraints on start- or end-states.*

THE TRAJECTORY NOISE OPTIMIZATION PROBLEM

We focus on approach trajectories because that is virtually where all the community noise problems arise and the problem for take-off trajectories is very similar. We will focus on A-weighted SEL as our noise exposure metric. RNM simulation provides a black box scoring function for candidate trajectories. Specifically, RNM produces an output file that assigns predicted noise for a set of ground points arranged in a two-dimensional grid on the X-Y plane. The grid size is defined in terms of the values of the corner nodes and the distance between nodes.

Upon this grid our model superimposes an organization of *nodes* associated with the state of the aircraft and the control decisions being made by the pilot. We introduce state variables X, Y, Z, V, H and associated domains for, respectively, location (X, Y), altitude (Z), airspeed (V), and heading (H). We use normal conventions for heading, whereby 0 is north, 90 is east, 180, south, and 270 west. Given a state variable Q we write q, to refer to domain elements of the variable. A *state* of the system is a 5-tuple $s = \langle x, y, z, v, h \rangle$.

Similarly, we introduce decision variables $\Delta V, \Delta Z, \Delta H, \Delta R$ for change in velocity, change in altitude, change in heading, and change in turn radius, also with associated domains, and we write Δv to denote a value in the domain of ΔV, etc. Change in heading involves addition modulo 360, one action to initiate the change (e.g. $\Delta H = 180$ to start a 180 degree turn) and a complementary action to come out of the turn (e.g. $\Delta H = -180$ to restore straight flight). A decision vector (or simply decision) is a tuple d of values for each decision variable.

A *node* is a pair $\langle s, d \rangle$ of a state and decision, representing the state of the rotorcraft when the pilot or automated system begins to apply decision d. Given node $N_i = \langle s_i, d_i \rangle$, we will denote with $\langle x_i, y_i, z_i, v_i, h_i \rangle$ and $\langle \Delta v_i, \Delta z_i, \Delta h_i, \Delta r_i \rangle$ its components.

A path (trajectory) is a sequence of k nodes. Between two adjacent nodes N_j, N_{j+1} there is an edge labeled with the distance flown $dist_j$ (in feet), between the locations corresponding to the nodes. For a turn, it measures the portion of the circumference of the circle flown. A *consistent* path is one in

which, for all $j = 1 \ldots k-1$, node N_{j+1} is the result of applying d_j at s_j for the entire length $dist_j$. We express this as a transition function $T : N \rightarrow N$, where N is the set of nodes.

We assume we are given two nodes designated as start and finish, with fixed state and control vectors and that a solution is any consistent flyable trajectory between them. To control the size of this space we initially start by limiting the paths to those that would be considered 'standard' by pilots. One example of a standard approach is a box pattern, as the one shown in Figure 3. This trajectory is represented by 6 nodes, $N_0 \ldots N_5$, where two 90^o turns start, respectively, at N_2 and N_3. The goal is to find an assignment $(s_0, \ldots, s_5, d_0, \ldots, d_5)$ to the state and control vectors of the nodes not fixed by initial and final conditions, such that the noise simulated by RNM on the corresponding trajectory is minimal.

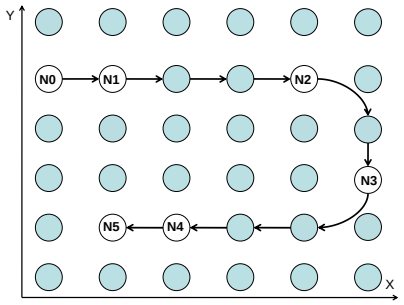

Fig. 3. A "box"-like approach pattern.

Flyability Constraints

Conditions that make a trajectory suitable to fly are usually expressed in terms of constraints over the glide slope angle and deceleration. In particular, any part of a trajectory should be characterized by an angle of descent $\gamma \in [0^o, 12^o]$ and a deceleration $a \in [0g, 0.1g]$ (or $a \in [40ft/sec^2, 201ft/sec^2]$). Such restrictions induce constraints on the change of velocity and altitude as follows. Given a pair of nodes N_i, N_j and a path between them of distance $dist_{ij}$ we have:

- the deceleration constraint (dec): $\Delta v_i \in \{\delta_v \| \exists a \in [0, 0.1], \delta_v = \sqrt{v_i^2 + 2a \times dist_{ij}} - v_i\}$, where a is expressed in gs.
- the angle-of-descent constraint (aod): $\Delta z_i \in \{\delta_z \| \exists \gamma \in [0^o, 12^o], tan(\gamma) = \frac{\delta_z}{dist_{ij}}\}$.

In addition, there are a minimal velocity and altitude (v_{mini}, z_{mini}) that a rotorcraft must have when starting the final part of the approach (that is at the, so called, landing decision point). Such values are a function of the distance of the landing decision point from the landing site. A trajectory is said to be *flyable* if it satisfies all the deceleration and angle-of-descent constraints along its path, and does not violate the bounds defined by v_{mini} and z_{mini}.

A *Trajectory Noise Optimization Problem* (TNOP) is a tuple $\langle S, D, s_0, s_f, aod, dec, v_{mini}, z_{mini} \rangle$, where S is a set of states, D s set of decisions, s_0, s_f are initial and final states, aod, dec are deceleration and descent angle constraints, and v_{mini} and z_{mini} are as just defined. A feasible solution to a TNOP is a path $P = N_0, N_1, \ldots, N_k$ where $N_0 = \langle s_0, d_0 \rangle$, $N_k = \langle s_f, 0_d \rangle$, where 0_d represents the decision of leaving everything unchanged, and for all $j = 2 \ldots k$, $N_j = T(N_{j-1})$, and where P satisfies the flyability constraints.

Cost Functions

We introduce two natural ways of 'aggregating' RNM contour noise data into scalar valued functions. One cost function identifies ranges of values that correspond to various levels 'high', 'medium' and 'low' noise, and creates 'bins' that store the number of grid noise data points in that range. Each bin is assigned a weight indicating its importance in determining solution quality, and the trajectory is evaluated as the weighted sum of the bin values.

Formally, we define a *Binning Heuristic function* (*Bin*) as follows. Given in input a solution t, RNM computes the A-weighted SEL value for each of the grid points. Let us denote with $SEL(t,x,y)$ such a value for the grid point (x,y) given trajectory t. We define a sequence of decreasing ranges, $\langle r_1, r_2, \ldots, r_n \rangle$ partitioning the SEL values of the grid points. Given a trajectory t let us denote by $S_i(t) = \{(x,y) | SEL(t,x,y) \in r_i\}$. We define the following vector $b(t) = \langle b_1(t), b_2(t), \ldots, b_n(t) \rangle$ where $b_i(t) = |S_i(t)|$. The bin-score of solution t is $Bin(t) = \Sigma_{i=1\ldots n} w_i b_i(t)$ where w_i is the weight associated to the i-th bin, $w_i > w_{i+1}$ and $\Sigma_{i=1,\ldots,n} w_i = 1$. Thus a solution that assigns lower levels of noise to larger regions of the grid is to be preferred. Weights are used to penalize the presence of, even small, extremely noisy regions. The goal will be that of minimizing the *Bin* value.

The other cost function is based on ordering two candidate solutions based on a notion of 'significant difference' in their predicted noise values. One noise data point is significantly different from another if the human ear can detect a change in the noise. Counting the number of significantly different pair of noise values for the same point between two solutions, we can generate a partial ordering of the candidates.

Formally, we define a *Significant Improvement Heuristic function* ($Diff$) as follows. Let s denote a reference solution and t another solution. Then the significant improvement score of t w.r.t. s is

$$Diff(s,t) = |\{(x,y) \| SEL(t,x,y) - SEL(s,x,y) \geq 1.5dB\}| - |\{(x,y) \| SEL(s,x,y) - SEL(t,x,y) \geq 1.5dB\}|.$$

In other words, this heuristic function considers a reference solution (that, in our case will be seed solution of the local search), and then scores all other solutions counting the number of grid points where they produce a noise that is at least 1.5dB lower than the one produced by s at the same point. A 1.5dB threshold has been identified to be the smallest improvement that can be perceived by a human. The intuition

behind this heuristic function is that of promoting solutions that improve significantly in the largest number of grid points. Given this heuristic function the goal is to minimize its value.

LOCAL SEARCH FOR TNOP

The technique we propose here to solve the optimization problem described in the previous section is a hill-climbing local search approach. The reasons for preferring local search include:

1. *Anytime performance:* On average, local search behaves well in practice, yielding low-order polynomial running times (Ref. 2). Since the trajectory space is large, it is difficult *a priori* to characterize globally preferred solutions. Consequently, we are interested in a system that can examine large parts of the search space quickly.

2. *Flexibility and ease of implementation:* deployment-related deadlines suggest the use of techniques which are easy to implement.

3. *Simulator Compatibility:* running RNM is heavy from a computational point of view. This means that the repetitive evaluation of partial trajectories, required by complete incremental solving paradigms (e.g. Branch and Bound), may be unacceptably time consuming. Local search, on the other hand, only requires the evaluation of complete solutions.

Figure 4 describes the pseudocode of our algorithm, which we call *Box-TNOP-HC*. The inputs to the algorithm are

- a randomly generated seed solution σ_{seed};
- a scoring function *score* that can be either *Bin* or *SI*;
- a positive integer *threshold*, representing the number of search steps after which the execution must terminate.

The technique we propose here to solve the optimization problem described in the previous section is a hill-climbing local search approach. Figure 4 describes the pseudocode of our algorithm, which we call *Box-TNOP-HC*.

The inputs to the algorithm are a seed solution σ_{seed}; a scoring function *score*; and a positive integer *threshold*, representing the number of search steps after which the execution must terminate. The output of *Box-TNOP-HC* is a solution denoted by σ_{best}. During the execution we keep track of the current solution, the neighborhood of which we are exploring, denoted by σ_{cur}, and the best flyable solution found so far, denoted with σ_{best}. Both such solutions are initially assigned the seed solution. Then, the algorithm starts exploring the neighborhood of σ_{cur}. As soon as it finds a solution that is better than the current one, it checks if it is flyable and if so it saves as the best incumbent. *Box-TNOP-HC* then updates σ_{cur} and starts scanning its neighborhood. Whenever no better solution is found, a random move in the neighborhood is taken.

Box-TNOP-HC(Trajectory σ_{seed}, function *score*, integer *threshold*)
$\sigma_{cur} = \sigma_{seed}$ // current trajectory
$\sigma_{best} = \sigma_{seed}$ // best incumbent trajectory
$step = 1$
do
  σ_0 = Neighbor(σ_{cur})
  $neighborhood(\sigma_{cur}) = neighborhood(\sigma_{cur}) \setminus \{\sigma_0\}$
  while $neighborhood(\sigma_{cur}) \neq \emptyset$ and $score(\sigma_0) \leq score(\sigma_{cur})$
    σ_0 = Neighbor(σ_{cur})
    $neighborhood(\sigma_{cur}) = neighborhood(\sigma_{cur}) \setminus \{\sigma_0\}$
  $\sigma_{cur} = \sigma_0$
  if $flyable(\sigma_{cur})$ and $score(\sigma_{cur}) > score(\sigma_{best})$
    $\sigma_{best} = \sigma_{cur}$
  $step++$
while $step \leq threshold$
return σ_{best}

Neighbor(Trajectory σ)
1 $n = random(\sigma)$ // randomly pick a node
2 $p = partner(n)$ // randomly choose partner for transfer
3 *select* $c \in \{\Delta v, \Delta z\}$ // randomly choose control variable
4 $v_c = val(c, p, n)$ // find an allowable value to transfer
5 $\sigma_n = transfer(n, p, v_c, \sigma)$ // add to n and subtract from p
6 $(n, p, c, v_c) = used$ // mark quadruple as used
return σ_n // return the neighbor

Fig. 4. Algorithm Box-TNOP-HC.

We note that the box trajectory is implicitly represented in σ_{seed}. Moreover, since in our case there is no way to test if an optimal solution as been found, the algorithm will always run for *threshold* number of steps.

The output of *Box-TNOP-HC* is a solution denoted by σ_{best}. During the execution we keep track of the current solution, the neighborhood of which we are exploring, denoted by σ_{cur}, and the best flyable solution found so far, denoted with σ_{best}. Both such solutions are initially assigned the seed solution. Then, the algorithm starts exploring the neighborhood of σ_{cur}. As soon as it finds a solution that is better than the current one, it checks if it is flyable and if so it saves as the best incumbent. *Box-TNOP-HC* then updates σ_{cur} and starts scanning its neighborhood. Whenever no better solution is found, a random move in the neighborhood is taken.

Neighborhood Function

The neighbor of a trajectory s is the result of applying one of two operators that alter the change of speed or altitude (ΔV, ΔZ) at two adjacent nodes of s. Figure 5 illustrates the general case where a node has two adjacent nodes with which to swap values.

More specifically, a node N_i is chosen at random to be the recipient of the transferred value. An node adjacent to N_i (i.e., N_{i-1} or N_{i+1}, called the *partner*) and a control variable, ΔV or ΔZ, are also chosen randomly. An amount $0 < \delta x_c \leq \Delta x_{i'}$ is then computed and *transferred* to N_i; that is, δx_c is added to the appropriate control variable in N_i and subtracted from the value of the partner. Note that given a trajectory with L nodes, $N_1, \ldots N_L$, no transference is possible for the final node, N_L.

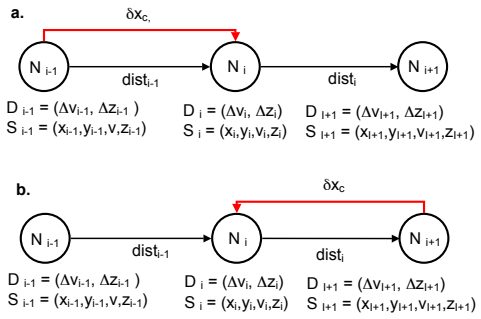

Fig. 5. Transferring values between adjacent nodes.

The first and $L-1$st nodes have only one partner; the rest have two.

The transfer value v_c to transfer must be chosen in a way to preserve the feasibility of the new trajectory. Intuitively, there are two considerations: first, if too much value is transferred to a node, the trajectory will force the pilot to ether descend or decelerate too quickly during the segment beginning at N_i, violating the limit constraints on these values. Second, if too much control is passed *backward* from N_{i+1} to N_i, then more deceleration is applied sooner, and if too much is transferred earlier the helicopter might end up flying too low or too slow at N_{i+1}. This test involves the lower bound values v_{mini} and z_{mini} defined earlier for the state at the partner node.

Moreover, since there is an infinite number of choices of values for v_c and changes in the sound levels occur only for large enough transfers, we have decided to consider only a fixed set of such values corresponding to relevant percentages (i.e. 25%, 50%, 75%, and 100%) .

Finally, the effects of the transference of control is propagated to the states of the relevant nodes. Specifically, if control is transferred forward to N_i, then the state of N_i is changed; if control is transferred backward to N_i, then the state of N_{i+1} changes.

EXPERIMENTS WITH LOCAL SEARCH

In previous work, (Ref. 8), we have summarized experiments conducted with local search; the interested reader should consult these references for details. These experiments consist in a number of comparisons, including:

- Conducting local search with pilot-generated initial solutions (seed) as well as randomly generated initial solutions;
- The cost functions defined earlier (*Bin* and $DIff$) in terms of their ability to discriminate among the different contour plots to produce the best aggregate;
- The tradeoff between different data resolution settings for RNM, exploring the trade between better resolution and local search exploration;
- Different refinements of the basic local search algorithm; for example, looking at a multi-phase search in which different data resolution settings were used.

The results were generally promising. Although it is difficult to summarize the overall improvement gained from optimization, due to the factors, such as data resolution, that influence the amount of improvement, such experiments showed roughly an average improvement of over 25% (using either *Bin* or $Diff$ as the cost function) over a random walk approach (in which solutions are simply randomly generated and evaluated). Furthermore, the best score found by local search methods often improved the random walk best score by 50%.

The ultimate goal of these experiments is to generate and verify feasible approach trajectories that are are expected to be quieter than 'standard' approach procedures for pilots. What constitutes 'standard' is, of course, to an extent subjective; we have queried more than one pilot for examples of approach trajectories they would consider quiet. One way to visualize this goal is to compare Figure 6 with Figure 7. The former shows the profile (velocity and altitude) of a pilot-defined quiet trajectory (here, the path is a straight line rather than a box), a trajectory actually flown in a recent set of experiments (Ref. 10). Also in the figure is the contour map generated by RNM for the trajectory. Figure 7 shows the result of applying simple local search starting with the pilot's preferred trajectory as the initial solution. The optimal profile differs significantly from the pilot's, preferring a more gradual decrease in velocity, but an earlier descent. The contour plots reveal the improvements over the pilot's trajectory in the form of a smaller reddish (most noisy) region, and larger green and light blue (moderately noisy) regions. We plan in the future to verify results such as these using real tests.

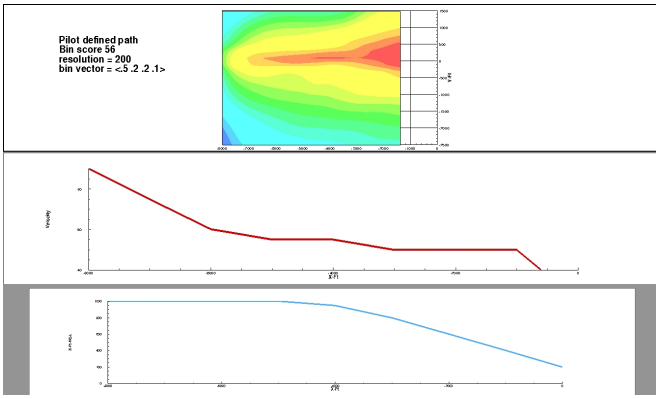

Fig. 6. 'Pilot Defined' Quiet Trajectory

SUMMARY AND FUTURE WORK

This work has explored the use of local search paired with a robust simulator to find approach trajectories that are low-noise. Local search is a simple, fast algorithm that consistently found trajectories that improved on standard approach

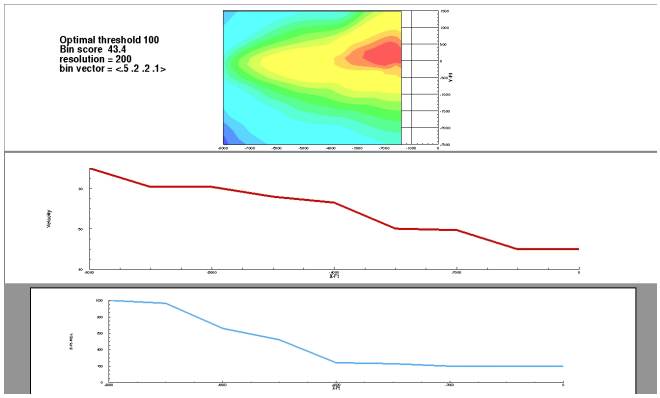

Fig. 7. Optimized Trajectory Using Local Search

trajectories. and allowed for a examination of procedural strategies during approach for reducing noise.

The results of this work, however, are limited in the following ways:

- employing an artificial grid model of the environment that ignores factors influencing good approach procedures aside from noise and comfort;
- searching over fixed 'box' structure trajectories;
- employing a simple dynamic control model that considered only the effects of reducing velocity or altitude and ignored the effects. e.g., of changes in angular momentum;
- using an incomplete search method, i.e., one that did not conduct a full search over the space of solutions.

Current work seeks to overcome these limitations. First, we are beginning to model real landing environments. Importing GIS data from airports, and visualizing them in NASA World Wind, we are constructing constraint maps based on land usage codes, allowing for paths to be generated that avoid sensitive areas such as hospitals. We're also looking into representing constraints involving air space, such as avoiding approach corridors for active fixed-wing runways. This will allow for the construction of a more sophisticated multi-attribute cost function that trades noise with other criteria such as land use and fuel consumption.

Second, we are looking at search over a wider area of feasible paths than those limited to the 'box' structure. Varying the X and Y values in this way expands the dimensionality of the search space and requires more sophisticated search. One class of techniques we're examining for exploring large search spaces is based on sampling methods such as Probabilistic Road Maps. At the same time, we are also considering complete path-planning methods in 3D such as A* and D*. Finally, we're building a full 6 Degree of Freedom (X,Y,Z,roll, pitch, yaw) dynamics model, which will result in a more robust causal model of the actions that influence noise.

REFERENCES

[1]Fly neighborly guide. Technical report, Helicopter Association International, 2009.

[2]E. Aarts and J. K. Lenstra. *Local Search in Combinatorial Optimization*. Princeton University Press, 1997.

[3]John T. Betts. Survey of numerical methods for trajectory optimization. *Journal of Guidance, Control and Dynamics*, 21(2):193–207, 1998.

[4]David A Conner, Casey L Burley, and Charles D Smith. Flight acoustic testing and data acquisition for the rotor noise model (rnm). In *Proceedings of the 62nd Annual Forum of the American Helicopter Society*, pages 1–17, 2006.

[5]Gaurav Goplan, Min Xue, Ella Atkins, and Frefric H Schmitz. Longitudinal-plane simultaneous non-interfering approach trajectory design for noise minimization. In *Proceedings of the 59th AHS International Forum and Technology Display*, pages 1–18, 2003.

[6]E. Greenwood and F. Schmitz. A parameter identification method for helicopter noise source identification and physics-based semi-empirical modeling. In *American Helicopter Society 66th Annual Forum*, Phoenix, AZ, May 11-13, 2010.

[7]P. Hagelauer and F. Mora-Camino. A soft dynamic programming approach for on-line aircraft 4d-trajectory optimization. *European Journal of Operational Research*, 107:87–95, 1998.

[8]R. Morris, K.B. Venable, and J. Lindsay. Simulation to support local search in trajectory optimization. In *Proceedings of the IEEE Conference on Aerospace Engineering*, Big Sky, Montana, March 3-11, 2012.

[9]Shen Z. P. Cheng and S. M. LaValle. rrt-based trajectory design for autonomous automobiles and spacecraft. *Archives of Control Sciences*, 11(3-4):167–194, 2001.

[10]M.E. Watts, R. Snider, E. Greenwood, and J. Baden. Flight test of the bell 430 helicopter. In *Proceedings of the 68th Annual American Helicopter Society Forum*. Fort Worth, TX, May 2012.

[11]Min Xue and Ella M Atkins. Terminal area trajectory optimization using simulated annealing. In *44th AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, 2006. AIAA.